Konputazio Zientziak eta Adimen Artifizialaren Saila

Departamento de Ciencias de la Computación e Inteligencia Artificial

Supervised classification in continuous domains with Bayesian networks

by

Aritz Pérez Martínez

Supervised by Pedro Larrañaga and Iñaki Inza

Dissertation submitted to the Department of Computer Science and Artificial Intelligenceof the University of the Basque Country as partial fulfilment of the requirements for the PhD degree in Computer Science

Acknowledgements

This dissertation would never have been possible if it were not for the support of my thesis supervisors Pedro Larrañaga and Iñaki Inza. Thank you very much for being my mentors, for your encouragement and patience.

I am grateful to Jose Antonio Lozano and my colleagues in Intelligent System Group: Alex Mendiburu, Borja Calvo, Carlos Echegoyen, Cristina Gonzaléz, Dinora Morales, Ekhiñe Irurozki, Guzmán Santafé, Jonathan Ortigosa, Jose Antonio Fernandes, Jose Antonio Pascual, Jose Luis Flores, Josu Galdiano, Juan Diego Rodriguez, Leticia Hernando, Ramón Sagarna, Roberto Santana, Rosa Blanco and Rubén Armañanzas for the working atmosphere, occasional advice and continuous support. Also to those that spent some time at the lab and contributed with their comments to enrich the work presented in this dissertation: Frederik Vincent, Robin Höns, Siddarta Shakya and Yvan Saeys.

I have a special thanks to Ivan Bratco and all the people in his group, specially to Matej Guid, Aleksander Shadikov and Martin Mozina, for hosting me during my stay at the Faculty of Computer and Information Sciences in Ljubljana.

I am grateful to the University of the Basque Country, the Basque Government, the Spanish Ministry of Science and Innovation and KB Data Mining S.L. for the financial support of the last years.

Finally, I would like to thank the support of my family, my girlfriend Juncal and my friends. This thesis is also for them.

Contents

List of Figures

List of Tables

Part I

Background

1

Supervised classification

1.1 Introduction

Supervised classification is one of the most important tasks in the field of pattern recognition. It has been widely applied to many real-world problems such as image, speech and document classification in entertainment, industrial, security, biomedical and financial domains, among others. Informally, supervised classification can be understood as learning to distinguish concepts from experience, e.g. learning to distinguish apples and bananas from a set of images of apples and bananas. Usually, the experience is represented by a set of examples (instances, cases or samples) of the given concepts, e.g. the available collection of images of bananas and apples. It is also possible to articulate the experience of the field by means of prior knowledge, but this is out of the scope of this dissertation. Supervised classification assumes the presence of a special variable called *class variable*, e.g. fruit type. In supervised classification, the class variable is discrete and it indicates the category (class label, e.g. apple) of a case (or instance) described in terms of its *features* (descriptors, predictor variables or input variables).

Supervised classification task involves the learning (or induction) of a *classification model* (classification function or classifier). A classifier is a function that assigns a class label to an unlabeled case described in terms of its features. The classifier is usually learned from a set of cases (training set, data set or samples) by means of a *classifier induction algorithm* (learning process, learning algorithm or inductor). Generally, in order to generate the *training set*, a set of samples representative of all the classes must be collected. Usually, these observations are a set of images, documents or sounds (signals in general) which must be processed in order to identify features (numerical or symbolical) that allow to distinguish the patterns of the different class labels. So we are looking for variables which provide discriminative information about the class of the samples. The task consisting of identifying the appropriate features is named *feature extraction*. For example, in order to distinguish apples from bananas using 2D images, the shape of the object or simply its color

could be considered as features with a great discriminative power. Although feature extraction is critical for supervised classification, it is out of the scope of this dissertation. After the most appropriate (discriminative) features are identified, the set of examples are quantified by obtaining the values for the features and the class variable. In other words, each example is summarized into the values of the previously identified features. This task is called *quantification* of the samples and, once the features are identified and defined, it can be considered straightforward.

A *sample* (instance, case or example) is an instantiation (quantification, realization or evaluation) of all the predictor variables, except for those with missing (or unknown) values. A *data set* is a set of cases. In supervised classification the training set has no missing class labels: the class label is known for all the instances in the training set. If there are unlabeled samples in the training set, the problem is named *semi-supervised classification*. A supervised classification *domain* is defined by a *generalized joint probability distribution*. A data set is supposed to be obtained by means of a random sampling of the generalized joint probability distribution of the domain and, thus, it has *independent and identically distributed* (iid) samples. We include as the first step in the classifier induction algorithm the preprocess of the data base. The preprocess usually consists of a feature reduction procedure, the imputation of missing values for the implied variables, and a discretization of the continuous features. Once a classifier is learned from the preprocessed training data by means of a classifier induction algorithm, it can be used for obtaining the class of a new unlabeled instance (a quantified sample with a missing value for the class).

Generally, before a classifier is applied to any real world domain, its performance is estimated. For example, the average misclassification rate of unlabeled instances can be estimated. Using the *performance estimation*, we can consider if a classifier has enough discriminative power for the concrete domain and, besides, we can compare different choices of classifiers in order to select the most suitable.

But, what is this dissertation about? This dissertation formally tries to superficially cover some of the most relevant issues of supervised classification, focusing attention on classifiers based on Bayesian networks for domains with continuous random variables. It should be highlighted that our work is focused on modeling the conditional densities of the continuous random variables directly, avoiding the discretization. For this purpose we base our work on two types of Bayesian networks: conditional Gaussian networks and kernel based Bayesian networks.

The document is divided into two parts: background and methodological contributions. The background part formally introduces fundamentals about probability theory, decision theory, generative, conditional and discriminative learning, estimation theory, error estimation, comparison and analysis of the error, measures of performance, information theory, curse of dimensionality and feature selection, density estimation and Bayesian multinomial networks.

In these fundamentals are included three algorithms for feature selection, three popular discretization algorithms, four procedures for the comparison of classifier induction algorithms for different experimental conditions, the most popular bias plus variance decomposition of the error, which can be used for analyzing the behavior of a classifier or the particularities of the domain, and the most popular procedures for error estimation.

And, what is a novel contribution? Two groups of novel methodological contributions can be identified in this dissertation:

- Supervised classification in mixed domains with probabilistic graphical models: we have adapted a set of algorithms taken from Bayesian multinomial networks to *conditional Gaussian networks*. We also have proposed novel classifier induction algorithms based on the particularities of conditional Gaussian networks. Moreover, we have proposed the novel *kernel based Bayesian network* paradigm which extends the idea of flexible naïve Bayes [John and Langley (1995)] breaking with the parametric assumptions. In addition, we have adapted some of the algorithms proposed for Bayesian multinomial networks to this novel paradigm. In order to present the kernel based Bayesian network paradigm, the mixed Gaussian kernel distribution is introduced.
- Information theory and classification error. Most of the filter algorithms presented in this dissertation are guided by the conditional mutual information. In order to analyze this score, we illustrate some of the intuitions which relate the classification error and the information theory. This intuitions can be helpful to understand the behavior of the presented family of filter classifier induction algorithms. In addition, we study the link between information theory and classification error to search novel scores for guiding the structural learning of the classifiers based on Bayesian networks. Based on this knowledge we analyze a discriminative version of tree-augmented naïve Bayes [Pernkopf and Bilmes (2005)]. Besides, we present a set of parametric (Gaussian) and non-parametric (kernel based) estimators of quantities based on information theory. The estimators proposed are used to guide the structural search of some of the classifiers induction algorithms introduced for conditional Gaussian network and kernel based Bayesian network paradigms.

1.2 Overview of the dissertation

This document concerns with Bayesian network paradigm for supervised classification and classifier induction algorithms in domains with continuous random variables avoiding the discretization. The document is divided into three main blocks: introduction, methodological contributions and conclusions.

The first block formally presents some of the most relevant concepts and procedures for supervised classification with Bayesian networks. This block

is divided into Chapters 1 and 2. Chapter 1 presents a formal framework for supervised classification. In Chapter 2 we introduce the Bayesian multinomial networks for supervised classification, indicating their main drawbacks when dealing with domains with both continuous and discrete random variables. This block includes concepts, procedures and tools for the correct understanding of the novel contributions of the dissertation.

The second block presents the main methodological contributions made by the author and is divided into Chapter 3, Chapter 4 and Chapter 5. In Chapter 3 we introduce conditional Gaussian networks for supervised classification, adapt a set of algorithms originally developed for Bayesian multinomial networks, introduce two novel algorithms, and perform an experimental analysis with all of them. In addition we provide a set of estimators for measures of information theory under Gaussian assumptions. Chapter 4 introduces the novel paradigm of kernel based Bayesian networks for supervised classification. We adapt a set of algorithms originally developed for Bayesian multinomial networks and perform an experimental analysis with them. Moreover, we introduce Gaussian kernel density function and mixed Gaussian kernel distribution and a set of results concerning their conditional and marginal forms. Besides, we provide a set of estimators for a set of measures of information theory based on the previously presented mixed Gaussian kernel distribution. Finally, In Chapter 5 we show the relation between information theory and the classification error. The motivation of this chapter consist of understanding the presented filter algorithms and finding alternative scores to conditional mutual information for guiding the filter classifier induction algorithms. Based on the concepts introduced in this chapter we analyze the discriminative structural learning algorithm for tree augmented naïve Bayes introduced in Pernkopf and Bilmes (2005).

The last block is composed of Chapter 6. This chapter summarizes the most important contributions of the dissertation, indicating their main future work lines.

1.3 Probability theory

This section formally introduces some of the main concepts of probability theory. Most of the theoretical results presented in this section have been adapted from [Peña (2001)]. Probability theory provides us with a formal framework in order to deal with uncertainty. The objective of this section is to present a set of definitions and theorems that gives us a mathematical interpretation of the concepts introduced through the dissertation, paying more attention to the semantics of probabilistic graphical models (see Chapter 2 for a formal introduction to probabilistic graphical models). Most of the notation that is used throughout this dissertation is also introduced in this section.

Definition 1.1 *Let Ω be the sample space of a random experiment, that is, the exhaustive set of mutually exclusive or disjoint possible outcomes of a random experiment. Ω is referred to as a discrete sample space when there is a countable number (finite or infinite) of distinct outcomes, otherwise Ω is referred to as continuous sample space.*

Given a random experiment, it is usual to be interested in a subset of outcomes rather than in a single one. The subsets of the sample space Ω of a given random experiment are referred to as *events*, ω. Thus, an event is a collection of outcomes of a random experiment. Each time that a random experiment is run, *a given event occurs when the outcome of the random experiment is an element of the event.* The sample space Ω of the random experiment at hand is itself the event that, by definition, always occurs. Alternatively, the empty set $\emptyset$ is the event that, by definition, never occurs.

Like most other mathematical theories, probability theory does not necessarily apply to the exhaustive collection of subsets of Ω. Usually the attention is restricted to collections of admissible subsets of Ω that are closed under certain sets of operations. Specifically, we usually impose the condition that any subset of Ω that can be constructed from a countable number of admissible subsets of Ω using a certain set of operations should itself be admissible. This leads to the following definition which plays an important role in probability theory.

Definition 1.2 *Let Ω be the sample space of a random experiment. A class $\mathcal{F}$ of subsets of Ω, i.e. events, is called a σ-algebra if it contains Ω itself and is closed under the formation of complements and countable unions, that is:*

1. *$\Omega \in \mathcal{F}$.*
2. *$\omega \in \mathcal{F}$ implies $\bar{\omega} \in \mathcal{F}$ where $\bar{\omega}$ is the complementary of ω.*
3. *$\{\omega_i \in \mathcal{F} | i \in I\}$ with I a countable index set implies $\bigcup_{i \in I} \omega_i \in \mathcal{F}$.*

In any random experiment with sample space Ω, we assume that the collection of interesting or admissible events $\mathcal{F}$ forms a σ-algebra. $\mathcal{F}$ is usually known as the *event space* of the random experiment.

Definition 1.3 *Let Ω be the sample space and $\mathcal{F}$ be the event space of a random experiment. A real-valued function on $\mathcal{F}$:*

$$Pr : \mathcal{F} \rightarrow \mathbb{R} \tag{1.1}$$

is called a probability measure if it satisfies the following conditions:

1. *$Pr(\omega) \geq 0$ for all $\omega \in \mathcal{F}$.*
2. *$Pr(\Omega) = 1$.*
3. *Given a set of events $\{\omega_i \in \mathcal{F} | i \in I\}$ with I a countable index and $\omega_i \cap \omega_j = \emptyset$ for all i and j such that $i \neq j$, $Pr(\bigcup_{i \in I} \omega_i) = \sum_{i \in I} Pr(\omega_i)$.*

Intuitively, the probability of an event is a measure of how likely the event will occur when the random experiment is run.

Definition 1.4 *Let Ω be the sample space and $\mathcal{F}$ be the event space of a random experiment. Let $Pr(\cdot)$ be a probability measure on $\mathcal{F}$. The triplet $(\Omega, \mathcal{F}, Pr)$ is called probability measure space or, simply, a probability space.*

Definition 1.5 *Let $(\Omega, \mathcal{F}, Pr)$ be an arbitrary probability space. A real-valued function on Ω*

$$X : \Omega \rightarrow \mathbb{R} \tag{1.2}$$

is called a unidimensional random variable.

The outcome of a random experiment may not be a number. A unidimensional random variable is a function that associates a numerical value with every outcome of a random experiment, in other words, is a quantification of an event. The variable is random in the sense that its values may vary from trial to trial as the experiment is repeated.

We follow the usual convention of denoting unidimensional random variables by an uppercase letter (or letters) and their values by the same letter (or letters) in lowercase.

Definition 1.6 *Let X be a unidimensional random variable. X is referred to as an unidimensional discrete random variable when there is a countable (finite or infinite) number of distinct values that X can have. Otherwise, X is referred to as unidimensional continuous random variable.*

Definition 1.7 *Let Z be a unidimensional discrete random variable. $p(z) = Pr(Z = z) = Pr(\{\omega \in \Omega | Z(\omega) = z\})$ is called a probability mass function for Z if it satisfies the following conditions:*

1. *$p(z) \geq 0$ for all values z of Z.*
2. *$\sum_z p(z) = 1$*

It is common to use the term probability distribution as a synonym of probability mass function. Strictly speaking, both terms are interchangeably used throughout this dissertation although the latter is more accurate than the former. Note also that $p(z)$ denotes the probability that $Z = z$ as well as a probability distribution for Z. Whether $p(z)$ refers to a probability or a probability distribution should be clear from the context.

Definition 1.8 *Let Y be a unidimensional continuous random variable. $f(y)$ is called probability density function if it satisfies the following conditions:*

1. *$f(y) \geq 0$ for all $y \in \mathbb{R}$.*
2. *$\int_{\mathbb{R}} f(y)dy = 1$*
3. *$Pr(\alpha \leq Y \leq \beta) = Pr(\{\omega \in \Omega | \alpha \leq Y(\omega) \leq \beta\}) = \int_{\alpha}^{\beta} f(y)dy$ for all α and β such that $\alpha \leq \beta$.*

Since unidimensional continuous random variables Y can take values in the continuum it no longer makes sense to talk about the probability that Y has a particular value y because the probability of any particular exact

value will almost always be zero. Rather, we talk about the probability that Y falls in some interval $[\alpha, \beta]$ with $\alpha \leq \beta$. Instead of having a probability mass function $p(y)$, we have a probability density function $f(y)$ which has the property that $Pr(y \in [\alpha, \beta]) = \int_{\alpha}^{\beta} f(y)dy$. If we consider a small interval $[\alpha, \beta]$ over which the probability density is constant, its probability is given by $Pr(y \in [\alpha, \beta]) = f(\alpha)(\beta - \alpha)$. The values taken by a probability density function are a measure of the probability mass per unit distance. Moreover, from Definition 1.8 we have that

$$f(y) = \lim_{h \to 0} \frac{1}{2h} Pr(y' \in [y - h, y + h]) \tag{1.3}$$

In general, most of the definitions and formulas for discrete random variables carry over to continuous random variables, replacing sums by integrals.

Definition 1.9 *Let $(\Omega, \mathcal{F}, Pr)$ be an arbitrary probability space. A function on Ω*

$$\boldsymbol{X} : \Omega \to \mathbb{R}^n \tag{1.4}$$

is called n-dimensional random variable.

One way to interpret an n-dimensional random variable $\boldsymbol{X} = (X_1, ..., X_n)$ is as an ordered set of n unidimensional random variables X_i for all i. Consequently, every ordered subset $\boldsymbol{Y} = (Y_1, ..., Y_m)$ of $\boldsymbol{X}$ may be seen as an m-dimensional random variable. This point of view over multidimensional random variables is very helpful in the remainder of this dissertation.

Note that we are using letter (or letters) in boldface uppercase $\boldsymbol{X}$ to designate a multidimensional random variable, and the same boldface lowercase letter (or letters) to denote an assignment of a value $\boldsymbol{x}$ to the multidimensional random variable.

Definition 1.10 *Let $\boldsymbol{X} = (X_1, ..., X_n)$ be an n-dimensional random variable. If there is a countable number (finite or infinite) of distinct values that $\boldsymbol{X}$ can have, i.e. every X_i is a unidimensional discrete random variable, then $\boldsymbol{X}$ is referred to as an n-dimensional discrete random variable. On the other hand, if every X_i is a unidimensional continuous random variable, then $\boldsymbol{X}$ is referred to as an n-dimensional continuous random variable. Finally, if there exist two proper subsets of $\boldsymbol{X}$, $\boldsymbol{Y} = (Y_1, ..., Y_m)$ and $\boldsymbol{Z} = (Z_1, ..., Z_{n-m})$, such that (i) $\boldsymbol{X} = \boldsymbol{Y} \cup \boldsymbol{Z}$ (exhaustive subsets) and $\boldsymbol{Y} \cap \boldsymbol{Z} = \emptyset$ (disjoint subsets), and (ii) $\boldsymbol{Y}$ is an m-dimensional continuous random variable and $\boldsymbol{Z}$ is an $m - n$-dimensional discrete random variable, then $\boldsymbol{X}$ is referred to as n-dimensional mixed random variable.*

In the remainder of this section we will limit our discussion to multidimensional random variables without loss of generality. The reason is that a unidimensional random variable represents a special case of multidimensional variables.

Definition 1.11 *Let $\boldsymbol{Z} = (Z_1, ..., Z_m)$ be an m-dimensional discrete random variable. $p(\boldsymbol{z}) = p(z_1, ..., z_m) = Pr(Z_1 = z_1, ..., Z_m = z_m) = Pr(\boldsymbol{Z} = \boldsymbol{z}) = Pr(\{\omega \in \Omega \mid \boldsymbol{Z}(\omega) = \boldsymbol{z}\})$ is called a joint probability mass function for $\boldsymbol{Z}$ if it satisfies the following conditions:*

1. *$p(\boldsymbol{z}) \geq 0$ for all value $\boldsymbol{z}$ of $\boldsymbol{Z}$.*
2. *$\sum_{\boldsymbol{z}} p(\boldsymbol{z}) = 1$.*

The term joint probability distribution is also used to refer to a joint probability mass function. Both terms are interchangeably used throughout this dissertation. However, strictly speaking, the latter is more correct than the former. Note also that $p(\boldsymbol{z})$ denotes the probability that $\boldsymbol{Z} = \boldsymbol{z}$, as well as a joint probability distribution for $\boldsymbol{Z}$. Whether $p(\boldsymbol{z})$ refers to a probability or a joint probability distribution should be clear from the context.

Definition 1.12 *Let $\boldsymbol{Y} = (Y_1, ..., Y_n)$ be an n-dimensional continuous random variable. $f(\boldsymbol{y}) = f(y_1, ..., y_n)$ is called a joint probability density function if it satisfies the following conditions:*

1. *$f(\boldsymbol{y}) \geq 0$ for all $\boldsymbol{y} \in \mathbb{R}^n$.*
2. *$\int_{\mathbb{R}} ... \int_{\mathbb{R}} f(y_1, ..., y_n) dy_1 ... dy_n = 1$.*
3. *$Pr(\boldsymbol{\alpha} \leq \boldsymbol{Y} \leq \boldsymbol{\beta})$*
   $= Pr(\alpha_1 \leq Y_1 \leq \beta_1, ..., \alpha_n \leq Y_n \leq \beta_n)$
   $= Pr(\{\omega \in \Omega | \alpha_i \leq Y_i(\omega) \leq \beta_i \text{ for all } i\})$
   $= Pr(\{\omega \in \Omega | \boldsymbol{\alpha} \leq \boldsymbol{Y}(\omega) \leq \boldsymbol{\beta}\})$
   $= \int_{\alpha_1}^{\beta_1}, ..., \int_{\alpha_n}^{\beta_n} f(y_1, ..., y_n) dy_1 ... dy_n$
   for all $\boldsymbol{\alpha} = (\alpha_1, ..., \alpha_n)$ and $\boldsymbol{\beta} = (\beta_1, ..., \beta_n)$ such that $\alpha_i \leq \beta_i$, $i = 1, ..., n$.

Definition 1.13 *Let $\boldsymbol{X} = (X_1, ..., X_{n+m})$ be an $(n+m)$-dimensional mixed random variable. If there exist two proper subsets of $\boldsymbol{X}$, $\boldsymbol{Y} = (Y_1, ..., Y_n)$ and $\boldsymbol{Z} = (Z_1, ..., Z_m)$, such that (i) $\boldsymbol{X} = \boldsymbol{Y} \cup \boldsymbol{Z}$ and $\boldsymbol{Y} \cap \boldsymbol{Z} = \emptyset$, and (ii) $\boldsymbol{Y}$ is an n-dimensional continuous random variable and $\boldsymbol{Z}$ is an m-dimensional discrete random variable, then $\rho(\boldsymbol{x}) = \rho(x_1, ..., x_{n+m})$ is called a generalized joint probability distribution for $\boldsymbol{X}$ if it satisfies the following conditions:*

1. *$\rho(\boldsymbol{x}) = \rho(\boldsymbol{y}, \boldsymbol{z}) \geq 0$ for all $\boldsymbol{y} \in \mathbb{R}^n$ and $\boldsymbol{z}$ of $\boldsymbol{Z}$.*
2. *$\int_{\mathbb{R}} ... \int_{\mathbb{R}} \sum_{z_1} ... \sum_{z_m} \rho(y_1, ..., y_n, z_1, ..., z_m) dy_1 ... dy_n = 1$.*
3. *$Pr(\boldsymbol{\alpha} \leq \boldsymbol{Y} \leq \boldsymbol{\beta}, \boldsymbol{\gamma} \leq \boldsymbol{Z} \leq \boldsymbol{\delta})$*
   $= Pr(\{\omega \in \Omega | \alpha_i \leq Y_i \leq \beta_i \text{ and } \gamma_j \leq Z_j \leq \delta_j, \ i = 1, ..., n \text{ and } j = 1, ..., m\})$
   $= Pr(\{\omega \in \Omega | \boldsymbol{\alpha} \leq \boldsymbol{Y} \leq \boldsymbol{\beta} \text{ and } \boldsymbol{\gamma} \leq \boldsymbol{Z} \leq \boldsymbol{\delta}\})$
   $= \int_{\alpha_1}^{\beta_1} ... \int_{\alpha_n}^{\beta_n} \sum_{\gamma_1 \leq z_1 \leq \delta_1} ... \sum_{\gamma_m \leq z_m \leq \delta_m} \rho(y_1, ..., y_n, z_1, ..., z_m) dy_1 ... dy_n$
   for all $\boldsymbol{\alpha} = (\alpha_1, ..., \alpha_n)$, $\boldsymbol{\beta} = (\beta_1, ..., \beta_n)$, $\boldsymbol{\gamma} = (\gamma_1, ..., \gamma_m)$ and $\boldsymbol{\delta} = (\delta_1, ..., \delta_m)$ such that $\alpha_i \leq \beta_i$, $i = 1, ..., n$, and $\gamma_j \leq \delta_j$, $j = 1, ..., m$.

Notice that $\rho(x_1, ..., x_{n+m}) = f(x_1, ..., x_n)$ if $m = 0$, and $\rho(x_1, ..., x_n) = p(x_1, ..., x_n)$ if $n = 0$.

For the sake of brevity, in the remainder of this section we treat multidimensional continuous, discrete or mixed random variables together without loss of generality. This means that all unnecessary references to the nature of the random variables (discrete, continuous or mixed) are omitted in the forthcoming definitions and theorems. Thus, the term joint probability mass function for $\boldsymbol{X}$, $p(\boldsymbol{x})$, and joint probability density function for $\boldsymbol{X}$, $f(\boldsymbol{x})$, are replaced by the more general term generalized joint probability distribution for $\boldsymbol{X}$, $\rho(\boldsymbol{x})$, which does not reflect the nature of $\boldsymbol{X}$. In order to particularize the definitions and theorems that appear below to multidimensional discrete, continuous, or mixed random variables, it is enough to consider the nature of the random variable involved.

Definition 1.14 *Let $\rho(\boldsymbol{x})$ be a generalized probability distribution for an n-dimensional random variable $\boldsymbol{X} = (X_1, ..., X_n)$. Let $\boldsymbol{U} = (U_1, ..., U_m)$ and $\boldsymbol{V} = (V_1, ..., V_{n-m})$ be two proper subsets of $\boldsymbol{X}$ such that $\boldsymbol{X} = \boldsymbol{U} \cup \boldsymbol{V}$ and $\boldsymbol{U} \cap \boldsymbol{V} = \emptyset$. Let $\boldsymbol{Y} = (Y_1, ..., Y_l)$ and $\boldsymbol{Z} = (Z_1, ..., Z_{m-l})$ be two subsets of $\boldsymbol{U}$ such that (i) $\boldsymbol{U} = \boldsymbol{Y} \cup \boldsymbol{Z}$ and $\boldsymbol{Y} \cap \boldsymbol{Z} = \emptyset$, and (ii) $\boldsymbol{Y}$ is an l-dimensional continuous random variable and $\boldsymbol{Z}$ is an $(m-l)$-dimensional discrete random variable. Then,*

$$\rho(\boldsymbol{v}) = \int_{\mathbb{R}} ... \int_{\mathbb{R}} \sum_{z_1} ... \sum_{z_{m-l}} \rho(y_1, ..., y_l, z_1, ..., z_{m-l}, v_1, ..., v_{n-m}) dy_1, ..., dy_l \tag{1.5}$$

is a generalized joint probability distribution for $\boldsymbol{V}$, which is referred to as the generalized marginal joint probability distribution *for $\boldsymbol{V}$.*

Again, it should be noted that in the theorem above $\boldsymbol{Y} = (Y_1, ..., Y_l)$ and $\boldsymbol{Z} = (Z_1, ..., Z_{m-l})$ are two subsets of $\boldsymbol{U}$ but not necessarily two proper subsets. In other words, either $\boldsymbol{Y}$ or $\boldsymbol{Z}$ could be empty.

Definition 1.15 *Let $\boldsymbol{X} = (X_1, ..., X_n)$ and $\boldsymbol{Y} = (Y_1, ..., Y_m)$ be two multidimensional random variables. Then,*

$$\rho(\boldsymbol{x}|\boldsymbol{y}) = \frac{\rho(\boldsymbol{x}, \boldsymbol{y})}{\rho(\boldsymbol{y})} \tag{1.6}$$

denotes the generalized conditional joint probability distribution for $\boldsymbol{X}$ given $\boldsymbol{Y} = \boldsymbol{y}$.

Note that the generalized conditional joint probability distribution $\rho(\boldsymbol{x}|\boldsymbol{y})$ is only defined in the case that $\rho(\boldsymbol{y}) > 0$.

Theorem 1. *Let $\boldsymbol{X} = (X_1, ..., X_n)$ be an n-dimensional variable. Then,*

$$\rho(\boldsymbol{X}) = \rho(x_1, ..., x_n) = \prod_{i=1}^{n} \rho(x_i | x_1, ..., x_{i-1}) \tag{1.7}$$

The proof of the theorem is trivial by considering the recursive application of Definition 1.15. Theorem 1 is also known as the *chain rule of probability*.

Theorem 2. *Let* $\boldsymbol{X} = (X_1, ..., X_n)$ *and* $\boldsymbol{Y} = (Y_1, ..., Y_m)$ *be two multidimensional random variables. Then,*

$$\rho(\boldsymbol{x}|\boldsymbol{y}) = \frac{\rho(\boldsymbol{y}|\boldsymbol{x})\rho(\boldsymbol{x})}{\rho(\boldsymbol{y})} \tag{1.8}$$

The proof of the theorem can be derived from Definition 1.15. Theorem 2 is also known as *Bayes' theorem*.

Definition 1.16 *Let* $\boldsymbol{X} = (X_1, ..., X_n)$ *and* $\boldsymbol{Y} = (Y_1, ..., Y_m)$ *be two multidimensional random variables.* $\boldsymbol{X}$ *and* $\boldsymbol{Y}$ *are independent if*

$$\rho(\boldsymbol{x}|\boldsymbol{y}) = \rho(\boldsymbol{x}) \tag{1.9}$$

for all $\boldsymbol{Y} = \boldsymbol{y}$.

Note that if $\boldsymbol{X}$ and $\boldsymbol{Y}$ are not independent, then they are dependent. The independence assertion that states that $\boldsymbol{X}$ and $\boldsymbol{Y}$ are independent is represented by $CI(\boldsymbol{X}; \boldsymbol{Y})$ (or equivalently $\boldsymbol{X} \perp \boldsymbol{Y}$) for the sake of brevity.

Definition 1.17 *Let* $\boldsymbol{X} = (X_1, ..., X_n)$, $\boldsymbol{Y} = (Y_1, ..., Y_m)$ *and* $\boldsymbol{Z} = (Z_1, ..., Z_l)$ *be three multidimensional random variables.* $\boldsymbol{X}$ *and* $\boldsymbol{Y}$ *are conditionally independent given* $\boldsymbol{Z}$ *if*

$$\rho(\boldsymbol{x}|\boldsymbol{y}, \boldsymbol{z}) = \rho(\boldsymbol{x}|\boldsymbol{z}) \tag{1.10}$$

for all $\boldsymbol{Y} = \boldsymbol{y}$ *and* $\boldsymbol{Z} = \boldsymbol{z}$.

Consequently, note that if $\boldsymbol{X}$ and $\boldsymbol{Y}$ are not conditionally independent given $\boldsymbol{Z}$, then they are conditionally dependent given $\boldsymbol{Z}$. As we are concerned with probabilistic graphical models, we are essentially more interested in conditional independencies than in conditional dependencies between random variables (see Section 2.2.1 for further details).

The conditional independence assertion that states that $\boldsymbol{X}$ and $\boldsymbol{Y}$ are conditionally independent given $\boldsymbol{Z}$ is represented by $CI(\boldsymbol{X}; \boldsymbol{Y}|\boldsymbol{Z})$ (or equivalently $\boldsymbol{X} \perp \boldsymbol{Y}|\boldsymbol{Z}$ for the sake of brevity). On the other hand, the conditional dependence assertion that states that $\boldsymbol{X}$ and $\boldsymbol{Y}$ are conditionally dependent given $\boldsymbol{Z}$ is represented by $CD(\boldsymbol{X}; \boldsymbol{Y}|\boldsymbol{Z})$. It should be noticed that (in)dependence is a particular case of conditional (in)dependence in which $\boldsymbol{Z} = \emptyset$. In the literature, these kinds of statements are simply known as independence statements. However, the term conditional independence statement seems to be more appropriate than the term independence statement as the latter is a special case of the former. A set of conditional independence statements is referred to as a *conditional independence model* (see Chapter 2). In the literature, conditional independence models are simply referred to as independence models. It should be noted that every generalized joint probability distribution for

an n-dimensional random variable $\boldsymbol{X} = (X_1, ..., X_n)$ encodes a conditional independence model that can be obtained by means of the definition of conditional independence. Also, the reader should note that every generalized joint probability distribution for $\boldsymbol{X}$ consists of a complete, i.e. *qualitative*, as well as *quantitative*, description of $\boldsymbol{X}$. However, the conditional independence model induced by a generalized joint probability distribution for $\boldsymbol{X}$ only represents the qualitative description of $\boldsymbol{X}$ [Castillo et al. (1997)].

Theorem 3. *Two multidimensional random variables $\boldsymbol{X} = (X_1, ..., X_n)$ and $\boldsymbol{Y} = (Y_1, ..., Y_m)$ are independent, i.e. $CI(\boldsymbol{X}; \boldsymbol{Y}|\emptyset)$, if and only if*

$$\rho(\boldsymbol{x}, \boldsymbol{y}) = \rho(\boldsymbol{x})\rho(\boldsymbol{y}) \tag{1.11}$$

Theorem 4. *Two multidimensional random variables $\boldsymbol{X} = (X_1, ..., X_n)$ and $\boldsymbol{Y} = (Y_1, ..., Y_m)$ are conditionally independent given a third multidimensional random variable $\boldsymbol{Z} = (Z_1, ..., Z_l)$, i.e. $CI(\boldsymbol{X}; \boldsymbol{Y}|\boldsymbol{Z})$, if and only if*

$$\rho(\boldsymbol{x}, \boldsymbol{y}|\boldsymbol{z}) = \rho(\boldsymbol{x}|\boldsymbol{z})\rho(\boldsymbol{y}|\boldsymbol{z}) \tag{1.12}$$

for all $\boldsymbol{Z} = \boldsymbol{z}$.

The proofs of both theorems can be directly obtained from the definitions and theorems that have been introduced above.

1.4 Supervised classification

This section formally defines supervised classification by means of the concepts described in the previous section.

An experimental domain, $\mathcal{D}$, is determined by its associated probability measure space, $(\Omega, \mathcal{F}, Pr)$. In order to model an experimental domain $\mathcal{D}$, the domain is quantified in terms of a set of random variables $\boldsymbol{X}$. This is the feature extraction step, and it is performed using expert knowledge. A *classification domain* is a domain defined in $(\boldsymbol{X}, \boldsymbol{C})$, where $\boldsymbol{C}$ and $\boldsymbol{X}$ are two multidimensional random variables. $\boldsymbol{C} = (C_1, ..., C_d)$ are said to be the class variables (or classes) and $\boldsymbol{X} = (X_1, ..., X_n)$ are the predictor variables (predictors or features). The classification consists of modeling (learning or inducting) a classification function (or classifier). This dissertation is not concerned with the feature extraction step, thus, henceforth, we will call the random variables $(\boldsymbol{X}, \boldsymbol{C})$ defined in the probability measure space $(\Omega, \mathcal{F}, Pr)$ the domain.

Definition 1.18 *Let $(\boldsymbol{X}, \boldsymbol{C})$ be an $(n + d)$-dimensional mixed random variable, with $\boldsymbol{X} = (X_1, ..., X_n)$ an n-dimensional predictor variable and $\boldsymbol{C} = (C_1, ..., C_d)$ a d-dimensional class variable. A function defined from $\boldsymbol{X}$ to $\boldsymbol{C}$*

$$\phi_{\boldsymbol{X};\boldsymbol{C}} : \boldsymbol{X} \rightarrow \boldsymbol{C} \tag{1.13}$$

is called classification function or classifier for $\boldsymbol{X}$ on $\boldsymbol{C}$.

The set of all the possible classifiers for $\boldsymbol{X}$ on $\boldsymbol{C}$ is denoted as $\boldsymbol{\varphi}_{\boldsymbol{X};\boldsymbol{C}}$. Henceforth, since the multidimensional predictor variable $\boldsymbol{X}$ and the multidimensional class variable $\boldsymbol{C}$ should be clear from the context, we denote as ϕ to a classifier for $\boldsymbol{X}$ on $\boldsymbol{C}$, for the sake of brevity. Similarly, the entire set of classifiers for $\boldsymbol{X}$ on $\boldsymbol{C}$ is referred to as $\boldsymbol{\varphi}$.

A classifier is usually modeled from a set of cases (instances or samples) $\mathcal{S} = \{(\boldsymbol{x}^{(1)}, \boldsymbol{c}^{(1)}), ..., (\boldsymbol{x}^{(N)}, \boldsymbol{c}^{(N)})\}$. From here on, the sets will be denoted in mathematical calligraphy. The data set used for learning a classifier is usually named training set.

Definition 1.19 *Let $(\boldsymbol{X}, \boldsymbol{C})$ be a multidimensional mixed random variable, a data set $\mathcal{S}$ of $(\boldsymbol{X}, \boldsymbol{C})$ is a collection of instances $(\boldsymbol{x}, \boldsymbol{c})$, $\mathcal{S} = \{(\boldsymbol{x}^{(1)}, \boldsymbol{c}^{(1)}), ..., (\boldsymbol{x}^{(N)}, \boldsymbol{c}^{(N)})\}$*

We are interested in data sets of $(\boldsymbol{X}, \boldsymbol{C})$ with *independent and identically distributed* (*iid*) samples $(\boldsymbol{x}^{(1)}, \boldsymbol{c}^{(1)}), ..., (\boldsymbol{x}^{(N)}, \boldsymbol{c}^{(N)})$. Samples that are drawn independently from the same distribution are said to be iid. Note that under iid assumption, the expectance of a function based on a mixed random variable $\boldsymbol{U} \in (\boldsymbol{X}, \boldsymbol{C})$ can be approximated as $E_{\boldsymbol{U}}[g(\boldsymbol{U})] \approx 1/N \sum_{i=1}^{N} g(\boldsymbol{u}^{(i)})$ [Bishop (2006)]. An iid data set of $(\boldsymbol{X}, \boldsymbol{C})$ can be generated by an independent random sampling of $(\boldsymbol{X}, \boldsymbol{C})$.

The classifiers are automatically learned from a training set by means of a *classifier induction algorithm* (learning algorithm or inductor).

Definition 1.20 *Let $(\boldsymbol{X}, \boldsymbol{C})$ be an $(n + d)$-dimensional random variable, where $\boldsymbol{X} = (X_1, ..., X_n)$ is the n-dimensional predictor variable and $\boldsymbol{C} = (C_1, ..., C_d)$ is the d-dimensional class variable. A function defined as*

$$A^{\mathcal{S}}_{\boldsymbol{X};\boldsymbol{C}} : \mathcal{S} \rightarrow \boldsymbol{\varphi}_{\boldsymbol{X};\boldsymbol{C}} \tag{1.14}$$

is called a classifier induction algorithm for $\boldsymbol{X}$ on $\boldsymbol{C}$, where $\mathcal{S}$ is a training set .

Since the multidimensional predictor variable $\boldsymbol{X}$ and the multidimensional class variable $\boldsymbol{C}$ should be clear from the context, we denote as A to a classifier induction algorithm for $\boldsymbol{X}$ on $\boldsymbol{C}$ for the sake of brevity.

The type of the classification problem is defined by the nature of the random variable $(\boldsymbol{X}, \boldsymbol{C})$, and the type of training set used to learn the classification model for $\boldsymbol{X}$ on $\boldsymbol{C}$. Classification problems with a multivariate class variable, $\boldsymbol{C} = (C_1, ..., C_d)$ with $d > 1$ are named *multidimensional classification* problems, and with $d = 1$ *unidimensional classification* problems. This document focuses on unidimensional classification problem.

If the class variable $\boldsymbol{C}$ is continuous, the problem is usually referred to as *regression*. From here on we call *supervised classification* problems to those with discrete class variable $\boldsymbol{C}$ having r different states (or classes), $\boldsymbol{c} \in \{\boldsymbol{c}^1, ..., \boldsymbol{c}^r\}$. When $\boldsymbol{C}$ is a mixed random variable, the problem is called *regression-classification* problem. If a class variable is not explicitly defined for the domain, i.e. $d = 0$, the problem is referred to as *unsupervised classification*.

If the class variable $\boldsymbol{C}$ has missing values in the training set $\mathcal{S}$, i.e. $\mathcal{S}$ is not complete in $\boldsymbol{C}$, the problem is usually named *semi-supervised classification*. On the other hand, if $\mathcal{S}$ is complete in $\boldsymbol{C}$, the problem is referred to as *supervised classification*. Note that if there are missing values for the predictors $\boldsymbol{X} = (X_1, ..., X_n)$, they are generally estimated using an imputation algorithm. The imputation is normally included as a step in the preprocess stage.

Definition 1.21 *Let $(\boldsymbol{X}, \boldsymbol{C})$, with $\boldsymbol{X} = (X_1, ..., X_n)$ and $\boldsymbol{C} = (C_1, ..., C_d)$, and a training set $\mathcal{S} = \{(\boldsymbol{x}^{(1)}, \boldsymbol{c}^{(1)}), ..., (\boldsymbol{x}^{(N)}, \boldsymbol{c}^{(N)})\}$ a classification problem is considered* unidimensional supervised classification *problem if it satisfies the following conditions:*

1. *$\boldsymbol{C}$ is a univariate, i.e. $d = 1$, discrete random variable, C, with r states, $c \in \{c^1, ..., c^r\}$.*
2. *$\mathcal{S}$ is complete in $\boldsymbol{C}$.*

From here on we consider only unidimensional supervised classification problems with a complete training set $\mathcal{S}$ in $\boldsymbol{X}$, and we simply call them supervised classification. Moreover, this dissertation is interested in supervised classification problems with $\boldsymbol{X}$ being a multidimensional mixed random variable.

1.5 Decision theory

This section introduces some concepts of decision theory [Duda et al. (2000); Bishop (2006)]. These concepts, combined with probability theory, allow us to make optimal decisions in situations involving uncertainty when $\rho(\boldsymbol{x}, c)$ is known.

As we noted before, the goal of supervised classification is to construct a classifier ϕ which predicts c given an unlabeled instance $\boldsymbol{x}$, i.e. with an unknown class label c. An intuitive approach involves two steps, the inference and the decision. Inference step consists of determining the generalized joint probability distribution for $(\boldsymbol{X}, C)$, $\rho(\boldsymbol{x}, c)$, which gives us the most complete probabilistic description of the domain. Although the determination of the joint distribution can be very useful and informative, in the end we must decide what class to assign to a new unlabeled instance. This is the decision step, and the subject of decision theory is to tell us how to make optimal decisions given the appropriate probabilities.

If our aim is to minimize the chance of assigning an unlabeled sample to the wrong class, then intuitively we would choose the class having the highest posterior probability. We now consider an intuitive measure of performance called classification error, $p(\phi(\boldsymbol{X}) \neq C)$, and we show that this intuition is correct for this score.

1.5.1 Classification error and decision boundaries

The classification error (or prediction error) ϵ of a classifier ϕ of $\boldsymbol{X} = (\boldsymbol{Y}, \boldsymbol{Z})$ on C, where $\boldsymbol{Y}$ and $\boldsymbol{Z}$ are multidimensional discrete and continuous random variables respectively, is defined as the probability of making a mistake using ϕ for classifying:

$$\begin{aligned}\epsilon(\phi) &= p(\phi(\boldsymbol{X}) \neq C) \\ &= E_{(\boldsymbol{X},C)}[1(\phi(\boldsymbol{X}); C)] \\ &= \sum_{\boldsymbol{y}} \int_{\mathbb{R}} \sum_{c \neq \phi(\boldsymbol{x})} \rho(\boldsymbol{y}, \boldsymbol{z}, c) d\boldsymbol{z} \\ &= \sum_{\boldsymbol{y}} \int_{\mathbb{R}} \rho(\boldsymbol{y}, \boldsymbol{z}) \sum_{c \neq \phi(\boldsymbol{x})} p(c|\boldsymbol{y}, \boldsymbol{z}) d\boldsymbol{z} \end{aligned} \tag{1.15}$$

where $\rho(\boldsymbol{x}, c)$ is the true underlying joint probability function of the domain, and $1(c, c') = 0$ when $c = c'$ and $1(c, c') = 1$ in the other case. This score is the most popular classification performance measurement. Henceforth, in this section we will consider $\boldsymbol{X} = \boldsymbol{Z}$ as a multidimensional continuous random variable for the sake of simplicity. The provided expressions can be generalized to a mixed random variable $\boldsymbol{X}$ by replacing the integrals associated to the discrete part $\boldsymbol{Y}$ by sums.

A classifier ϕ divides the input space $\Omega_{\boldsymbol{X}}$ into disjoint regions R_i called *decision regions*, one for each class, such that all points in R_i are assigned to class c^i

$$R_i = \{\boldsymbol{x} : \phi(\boldsymbol{x}) = c^i\} \tag{1.16}$$

The boundaries between decision regions are called *decision boundaries* [Bishop (2006)]. We can redefine the classification error by means of decision regions as follows:

$$\begin{aligned}\epsilon(\phi) &= \sum_{c^i} \epsilon_{R_i}(\phi) \\ &= \sum_{c^i} \int_{R_i} \sum_{c^j \neq c^i} \rho(\boldsymbol{x}, c^j) d\boldsymbol{x} \\ &= \sum_{c^i} \int_{R_i} f(\boldsymbol{x}) \sum_{c^j \neq c^i} p(c^j|\boldsymbol{x}) d\boldsymbol{x} \end{aligned} \tag{1.17}$$

where $\epsilon_{R_i}(\phi) = \int_{R_i} \sum_{c^j \neq c^i} \rho(\boldsymbol{x}, c^j) d\boldsymbol{x}$ is the error associated to the decision region R_i defined by the classifier ϕ. From Equation 1.17, it is easy to demonstrate that the minimum classification error is obtained if each value of $\boldsymbol{x}$ is assigned to the class for which the posterior probability $p(c|\boldsymbol{x})$ is largest, i.e. *Bayes classifier*. This is known as *winner-takes-all* rule (or *Bayes rule*). These concepts are illustrated in Figure 1.1 for two classes, and a single predictor variable.

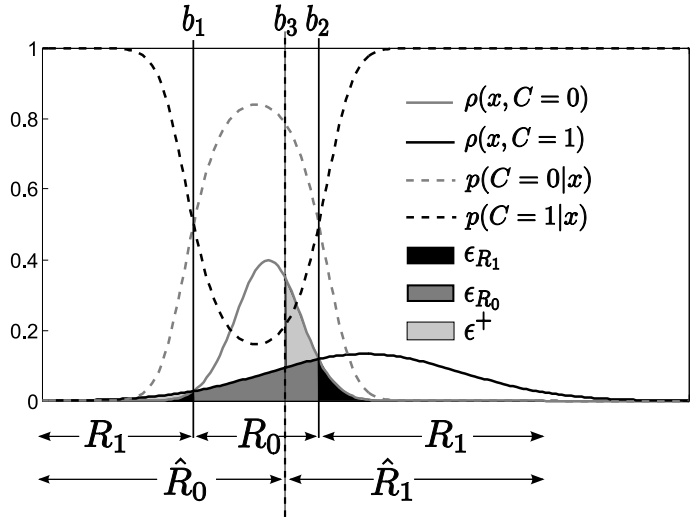


Fig. 1.1. This figure illustrates the equivalence of the decision rules $\arg_c \max p(c|x)$ and $\arg_c \max \rho(x, c)$. The vertical black lines, denoted as b_1 and b_2, are the decision boundaries determined by the Bayes classifier, ϕ_B. The decision boundaries delimit the optimal decision regions R_0 and R_1 associated to the class $C = 0$ and $C = 1$, respectively. Following Definition 1.17, the error ϵ can be decomposed by regions as $\epsilon_B = \epsilon_{R_0} + \epsilon_{R_1}$, where $\epsilon_{R_0} = \int_{R_0} \rho(x, C = 1)dx$ and $\epsilon_{R_1} = \int_{R_1} \rho(x, C = 0)dx$. The error components ϵ_{R_0} and ϵ_{R_1} are the surfaces marked in grey and black, respectively. Otherwise, a suboptimal classifier ϕ determines the suboptimal decision boundary, b_3, represented by a dashed vertical line. The boundary b_3 delimitates two suboptimal regions denoted as $\hat{R}_0$ and $\hat{R}_1$, which are associated to the class $C = 0$ and $C = 1$, respectively. The suboptimal classifier ϕ increases the error with respect to ϕ_B in $\epsilon^+ = \epsilon(\phi) - \epsilon(\phi_B) = \int_{\hat{R}_1 \cap R_0} \rho(x, C = 0) - \rho(x, C = 1)dx + \int_{\hat{R}_0 \cap R_1} \rho(x, C = 1) - \rho(x, C = 0)dx$.

1.5.2 Expected loss

For many real world domains the minimization of the prediction error is not convenient because the consequences of different types of mistakes can be dramatically different. For example, let us consider a cancer diagnosis problem. If a patient who does not have cancer is incorrectly diagnosed as having cancer, the consequences may cause some stress for the patient. Conversely, if a patient with cancer is diagnosed as healthy, the result may be premature death due to the lack of treatment.

The differences in the error cost are formalized by means of a loss function L:

$$\begin{aligned} L : (c, \phi(\boldsymbol{x})) &\rightarrow \mathbb{R} \\ (c^i, c^j) &\rightarrow L_{i,j} \end{aligned}$$

Suppose that, for a new unlabeled sample $\boldsymbol{x}$, the true class is c^i and the predicted class is c^j. This prediction incurs in some level of loss that we denote by $L_{i,j}$, which can be viewed as the (i,j) element of a *loss matrix*. Note that $L_{i,i} = 0$. The optimal prediction can be reformulated as the one which minimizes the *expected loss*. The expected loss is given by the expectance of the *loss function* L:

$$\epsilon_L(\phi) = \sum_{\forall c^i} \sum_{\forall c^j} \int_{R_j} L_{i,j} \rho(\boldsymbol{x}, c^i) d\boldsymbol{x} \tag{1.18}$$

Note that the classification error is the particular case of expected loss when $L_{i,j}$ is constant for all $i \neq j$ (*0-1 loss function*).

The goal is to choose the regions R_j in order to minimize the expected loss. The minimization of the expected loss implies that for each $\boldsymbol{x}$ the selected class $\phi(\boldsymbol{x}) = c^i$ should minimize $\sum_{c^j \neq c^i} L_{i,j} \rho(\boldsymbol{x}, c^j)$, which is equivalent to minimize $\sum_{c^j \neq c^i} L_{i,j} p(c^j|\boldsymbol{x})$. This leads to the *weighted winner-takes-all* rule:

$$c^B = arg_{c^i} min \sum_{c^j \neq c^i} L_{i,j} p(c^j|\boldsymbol{x}) \tag{1.19}$$

The classifier that uses the true underlying joint probability $\rho(\boldsymbol{x}, c)$ together with the weighted winner-takes-all rule for classifying new unlabeled instances is the *weighted Bayes classifier*, $\phi_B(\boldsymbol{x}) = c^B$. Given a loss function L, the associated Bayes classifier achieves the smallest possible expected loss. Note that under 0-1 loss functions, the weighted winner-takes-all rule is equivalent to the winner-takes-all rule. From here on, since this dissertation is not concerned with decision theory we will consider only 0-1 loss functions.

1.6 Generative, conditional and discriminative learning

In Section 1.5 we suggest an intuitive approach for learning a classifier which consisted of two stages: inference and decision. Depending on how the steps of the learning process are performed, three approaches could be distinguished: generative, conditional and discriminative learning [Jebara (2004)]. Next, we briefly introduce the three approaches highlighting their relative merits.

Generative learning solves the problem of estimating the underlying generalized joint probability distribution, $\rho_\phi(\boldsymbol{x}, c)$. Then marginalizing and conditioning we can obtain:

$$\begin{aligned} p_\phi(c|\boldsymbol{x}) &= \frac{\rho_\phi(\boldsymbol{x}, c)}{\sum_{c'} \rho_\phi(\boldsymbol{x}, c')} \\ &= \frac{\rho_\phi(\boldsymbol{x}, c)}{\rho_\phi(\boldsymbol{x})} \end{aligned} \tag{1.20}$$

Taking into account that $\rho_\phi(\boldsymbol{x})$ is constant for all c we can obtain $p_\phi(c|\boldsymbol{x})$ normalizing the joint distribution for each $\boldsymbol{x}$, $p_\phi(c|\boldsymbol{x}) \propto \rho_\phi(\boldsymbol{x}, c)$. Finally,

based on the conditional probability distribution obtained $p_\phi(c|\boldsymbol{x})$ and inserting it into weighted winner-takes-all rule (Equation 1.19), the class label c is determined for each $\boldsymbol{x}$.

$$\phi(\boldsymbol{x}) = arg_{c^i} min \sum_{c^j \neq c^i} L_{i,j} p_\phi(c^j|\boldsymbol{x}) \tag{1.21}$$

Note that Figure 1.1 summarizes all this process for one predictor, from the joint generalized probability distribution $\rho(c, x)$ to decision regions, R_0 and R_1. Approaches that explicitly or implicitly model the joint generalized joint probability distribution, $\rho(\boldsymbol{X}, C)$, are known as generative models. Examples of generative learning are the classifiers based on Bayesian networks (see Chapters 2, 3 and 4) or Parzen window classifier (see Chapter 4).

Conditional learning solves the problem of estimating the conditional joint probability distribution $p_\phi(c|\boldsymbol{x}) \approx p(c|\boldsymbol{x})$ and, then, it is plugged-in weighted winner-takes-all-rule (Equation 1.19) for inducting the classifier. Classifier induction algorithms that model the posterior probabilities directly are called conditional learning. Many authors consider this approach discriminative [Pernkopf and Bilmes (2004); Santafé et al. (2005)]. For example, some Bayesian networks based classifiers (see Chapter 5) and logistic regression can be considered conditional approaches.

Finally, *discriminative learning* performs the inference and decision steps at the same time. It finds a classification function $\phi(\boldsymbol{x})$ which maps each input $\boldsymbol{x}$ onto a class label c, i.e. discriminative learning directly determines the decision boundaries. In this case, probabilities play no role. Linear discriminant analysis and support vector machines are examples of discriminative classifiers.

Generative approach is the most demanding (in terms of number of parameters) because it involves the estimation of the joint distribution. For many applications, $\boldsymbol{x}$ will have high dimensionality, and consequently we may need a large training set in order to be able to determine the class conditional densities to reasonable accuracy (see Section 1.11). On the other hand, some advantages can be considered with respect to conditional and discriminative learning. It is possible to generate synthetic data representative of the domain by sampling a generative model because it is based on an approach of the joint generalized probability distribution, $\rho_\phi(\boldsymbol{x}, c) \approx \rho(\boldsymbol{x}, c)$. Besides, generative approach allows the marginal density $\rho(\boldsymbol{x})$ to be approached by marginalization of $\rho_\phi(c, \boldsymbol{x})$ over C. This can be useful for outlier detection, that is new unlabeled cases that have low probability $\rho_\phi(\boldsymbol{x})$, and for which the predictions may be of low accuracy. However, if we only wish to make classification decisions, we only really need the posteriori probabilities $p_\phi(c|\boldsymbol{x})$ which can be directly obtained using the conditional learning.

The simplest approach (in terms of the number of parameters) is the discriminative learning in which we use the training data to find a discriminant function, i.e. a set of decision boundaries. However, the optimization of the parameters in discriminative approaches tends to be harder (computationally)

than in the generative approach. Besides, discriminative approach does not have available the posterior probabilities, $p_\phi(c|\boldsymbol{x})$, and this knowledge could be useful in the following situations:

- When the elements of the loss matrix are evolving from time to time. If we have only a discriminant function, then any change to the loss matrix would require to relearn the classification problem given by the new loss matrix (see Section 1.9.2).
- Posterior probabilities allows us to determine a rejection criterion τ for which an unlabeled sample $\boldsymbol{x}$ is not classified if $\nexists c : p_\phi(c|\boldsymbol{x}) > \tau$.
- When the classes are unbalanced. On one hand, if the posterior probability is unknown, we could generate a balanced training set from the original training set and, then, learn the model using the discriminative learning. On the other hand, if the posterior probability distribution is known we can simply multiply the modeled a posteriori probability $p_\phi(c|\boldsymbol{x})$ by the balanced a priori $p^b(c)$ and then divide by the unbalanced one $p^u(c)$. Finally we need to normalize a posteriori distribution $p_\phi(c|\boldsymbol{x})$ to ensure that the new posterior probabilities sum to one, $p^b_\phi(c|\boldsymbol{x}) \propto p^b(c)/p^u(c)p_\phi(c|\boldsymbol{x})$.
- For combining different models. In complex classification applications, often the problem is divided into smaller classification subproblems. For example, given a classification problem in a domain described by $p(\boldsymbol{x}, c)$ the problem can be divided into subproblems in the sub-domains $\{\boldsymbol{U}, C\}$, $\{\boldsymbol{V}, C\}$ and $\{C\}$, where $\boldsymbol{U} \cup \boldsymbol{V} = \boldsymbol{X}$ and $\boldsymbol{U} \cap \boldsymbol{V} = \emptyset$. Three probabilistic models can be created for each sub-domain, $\phi_{\boldsymbol{U};C}$, $\phi_{\boldsymbol{V};C}$ and $\phi_{;C}$. As long as each of the models gives posterior probabilities, $p_{\phi_{\boldsymbol{U};C}}(c|\boldsymbol{u})$, $p_{\phi_{\boldsymbol{V};C}}(c|\boldsymbol{v})$ and $p_{\phi_{;C}}(c)$, we can combine the outputs systematically using the rules of probability. One simple way is to assume class conditional independence between $\boldsymbol{U}$ and $\boldsymbol{V}$ given the class C and combine the conditional probabilities as $p_{\phi_{\boldsymbol{X};C}}(c|\boldsymbol{u}, \boldsymbol{v}) = p_{\phi_{\boldsymbol{U};C}}(c|\boldsymbol{u})p_{\phi_{\boldsymbol{V};C}}(c|\boldsymbol{v})/p_{\phi_{;C}}(c) \simeq p(c|\boldsymbol{u}, \boldsymbol{v})$.

For further details in generative, conditional and discriminative learning the reader may consult [Jebara (2004); Bishop (2006)].

1.7 Estimation theory

This section tries to briefly introduce some concepts of estimation theory which are useful for the understanding of this document. We have decided to introduce these concepts for unidimensional parameters for the sake of simplicity. But, it should be noted that the parameter in its most general form is a multidimensional value.

Estimation theory tries to guess a magnitude (or parameter) θ from a set of nosy observations (samples or instances), $\mathcal{S} = \{\boldsymbol{x}^{(1)}, ..., \boldsymbol{x}^{(N)}\}$, where $\boldsymbol{X}$ is a multidimensional mixed random variable for the sake of generality. We do not have direct access to θ parameter but it is assumed that it has some functional relationship with the collected observations. All the estimators provided in this

dissertation assume that the observations, $\boldsymbol{x}^{(i)}$, are *independent and identically distributed* (iid).

In the general sense, a *parameter* θ is a function of the generalized joint probability distribution for $\boldsymbol{X}$, $\rho(\boldsymbol{x})$. A statistic $\hat{\Theta}$ that is used to estimate θ is called an estimator of θ. The estimator $\hat{\Theta}$ is an observable function of the data $\mathcal{S}$, $\hat{\theta} = \hat{\Theta}(\mathcal{S})$, where $\hat{\theta}$ is the estimation (or estimated parameter) and $\mathcal{S}$ depends on the true θ. It should be noted that the data $\mathcal{S}$ can be considered a random variable, whose generalized probability distribution is given by $\rho(\mathcal{S}) = \prod_{i=1}^{N} \rho(\boldsymbol{x}^{(i)})$. Thus, assuming that $\mathcal{S}$ is observable, $\hat{\Theta}(\mathcal{S})$ can be considered a random variable derived from the data variable $\mathcal{S}$. Let us assume that $\hat{\Theta}$ is a continuous random variable for the sake of simplicity, then its probability density function is defined as $f(\hat{\Theta} = \hat{\theta}) = f_{\hat{\Theta}}(\hat{\theta}) = \sum_{\mathcal{S}|\hat{\Theta}(\mathcal{S})=\hat{\theta}} \rho(\mathcal{S})$.

The quality of the estimator $\hat{\Theta}$ can be given in probabilistic terms because $\hat{\Theta}$ is a random variable. The deviation (or error) of an estimation $\hat{\Theta}(\mathcal{S}) = \hat{\theta}$ is given by $\Delta_{\hat{\Theta}} = \theta - \hat{\theta}$ and, hence, $\Delta_{\hat{\Theta}}$ is also a random variable. The average of $\Delta_{\hat{\Theta}}$ is known as the *bias* of the estimator $\hat{\Theta}$, $\delta_{\hat{\Theta}} = E_{\mathcal{S}}[\Delta_{\hat{\Theta}}]$. Thus, the mean value of an estimator can be decomposed as $E[\hat{\Theta}] = E_{\mathcal{S}}[\theta - \Delta_{\hat{\Theta}}] = \theta - \delta_{\hat{\Theta}}$. The bias can be understood as the average maladjustment or fitness error of the estimator. It can be positive, negative or zero. If, on average, an estimator has bias equal to zero, it is said to be an *unbiased* estimator. On the other hand, the *variance* of an estimator is defined as $\sigma^2_{\hat{\Theta}} = E[\hat{\Theta}^2] - E[\hat{\Theta}]^2$. The variance of an estimator can be understood as the average sensitivity to changes in the observations set, $\mathcal{S}$, used to perform the estimation. The variance can be used as a measure of instability of the estimations. In general, the variance of an estimator $\sigma^2_{\hat{\Theta}}$ cannot be considered completely independent from θ, because the values $\hat{\Theta} = \theta$ are given by $\mathcal{S}$, $\mathcal{S}$ is obtained by randomly sampling $\rho(\boldsymbol{x})$, and θ is deterministically given by $\rho(\boldsymbol{x})$. An estimator is called *efficient* if it has minimum variance.

As we will note in Section 1.8.2, in order to perform powerful comparisons between classifier induction algorithms, the variance of the performance estimator used in the experimentation is crucial. Figure 1.2 shows an example of the distribution of two estimators $\hat{\Theta}_1$ and $\hat{\Theta}_2$.

Intuitively, a desirable property for an estimator is to have good fitness and low sensitivity to changes in $\mathcal{S}$ and, so, we are interested in low biased estimators with low variance. The estimators with low bias and variance obtain precise and stable estimations of the required parameter. As we will see in Section 1.8.2, the bias and the variance are used in order to evaluate the quality of the error estimators. The following heuristic for choosing an appropriate estimator can be used

- Given two estimators equally biased, choose the estimator with lower variance.
- Given two estimators with equal variance, choose the estimator with lower bias.

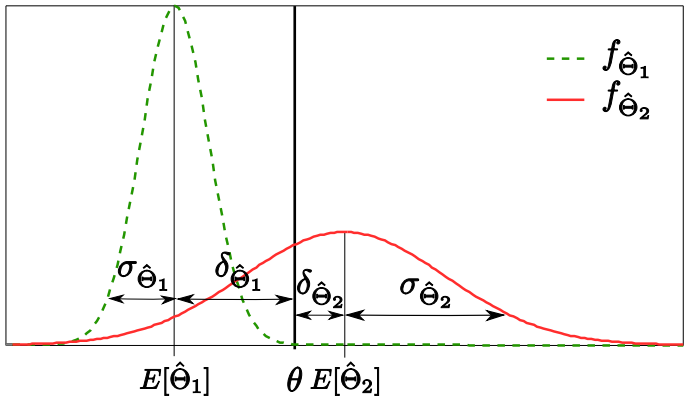


Fig. 1.2. This figure shows the distributions of two estimators of the parameter θ, $\hat{\Theta}_1$ and $\hat{\Theta}_2$, indicating their respective biases $\delta_{\hat{\Theta}_1}$ and $\delta_{\hat{\Theta}_2}$, and deviations $\sigma_{\hat{\Theta}_1}$ and $\sigma_{\hat{\Theta}_2}$, where the standard deviation is defined as the square root of the variance. The figure shows that the estimator $\hat{\Theta}_1$ has higher bias and lower variance than $\hat{\Theta}_2$.

These heuristics can be understood from the mean squared error point of view, which is given by $MSE(\hat{\Theta}^2) = E[\Delta^2]$. The error can be decomposed into bias and variance terms as follows

$$MSE(\hat{\Theta}^2) = \delta^2_{\hat{\Theta}} + \sigma^2_{\hat{\Theta}} \tag{1.22}$$

Thus the mean squared error depends on both bias and variance. Given two estimators equally biased, the difference of their mean square errors is the difference between the variances. On the other hand, given two estimators with equal variance, the difference between their mean square error is the difference between their squared biases.

Usually, estimators are based on a set of assumptions. For example, as we noted before, all of the estimators presented in this document assume that the samples provided for estimating, $\mathcal{S}$, are independent and identically distributed. The (useful) assumptions made by an estimator allow to more efficiently extract information from data, provided they are true. For example, if an estimator assumes that the underlying distribution of the data is normally distributed and it is true, the estimator could reduce its variance with respect to a similar estimator without normality assumption. On the other hand, when the data is far from the assumptions, the estimator will extract incorrect information from the data, and the estimator could increase its bias. Normally, the assumptions are related with the nature of the underlying distribution of the data. An estimator which assumes that the data comes from a parametric distribution family is named as parametric, and, on the other hand if the estimator does not assume any particular parametric family of distributions, it is called non-parametric.

For further details on estimation theory the reader may consult [Dudewicz and Mishra (1988)].

1.8 Error estimation, comparison and analysis

This section includes some estimators of the classification (or prediction) error (see Equation 1.15), some statistical tools for the comparison of classifiers based on the estimated errors and, finally, a tool for analyzing the sources of the error.

1.8.1 Classification error estimation

We are interested in estimating the generalization capability of a classifier or a classifier induction algorithm. The generalization capability could be intuitively defined as the aptitude that allows to correctly classify instances that have been not used for training the model. It could be thought to be the opposite to overfitting, the skill of learning by heart the instances used for training the model. Generalization capability is a desirable property for any classifier. The generalization capability can be measured in terms of performance scores, such us those described in Section 1.9.

However, this section focuses on classification (or prediction) error (see Equation 1.15), because it is one of the most simple and intuitive performance measures. Through this subsection, we introduce some pointwise classification error estimators. It should be noted that the same estimators can be used for estimating other quantities that depend on a training-test procedure using data samples. For example, they can be used to estimate different measures of the performance of a classifier or to estimate the likelihood of a discretization policy, where both the classifier and the discretization policy are learned from a training set.

In this section, depending on the purpose of the experimentation, we consider two interrelated error estimation problems: *classifier evaluation problem* and *algorithm evaluation problem*. Both problems are introduced in terms of discrete predictors, $\boldsymbol{X}$, for the sake of simplicity.

In the classifier evaluation problem, we want to estimate the prediction error of a particular classifier, ϕ, which has been learned from an iid data set, $\mathcal{S}_N$, obtained by a random sampling of $p(\boldsymbol{x}, c)$, using a classifier induction algorithm, $A(\mathcal{S}_N) = \phi$. The error of a particular classifier trained in $\mathcal{S}_N$ using the algorithm A is denoted as $\epsilon_{\mathcal{S}_N}(A)$, or simply as $\epsilon_{\mathcal{S}_N}$ when the algorithm is clear from the context, and it is given by.

$$\epsilon_{\mathcal{S}_N}(A) = \epsilon(\phi = A(\mathcal{S}_N)) \tag{1.23}$$

It can be considered an application oriented evaluation because in applications we are interested in finding the best algorithm to solve a task at hand,

specified by a particular training set $\mathcal{S}_N$ and some information about the data generating process. It can be thought of as an application based evaluation, because the goal of an application usually consists of selecting the best classifier learned from the data set $\mathcal{S}_N$. Sometimes, prediction error is also called *conditional error* [Braga-Neto (2005)] because the classifier is trained in a particular data set of size N, $\mathcal{S}_N$.

On the other hand, in algorithm evaluation problem, we are not really interested in the performance of a specific classifier induced from a particular training set, $\mathcal{S}_N$. Thus, we are concerned with the sensitivity of the learning algorithm to the choice of the training set. So, we will take into account a set of training sets rather than a single one $\mathcal{S}_N$: the set of all training sets of size N. Since the prediction error, $\epsilon(A(\mathcal{S}_N))$, is a function of the random variable $\mathcal{S}_N$, it can be thought of as a random variable. The prediction error $\epsilon(A(\mathcal{S}_N))$ is distributed according to the probability distribution $p(\epsilon(A(\mathcal{S})) = e) = \sum_{\mathcal{S}_N|\epsilon(A(\mathcal{S}_N))=e} p(\mathcal{S}_N)$, where $p(\mathcal{S}_N) = \prod_{i=1}^{N} p(\boldsymbol{x}^{(i)}, c^{(i)})$. Then, we can define the *expected prediction error* as the expectation of the error of a classifier trained with sample sets of size N, using the inductor A, and it will be denoted as $\epsilon_N(A)$ or simply as ϵ_N when the inductor algorithm is clear from the context:

$$\begin{aligned} \epsilon_N(A) &= E_{\mathcal{S}_N}[\epsilon_{\mathcal{S}_N}(A)] \\ &= \sum_{\mathcal{S}_N} p(\mathcal{S}_N)\epsilon(A(\mathcal{S}_N)) \end{aligned} \tag{1.24}$$

The subscript N is used to differentiate the expected classification error, ϵ_N, from classification error, $\epsilon_{\mathcal{S}_N}$. Expected prediction error is sometimes called *unconditional error* since it does not depend on a particular training set because it considers all $\mathcal{S}_N$ [Braga-Neto (2005)]. It should be noted that both problems are closely related and $\epsilon_{\mathcal{S}_N}$ can be used as an estimate of ϵ_N.

Note that in both problems the number of cases is treated explicitly because the size of the training set has a high impact on the prediction error. It is well known that both errors $\epsilon_{\mathcal{S}_N}$ and ϵ_N tend to decrease as the size of the training set increases. For instance, the consistent classifiers, such us Parzen window or k-nearest neighbor classifiers, converge (under mild conditions) to Bayes classifier as the number of cases goes to infinity.

When the data distribution $p(\boldsymbol{x}, c)$ is unknown, both $\epsilon_{\mathcal{S}_N}$ and ϵ_N cannot be computed. They have to be estimated from the observed data, $\mathcal{S}_N$. Both problems consist of estimating a single unknown value using a single training set, $\mathcal{S}_N$ (see Section 1.7). But, often, it is crucial to assess the uncertainty attached to this estimation, which can be also measured in terms of variance. In application-oriented experiments, the variance can be used to give a confidence interval on the error associated to the model learned from $\mathcal{S}_N$, $\epsilon_{\mathcal{S}_N}$. On the other hand, in algorithm-oriented experiments, it is important to give a measure related to the degree of (in)stability of the induction algorithm, A, which can be also measured in terms of variance. The degree of (in)stability

is essential to perform comparisons between both classifiers and algorithms, as we will see in Section 1.8.2.

The estimators of the error presented will be based on the following estimator of the error

$$\hat{\epsilon}(A(S) = \phi; \mathcal{T}) = \frac{1}{|\mathcal{T}|} \sum_{(\boldsymbol{x},c)\in\mathcal{T}} 1(\phi(\boldsymbol{x}), c) \tag{1.25}$$

where A is the classifier induction algorithm, $\mathcal{S}$ is the training set, ϕ is the classifier inducted from $\mathcal{S}$ using A, $\mathcal{T}$ is the test set, and $1(i, j) = 1$ iff $i \neq j$ and zero otherwise (0-1 loss function). Usually, the error estimators consist of a procedure for generating a set of pairs training-test data $\{(\mathcal{S}^{(1)}, \mathcal{T}^{(1)}), ..., (\mathcal{S}^{(k)}, \mathcal{T}^{(k)})\}$. A classifier is learned from training set $\mathcal{S}^{(i)}$ and, then, the error of the learned classifier in the generated test set is computed $\hat{\epsilon}^{(i)} = \hat{\epsilon}(A(\mathcal{S}^{(i)}); \mathcal{T}^{(i)})$. The estimated error $\hat{\epsilon}$ is usually given by an appropriate statistic over the computed *intermediate errors* $\{\hat{\epsilon}^{(1)}, ..., \hat{\epsilon}^{(k)}\}$ taking into account the procedure used for generating the pairs training-test. For example, as we will see later, most of the error estimators perform an average of the intermediate errors.

The section is divided in four parts. Section 1.8.1.1 shows some useful intuitions about the bias and variance of the error estimators provided through this section. Then, Section 1.8.1.2 and Section 1.8.1.3 present a set of estimators of the error taking into account their appropriateness to big and medium-small sample sizes, respectively. We consider that a small sample size has less than 100 instances and a big sample size more than 5000. When the number of instances is between 100 and 5000 we say that the sample size is medium.

1.8.1.1 Bias and variance

We consider in this subsection some intuitions behind the bias and variance of the error estimators which will be presented in the following two subsections. These concepts will allow to intuitively understand the theoretical properties of the estimators introduced.

An error estimator is said to be *optimistic* when it has a positive bias, i.e. on average it estimates an error lower than the true error, and *pessimistic* when it has negative bias. Figure 1.2 shows an example with two estimators $\hat{\Theta}_1$ and $\hat{\Theta}_2$. Let us consider $\Theta = \epsilon$, then $\hat{\Theta}_1$ and $\hat{\Theta}_2$ are optimistic and pessimistic error estimators respectively. Usually, the error estimators can be optimistic due to an overlap between each test set $\mathcal{T}^{(i)}$ and its corresponding training set $\mathcal{S}^{(i)}$, and to the overfitting of the classifier induction algorithm. For example, a complete optimistic estimator for nearest neighbor classifier [Devroye et al. (1996)] is the resubstitution estimator (see Paragraph A. in Section 1.8.1.2). When the test set $\mathcal{T}^{(i)}$ shares some instances with its corresponding training set, $\mathcal{S}^{(i)}$, we say that there exists a *training-test dependence*. On the other

hand, they can be pessimistic because the generated training set sizes $\mathcal{S}^{(i)}$ are smaller than the original data set $\mathcal{S}_N$ and they could contain less information than $\mathcal{S}_N$. These intuitions will be useful to understand the nature of the estimators given in Section 1.8.1.3.

Now we will focus our attention on the variance over the computed errors. A low variance is crucial for making stable comparisons between classifiers (see Section 1.8.2), and it is a desirable property for an estimator in general. Besides, it is advisable to provide the variance as an indicator of the uncertainty around the obtained estimation. The variance can be overestimated or underestimated. Usually, the computed variance can be underestimated due to the dependence between the computed errors, which are supposed to be i.d.d. samples. The dependence between the computed errors comes from the overlapping among the different pairs training-test $(\mathcal{S}^{(i)}, \mathcal{T}^{(i)})$ and $(\mathcal{S}^{(j)}, \mathcal{T}^{(j)})$, $\mathcal{S}^{(i)} \cap \mathcal{S}^{(j)} \neq \emptyset$ and $\mathcal{T}^{(i)} \cap \mathcal{T}^{(j)} \neq \emptyset$. If the cases are shared by the generated training sets, $\mathcal{S}^{(i)} \cup \mathcal{S}^{(j)} \neq \emptyset$ (training-training dependence), the obtained classifiers could be dependent and, thus, the computed errors could be also dependent. Otherwise, if the cases are shared by the generated test sets, $\mathcal{T}^{(i)} \cup \mathcal{T}^{(j)} \neq \emptyset$ (test-test dependence), the obtained errors $\{\hat{\epsilon}^{(1)}, ..., \hat{\epsilon}^{(k)}\}$ could be dependent because each intermediate error $\hat{\epsilon}^{(i)}$ consists of an average over the instances contained in its corresponding test set $\mathcal{T}^{(i)}$ (see Definition 1.25).

1.8.1.2 With big sample size

This subsection presents two simple estimators based on a single pair training-test, $(\mathcal{S}, \mathcal{T})$. Both estimators are appropriate for large training sets due to their simplicity and low computation requirements. On the contrary, the variance of both estimators in small to medium data sets tends to be prohibitive because the estimators are constructed by training a single model.

If the amount of data is large enough, the variance of an estimator based on a single pair training-test, $\hat{\epsilon}$, can be estimated as $\hat{\sigma}^2_{\hat{\epsilon}} = 1/|\mathcal{T}| \sum_{(\boldsymbol{x},c)\in\mathcal{T}} (1(\phi(\boldsymbol{x}); c) - \hat{\epsilon})^2$, where $\phi = A(\mathcal{S})$ and $|\mathcal{T}|$ is the size of the test set, $\mathcal{T}$. This variance is useful to quantify the uncertainty surrounding an estimation of the prediction error $\epsilon_{\mathcal{S}}$. But estimators based on a single pair training-test do not account for the variance with respect to the training set and, thus, they may be considered inappropriate to quantify the uncertainty surrounding an estimation of the error, i.e. it tends to underestimate the true variance. Therefore, they may be considered not suitable for the purpose of the expected prediction error estimation, ϵ_N, or for comparing induction algorithms [Dietterich (1999)].

A. Resubstitution

Resubstitution [Devroye and Wagner (1979)] consists of training a classifier using the available data $\mathcal{S}_N$ and computing the error in the test set $\mathcal{S}_N$ using Equation 1.25. The estimation process is summarized as follows

$$\mathcal{S}_N \rightarrow (\mathcal{S}_R, \mathcal{T}_R) \rightarrow \hat{\epsilon}_R = \hat{\epsilon}(A(\mathcal{S}_R), \mathcal{T}_R) \tag{1.26}$$

where $\mathcal{S}_R = \mathcal{T}_R = \mathcal{S}_N$, $\hat{\epsilon}(\cdot)$ is the simple estimator defined in Equation 1.25, and $\hat{\epsilon}_R$ is the resubstitution error estimation.

This estimator tends to be optimistic due to its strong training-test dependency because $\mathcal{S}_R = \mathcal{T}_R$. Clearly, the estimator tends to be more optimistic as the overfitting of a classifier increases. For example, the extreme case of nearest neighbor induction algorithm, A_{NN}, trained in $\mathcal{S}_N$ and tested in the same data obtains zero error, $\hat{\epsilon}(A_{NN}(\mathcal{S}_N), \mathcal{S}_N) = 0$. Besides, resubstitution is an estimator with a high variance because the estimated error depends on a single value determined from the particular data set available $\mathcal{S}_N$.

B. Holdout

Holdout estimator [Larson (1931); Horst (1941)] tries to avoid the overfitting problem of resubstitution estimator following a similar procedure, that is, based on a single training-test pair. It splits the available data into a disjoint pair training-test, $\mathcal{S}_N \rightarrow (\mathcal{S}_H, \mathcal{T}_H)$ where $\mathcal{S}_H \cup \mathcal{T}_H = \mathcal{S}_N$ and $\mathcal{S}_H \cap \mathcal{T}_H = \emptyset$. Typically, training set contains $|\mathcal{S}_H| = 2/3N$ instances while the size of the test set is $|\mathcal{T}_H| = 1/3N$. The estimation process is equivalent to the process described in Equation 1.26 with different training-test pair, $(\mathcal{S}_H, \mathcal{T}_H)$. The holdout estimator of the error is given by

$$\hat{\epsilon}_H = \hat{\epsilon}(A(\mathcal{S}_H), \mathcal{T}_H) \tag{1.27}$$

where $\hat{\epsilon}(\cdot)$ is defined in Equation 1.25.

This estimator avoids the complete training-test dependency of resubstitution estimator by splitting the original data into disjoint training and test sets $\mathcal{S}_H$ and $\mathcal{T}_H$. But, it is known that the error of a classifier tends to decrease as the training set size increases, because more information could be obtained from the data. Thus, the holdout estimator tends to be pessimistic because the size of the training set generated, $\mathcal{S}_H$, is smaller than N (typically $2/3N$). On the other hand, holdout tends to have more variance than resubstitution because the learned classifier is tested in only $|\mathcal{T}_H| = 1/3N$ cases, while resubstitution tests the learned classifier in the entire available data $\mathcal{S}_N$.

1.8.1.3 With small and medium sample sizes

This section introduces the most popular estimators for the machine learning community. They are based on a set of pairs training-test $\{(\mathcal{S}^{(1)}, \mathcal{T}^{(1)}), ..., (\mathcal{S}^{(k)}, \mathcal{T}^{(k)})\}$ rather than in a single pair $(\mathcal{S}, \mathcal{T})$ and in the assumption of stability of the classifiers obtained using the inductor A. The training-test pairs are usually generated by sampling the available data, $\mathcal{S}_N$.

The estimation process usually consists of an iterative process of train and test classifiers for each generated pair training-test, $(\mathcal{S}^{(i)}, \mathcal{T}^{(i)})$, and then,

averaging the obtained partial errors to compute the final estimator. Thus, instead of using a single classifier learned from a single training test, these estimators are based on $\mathcal{O}(k)^*$ different classifiers, $\{A(\mathcal{S}^{(1)}), ..., A(\mathcal{S}^{(k)})\}$, which are related and dependent, i.e. they are learned using the same algorithm A and the training sets $\{\mathcal{S}^{(1)}, ..., \mathcal{S}^{(k)}\}$ usually shares some of their instances, $\mathcal{S}^{(i)} \cap \mathcal{S}^{(j)} \neq \emptyset$. The use of different models $\{A(\mathcal{S}^{(1)}), ..., A(\mathcal{S}^{(k)})\}$ closely related to the classifier that we are interested in, $\phi = A(\mathcal{S}_N)$, could allow to better study its generalization capability. Summarizing, the estimators presented in this subsection could exploit the information implicit in the available data $\mathcal{S}_N$ in a more efficient way than resubstitution and holdout estimators presented in the previous subsection.

The estimators based on repetitions assume that the classifier induction algorithms are stable in the data set $\mathcal{S}_N$. Moreover, they suppose that the process of generation of pairs training-test should closely approximate the real world.

The main difference between the provided estimators consists of the process of generating the training-test pairs and the computation of the error. All of the estimators of this subsection are appropriate for small and medium size data sets and it is advisable to use them in large data sets when the computational requirements are not prohibitive. Besides, we recommend the use of repeated k-fold cross validation, bootstrap and bolstered for small size data sets.

On the other hand, there is a test-test dependence due to $\mathcal{T}^{(i)} \cap \mathcal{T}^{(j)} \neq \emptyset$ for all $i \neq j$. Thus, the estimated intermediate errors $\{\hat{\epsilon}^{(1)}, ..., \hat{\epsilon}^{(k)}\}$ could be dependent and, thus, the variance associated to the estimated error could be underestimated.

A. Repeated holdout

Repeated holdout estimation is based on iterating a random holdout process k times with different training-test splits $(\mathcal{S}^{(i)}, \mathcal{T}^{(i)})$ of constant size. Then, it computes the estimated error as the average of the obtained intermediate errors. The process of repeated holdout estimator can be summarized as follows:

$$\mathcal{S}_N \rightarrow \{(\mathcal{S}^{(1)}, \mathcal{T}^{(1)}), ..., (\mathcal{S}^{(k)}, \mathcal{T}^{(k)})\} \rightarrow \{\hat{\epsilon}^{(1)}, ..., \hat{\epsilon}^{(k)}\} \rightarrow \hat{\epsilon}_{RH} = 1/k \sum_{i=1}^{k} \hat{\epsilon}^{(i)} \tag{1.28}$$

where $(\mathcal{S}^{(i)}, \mathcal{T}^{(i)})$ is the i-th training-test pair, $\mathcal{S}^{(i)} \cup \mathcal{T}^{(i)} = \mathcal{S}_N$, $\mathcal{S}^{(i)} \cap \mathcal{T}^{(i)} = \emptyset$ and $\hat{\epsilon}^{(i)} = \hat{\epsilon}(A(\mathcal{S}^{(i)}), \mathcal{T}^{(i)})$.

* In this work, $\mathcal{O}(\cdot)$ expressions denote complexity orders, sometimes in terms of time (or operations) and sometimes in terms of number of parameters. The expression "f is $\mathcal{O}(g)$" indicates that, in the worst case, f is bounded tightly by g asymptotically [Cormen et al. (2003)].

In contrast to holdout estimator, repeated holdout reduces the variance of the estimator, but the repeated version maintains at the same time the negative (pessimistic) bias of the simple holdout estimator. The reduction of the variance is due to considering different splits of the available data into training-test pairs rather than a single one, reducing the dependence from a particular split.

B. k-fold cross-validation

The estimator *k-fold cross-validation* (k-cv) [Hills (1966); Anscombe (1967); Lachenbruch and Mickey (1968); Cochran (1968); Stone (1974); Burman (1989); Burman et al. (1994); Efron and Tibshirani (1995); Droge (1996); Braga-Neto and Dougherty (2004); Bengio and Grandvalet (2004, 2005)] improves repeated holdout estimator by creating more appropriate pairs training-test. The procedure consists of randomly creating a mutually exclusive partition of the data $\mathcal{P} = \{\mathcal{P}^{(1)}, ..., \mathcal{P}^{(k)}\}$, where $\mathcal{P}^{(i)} \cap \mathcal{P}^{(j)} = \emptyset$ for all $i \neq j$, $\bigcup_{i=1}^{k} \mathcal{P}^{(i)} = \mathcal{S}_N$ and each $\mathcal{P}^{(i)}$ has equal size when possible $|\mathcal{P}^{(i)}| \approx N/k$. Then, the pairs training-test $(\mathcal{S}^{(i)}, \mathcal{T}^{(i)})$ are created, where $\mathcal{S}^{(i)} = \bigcup_{j \neq i} \mathcal{P}^{(j)}$ and $\mathcal{T}^{(i)} = \mathcal{P}^{(i)}$. We said that these pairs are more appropriate than those used by repeated holdout because (i) all instances in $\mathcal{S}_N$ are used only once for testing (ii) the test sets are disjoint $\mathcal{T}^{(i)} \cap \mathcal{T}^{(j)} = \emptyset$ for all $i \neq j$. However, it still has a big dependence between training sets, specially for values of $k > 2$. The process of k-cv can be summarized as follows

$$\begin{aligned} \mathcal{S}_N \rightarrow \mathcal{P} = \{\mathcal{P}^{(1)}, ..., \mathcal{P}^{(k)}\} \rightarrow \{(\mathcal{S}^{(1)}, \mathcal{T}^{(1)}), ..., (\mathcal{S}^{(k)}, \mathcal{T}^{(k)})\} \rightarrow \\ \rightarrow \{\hat{\epsilon}_1, ..., \hat{\epsilon}_k\} \rightarrow \hat{\epsilon}_{kfcv} = 1/k \sum \hat{\epsilon}_i \end{aligned} \tag{1.29}$$

where $\hat{\epsilon}_i = \hat{\epsilon}(A(\mathcal{S}^{(i)}) = \phi_i, \mathcal{T}^{(i)})$. So the k-cv error estimator is the average of the errors committed by the classifiers $\phi_i = A(\mathcal{S}^{(i)})$ in their respective test sets $\mathcal{T}^{(i)}$.

On the other hand, many works have assessed the difficulties of estimating the variance [Dietterich (1999); Kohavi (1995a); Nadeau and Bengio (2003)]. The difficulties come from the fact that k-cv produces intermediate dependent errors $\{\hat{\epsilon}_1, ..., \hat{\epsilon}_k\}$ mainly due to the training-training dependencies. Moreover, it has been demonstrated that there is no unbiased estimator of the variance of k-cv [Bengio and Grandvalet (2004)]. For a set of corrected estimators of the variance the reader should refer to Nadeau and Bengio (2003).

The most popular k value for k-cv estimator is $k = 10$. With $k = 2$ we obtain the most independent intermediate errors $\hat{\epsilon}_i$ because not only test sets but also training sets do not share any instance among them. Note that the training sets share $(k-2)/kN$ instances. With $k = N$ the estimator is usually named *leaving-one-out* (LOO) estimator [Mosteller and Tukey (1968); Fukunaga and Hummels (1989); Kohavi (1995a)] which is supposed to slightly decrease the variance of k-cv for $k = 10$ at the expense of higher computation requirements, specially when the classifier induction algorithm used is computationally intensive.

The k-cv estimator has a negative bias (pessimistic) because the intermediate classifiers $\phi_i = A(\mathcal{S}^{(i)})$ are learned from training sets $\mathcal{S}^{(i)}$ of size $(k-1)N/k$ rather than N. Alternatively, a k-cv error estimator is an unbiased estimator of the expected prediction error on data sets of size $N - N/k$, $\epsilon_{N-N/K}$, [Bengio and Grandvalet (2005)] but it is biased for the expected prediction error on data sets of size N, ϵ_N [Braga-Neto and Dougherty (2004)]. Recently, an study of the sensitivity of k-cv estimator has presented in [Rodríguez et al. (2009)]. This work analyzes the effect of two sources of variability in the bias and variance of k-cv estimator: changes in the available data and changes in the generated partitions. The study suggest the use of repeated k-cv estimator in order to reduce the variance.

The k-cv estimator has a variance somewhat lower than repeated hold-out due to the procedure for creating training-test pairs, which regulates the number of instances shared by the training sets (training dependence). But in many situation its variance is still considered too high.

In order to reduce the variance of k-cv estimator, two useful approaches should be highlighted. It is advisable to perform a *stratified k-fold cross-validation* which differs from k-cv in the creation of the partitions $\mathcal{P} = \{\mathcal{P}_1, ..., \mathcal{P}_k\}$. The partitions are generated in a supervised way, that is, taking into account the class of the instances in the data $\mathcal{S}_N$. The created stratified partitions $\mathcal{P}_i$, and therefore each training-test pair, contain approximately the same proportion of classes as the original data sets $\mathcal{S}_N$. The stratification reduces the variance of k-cv in general, without increasing the computational requirements of the estimator.

The second approach consists of repeating m times the k-cv estimator for different partitions $\{\mathcal{P}^{(1)}, ..., \mathcal{P}^{(m)}\}$, where $\mathcal{P}^{(i)} = \{\mathcal{P}_1^{(i)}, ..., \mathcal{P}_k^{(i)}\}$. The estimator constructed by averaging the m different errors estimated by k-cv for different partitions is called *m-repeated k-fold cross-validation*. It is supposed that the repeated version stabilizes the error estimation and it could reduce the variance of the k-cv estimator [Rodríguez et al. (2009)], specially with small data sets [Kohavi (1995b)]. Note that both stratification and repetition can be performed at the same time for variance reduction. This estimator reduces the variance of k-cv estimator at the expense of multiplying the computational requirements of k-cv by m, where m is the number of repetitions.

C. Bootstrap

There are different bootstrap estimators. In this paragraph we will consider the *0.632 bootstrap* estimator [Efron and Gong (1983)]. This estimator is based on pairs training-test. In order to create the pairs, this estimator performs a random sampling of the empirical distribution or, in other words, it samples with replacement the available data $\mathcal{S}_N$. The pairs $(\mathcal{S}^{(i)}, \mathcal{T}_h^{(i)})$ consist of a training set $\mathcal{S}^{(i)}$ of size N obtained by sampling $\mathcal{S}_N$ with replacement, and a disjoint test sets, $T_h^{(i)} = \mathcal{S}_N \setminus \mathcal{S}^{(i)}$. From the sets of pairs the set $\{\hat{\epsilon}_h^{(1)}, ..., \hat{\epsilon}_h^{(k)}\}$ of errors is obtained, where $\hat{\epsilon}_h^{(i)} = \hat{\epsilon}(A(\mathcal{S}^{(i)}), \mathcal{T}_h^{(i)})$ could be considered the

intermediate holdout errors. Then, the 0.632 bootstrap estimator is obtained by first averaging the intermediate holdout errors $\hat{\epsilon}_h = 1/k \sum_{i=1}^{k} \hat{\epsilon}_h^{(i)}$, and then, computing the weighted average $\hat{\epsilon}_{.632} = 0.368\hat{\epsilon}_r + 0.632\hat{\epsilon}_h$, where $\hat{\epsilon}_r$ is the resubstitution error, $\hat{\epsilon}_r = \hat{\epsilon}(A(\mathcal{S}_N), \mathcal{S}_N)$. The 0.632 bootstrap estimator process is summarized as follows:

$$\begin{aligned} \mathcal{S}_N \rightarrow \{\mathcal{S}^{(1)}, ..., \mathcal{S}^{(k)}\} \rightarrow \{(\mathcal{S}^{(1)}, \mathcal{T}_h^{(1)}), ..., (\mathcal{S}^{(k)}, \mathcal{T}_h^{(k)})\} \rightarrow \\ \rightarrow \{\hat{\epsilon}_h^{(1)}, ..., \hat{\epsilon}_h^{(k)}\} \rightarrow \{\hat{\epsilon}_r, \hat{\epsilon}_h\} \rightarrow \hat{\epsilon}_{.632} \end{aligned} \tag{1.30}$$

Since the data set is sampled with replacement, the probability of any instance of not being chosen after N samples is $(1 - 1/N)^N \approx 0.368$; the expected number of distinct instances from the original data set $\mathcal{S}_N$ appearing in the holdout test set, $\mathcal{T}_h^{(i)}$, is thus $0.632N$. Intuitively, the weight associated to each error is related with the number of different instances contained in the test sets. It must be noted that, the estimated intermediate holdout error $\hat{\epsilon}_h$ is also *named zero bootstrap estimator*, which is more biased than 0.632 bootstrap estimator.

The 0.632 bootstrap estimator has a bias closer to zero than k-cv. This estimator learns the classifiers using training sets $\mathcal{S}^{(i)}$ of size $N = |\mathcal{S}_N|$. Moreover, the estimator tends to have a variance lower than k-cv [Kohavi (1995a)].

However, the 0.632 bootstrap estimator fails to give the expected result when a classifier is a perfect memorizer and the data sets are completely random due to the highly biased resubstitution error, $\hat{\epsilon}_r$ [Kohavi (1995a)]. For example, when the classifier is a nearest neighbor ϕ and its holdout estimated error is $\hat{\epsilon}_h$ the 0.632 bootstrap error estimator will report a highly biased estimated error $\hat{\epsilon}(\phi) = 0.632\hat{\epsilon}_h + 0.3680.0 = 0.632\hat{\epsilon}_h$ because $\hat{\epsilon}_r = 0$.

Depending on the way of generating the training and test sets, different versions of the bootstrap estimators have been proposed in the literature. The *balanced bootstrap resampling* [Chernick (1999)], following the idea of the stratified k-fold cross validation, generates training and test sets with approximately the same proportion of classes as the original data sets $\mathcal{S}_N$. The *parametric bootstrap resampling* [Efron and Tibishirani (1993); Friedman et al. (1999)] learns a factorization of the joint probability distribution $\rho(\boldsymbol{x}, c)$ and then, it samples the learned model to generate the training and test sets. Finally, the *smoothed bootstrap resampling* [Efron and Tibishirani (1993)] learns $p(\boldsymbol{x}|c)$ by means of kernel density estimation and, then, it samples the estimated density to generate the training and test sets.

D. Bolstered

The bolstered estimator [Braga-Neto and Dougherty (2004); Sima et al. (2005b,a)] rather than an estimator is a novel proposal for estimating the prediction error based on a test set. The bolstered estimators are adaptations of classic estimators which perform the computation of the prediction error using the bolstered approach. Bolstering can be applied to any error-counting estimation method.

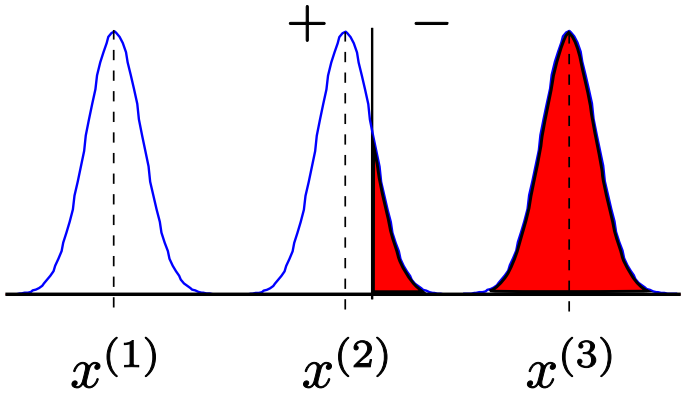


Fig. 1.3. This figure illustrates the intuition behind bolstered estimators. In this example there are three instances, $(x^{(1)}, c^{(1)})$, $(x^{(2)}, c^{(2)})$ and $(x^{(3)}, c^{(3)})$ where $c^{(1)} = c^{(2)} = c^{(3)} = +$. The decision boundary of the classifier is represented by the black solid vertical line. The correct classified examples are those at the right of the decision boundary, $(x^{(1)}, +)$ and $(x^{(2)}, +)$. On the other hand $(x^{(3)}, +)$ is misclassified. Each point contributes to the error its associated red area. Note that, in spite of $(x^{(2)}, +)$ being correctly classified, it contributes to the error its associated red area because it is close to the decision boundary. The contributions of each point are determined with a kernel based density (see Section 4.2 for further details).

The intuition behind bolstered is to perform a specialized computation of the error, distinguishing classified samples as close or far from a decision boundary: a correctly classifier point near the decision boundary can change its contribution from 0 to $1/N$ (and vice versa) [Braga-Neto and Dougherty (2004)]. Bolstered estimators replace the empirical distribution with a kernel based density (see Section 1.13.3) in order to compute the expectance of the error. Using kernel based densities bolstered estimators spread the empirical probability of each case, i.e. $1/N$, in its neighborhood. This intuition is graphically illustrated in Figure 1.3.

Bolstered error estimators improve k-fold cross-validation in terms of both bias and variance [Braga-Neto and Dougherty (2004)]. The implementations of bolstered estimators depend on the selected non-bolstered estimator, e.g. resubstitution, and the implied classifier. The main difficulty in order to implement a bolstered estimator for a concrete classifier consists of obtaining explicit expressions of the decision boundaries in order to compute the error contribution associated to each point.

1.8.2 Comparing classifiers or induction algorithms using hypothesis tests

The purpose of this section is the description of some appropriate procedures based on statistical tests that can be used for comparing the performance of

two or more classifiers based on single or multiple data sets. We consider the use of these safe procedures crucial in order to make proper comparisons, because the statistical evaluation of the experimental results has been considered an essential part of the validation of new classifier induction algorithms in the machine learning field. Before continuing, we want to include a brief explanation of the classifier comparison procedure based on the standard notation used throughout this section.

The procedures based on statistical tests usually consist of two parts: obtain a set of (estimated) performance scores for each classifier induction algorithm. The first part is usually achieved using a performance estimator, such as those presented in Section 1.8.1. However, we introduce the comparison procedures based on the classification error for the sake of simplicity.

In order to compare n different algorithms, $\{A_1, ..., A_n\}$, on N data sets, $\{\mathcal{S}^{(1)}, ..., \mathcal{S}^{(N)}\}$, we should estimate the classification error $\epsilon_{\mathcal{S}^{(j)}}(A_i)$, of each algorithm i on each data set $\mathcal{S}^{(j)}$. For the sake of brevity, the true and estimated errors of the classifier learned with the i-th induction algorithm using the j-th training set is denoted as ϵ_i^j and $\hat{\epsilon}_i^j$, respectively. The second part consists of applying the appropriate statistical test to the set of the obtained estimated errors $(\hat{\epsilon}_i^j)_{i=1,j=1}^{n,N}$.

The section starts introducing the statistical hypothesis tests focusing on null-hypothesis tests [Kanji (2006)]. Then, we focus our attention to the problem of obtaining samples of scores appropriate for classifier comparison. We take into consideration two practical situations: comparisons using a single or multiple data sets. Then, we present two methods for comparing a pair of classifiers, in single or multiple data sets, and finally we present two simple alternatives for making comparisons of n classifiers over multiple data sets.

A statistical hypothesis test is a method of making statistical decisions using experimental data. We will focus our attention on the null-hypothesis tests. Null-hypothesis tests try to answer the question: *assuming that the null hypothesis H_0 is true, what is the probability of observing a value for the test statistic that is at least as extreme as the value that was actually observed?*. These tests are designed to neither prove nor disprove hypothesis. They never set out to prove anything; their aim is to show that an idea is untenable as it leads to an unsatisfactorily small probability.

The process of hypothesis testing can be summarized in the following four steps:

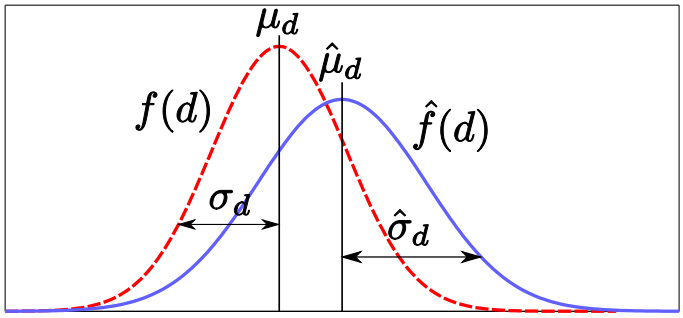
1. Formulate the practical problem in terms of hypothesis. We should first concentrate on what is called the alternative hypothesis H_1, since this is the more important from the practical point of view. The null hypothesis H_0 needs to be very simple and represents the status quo. It is basically a standard or control with which the evidence pointing to the alternative can be compared. Once both hypothesis are formulated it should be obvious whether we carry out a one- or two-tailed test. Since we are using the hypothesis tests for making comparisons between classifiers, our null

hypothesis is "the differences of the scores of the classifiers is zero" and the alternative hypothesis is "the differences of the scores of the classifiers is non-zero".

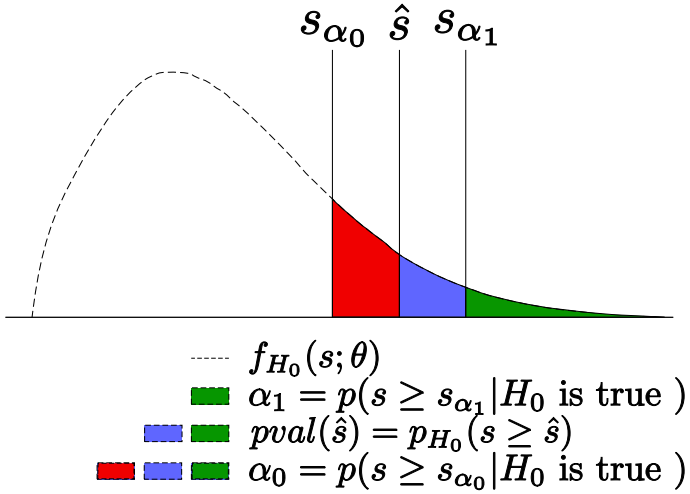
2. Calculate a statistic, s, a function purely of the data. All good test statistics should have two properties: (i) they should tend to behave differently when H_0 is true from when H_1 is true; and (ii) their probability distribution should be calculable under the assumption that H_0 is true.
3. Choose a *critical region*. The *critical region* of a hypothesis test is the set of all outcomes $S = s$ which, if they occur, cause the null hypothesis to be rejected. We could also conclude that the null hypothesis is not rejected assuming that an event of probability less than or equal to the size of the test has occurred, but generally the first option is chosen. If s lies outside the critical region, we do not reject H_0, but we should never conclude by accepting H_0.
4. Decide the significance level of the test, α, which determines the critical region of the test. This involves specifying how great a risk we are prepared to run of coming to an incorrect conclusion. The critical region is the set of outcomes for which the null hypothesis can be rejected. The probability of the test statistic falling in the critical region when the null hypothesis is correct is α. The significance level can be understood as the risk we are prepared to take in rejecting the null hypothesis when it is in fact true and, as we see below, it is called type I error. We usually set α to between 1 and 10 percent, depending on the severity of the consequences of making such an error.

In hypothesis test two types of errors could be defined: type I error and type II error. The *type I error* is the probability of rejecting the null hypothesis when it is true, $p(\text{reject } H_0|H_0 \text{ is true}) = \alpha$. The *type II error* is the probability of not rejecting the null hypothesis when it is in fact false, $p(\text{not reject } H_0|H_0 \text{ is false}) = \beta$. The *power* of a test is the probability of correctly rejecting the null hypothesis, $p(\text{reject } H_0|H_0 \text{ is false}) = (1 - \beta)$. It should be noted that the power is different from the size of the test.

Figure 1.4 illustrates the main concepts introduced for the hypothesis testing procedure. The figure shows the main statistical concepts used to compare two classifier induction algorithms, A_1 and A_2, based on the estimated errors obtained in N data sets, $\{\mathcal{S}^{(1)}, ..., \mathcal{S}^{(N)}\}$. The statistical tests consist of a null hypothesis, H_0, a set of estimated errors in different data sets for both algorithms, $\{\hat{\epsilon}_1^{(1)}, ..., \hat{\epsilon}_1^{(N)}\}$ and $\{\hat{\epsilon}_2^{(N)}, ..., \hat{\epsilon}_2^{(N)}\}$, and a statistic based on the estimated errors with a known distribution provided that the null hypothesis is true. First, in order to compare classifiers, the typical null hypothesis H_0 is stated as: *the error differences between the inducted classifiers* $\phi_1^{(i)} = A_1(\mathcal{S}^{(i)})$ *and* $\phi_2^{(i)} = A_2(\mathcal{S}^{(i)})$ *in each data set* $\{\mathcal{S}^{(1)}, ..., \mathcal{S}^{(N)}\}$ *is zero*. Usually, the alternative hypothesis H_1 can be simply stated as the negation of the null hypothesis. When it is clear that A_1 performs at least equally to A_2, we can state the alternative hypothesis H_1 as A_1 *obtains better performance than* A_2.



(a) True and estimated probability density functions, $f(d)$ and $\hat{f}(d)$, of the differences, $\{d^{(1)}, ..., d^{(k)}\}$ where $d^{(i)} = \hat{\epsilon}_1^{(i)} - \hat{\epsilon}_2^{(i)}$.



(b) Distribution of the statistic, s, under the null hypothesis, H_0.

Fig. 1.4. This figure graphically illustrates some of the concepts about statistical tests introduced in this section. The figure can be interpreted as a procedure in order to compare two classifier induction algorithms, A_1 and A_2, based on their estimated errors, $\{\hat{\epsilon}_1^{(1)}, ..., \hat{\epsilon}_1^{(N)}\}$ and $\{\hat{\epsilon}_2^{(1)}, ..., \hat{\epsilon}_2^{(N)}\}$.

Next, we need to collect the set of estimated errors $(\hat{\epsilon}_i^{(j)})_{i=1,j=1}^{2,N}$, where $\hat{\epsilon}_i^{(j)} = \hat{\epsilon}_{\mathcal{S}^{(j)}}(A_i)$. Then, the (paired) differences are computed in each data

set, $(d^{(i)})_{i=1}^{N}$, where $d^{(i)} = \hat{\epsilon}_1^{(i)} - \hat{\epsilon}_2^{(i)}$. Figure 1.4(a) shows the true probability density function of the differences $D \rightsquigarrow f(d)$ and the estimated distribution $\hat{f}(d)$ obtained from the sample of the differences $\{d^{(1)}, ..., d^{(N)}\}$. The test of the example shown in Figure 1.4 assumes that the differences are normally distributed with unknown parameters μ_D and σ_D^2, and thus, it is considered a parametric test. However, the estimated parameters, $\hat{\mu}_D$ and $\hat{\sigma}_D^2$, based on the obtained sample of differences $\{d^{(1)}, ..., d^{(N)}\}$, differs from those modeling the true distribution. The difference between the true $f(d)$ and the estimated $\hat{f}(d)$ (difference between (μ_D, σ_D^2) and $(\hat{\mu}_D, \hat{\sigma}_D)$) can result in rejectint the true hypothesis when it is true in fact, or conversely, not rejecting it when it is false. When the parametric assumption about $f(d)$ is true the probability density function tends to be estimated more accurately but, when we are not sure about the nature of $f(d)$, it is advisable to perform a non parametric test. On the other hand, most non-parametric tests are based on ranks and they do not need to explicitly estimate $f(d)$.

Once the data is collected, we compute an appropriate statistic of the data $\hat{s} = \hat{s}(\{d^{(1)}, ..., d^{(N)}\})$. The probability density function of the statistic, s, under the null hypothesis, $f_{H_0}(s; \boldsymbol{\Theta})$, must be known where $\boldsymbol{\Theta}$ are the parameters of the function. Usually, the function $f_{H_0}(s; \boldsymbol{\Theta})$ belongs to a parametric family of probability density functions and its parameter $\boldsymbol{\Theta}$ determines the shape of the function. For example, s could follow a chi squared distribution with $\boldsymbol{\Theta} = N$ degrees of freedom. Normally, the parameters $\boldsymbol{\Theta}$ are determined from the data, $\{d^{(1)}, ..., d^{(N)}\}$. Figure 1.4(b) shows the probability density function of the statistic s under the null hypothesis, $f_{H_0}(s; \boldsymbol{\Theta})$. Besides, an estimator $\hat{s}(\{d^{(1)}, ..., d^{(N)}\})$ of s should be known. Intuitively, higher values of the estimated statistics, $\hat{s}$, should indicate lower probability of H_0 being true, because $\int_{\hat{s}}^{\infty} f_{H_0}(s; \boldsymbol{\Theta}) ds$ tends to zero as $\hat{s}$ increases. The p-value of $\hat{s}$ represents the probability of obtaining a test statistic as large or larger than that which was obtained in the data, given the null hypothesis holds, $pval(\hat{s}) = p_{H_0}(s > \hat{s}) = \int_{\hat{s}}^{\infty} f_{H_0}(s; \boldsymbol{\Theta}) ds$. The p-value can be interpreted as an approximate measure of how surprised we should be by a result. The test requires to specify the size α of the test, which is a threshold used to determine whether the null hypothesis is rejected, $pval(\hat{s}) \leq \alpha$ ($\hat{s} \geq s_\alpha$), or not. As the size of the test α increases, the type I error of the test increases, provided that the null hypothesis is true. Note that by definition $p(\text{reject } H_0 | H_0 \text{ is true}) = p_{H_0}(s > s_\alpha) = \int_{s_\alpha}^{\infty} f_{H_0}(s; \boldsymbol{\Theta}) ds$ where $\alpha = pval(s_\alpha)$. On the other hand, as α increases the type II error of the test decreases, provided that the alternative hypothesis is false because the probability of falling out of the critical region decreases. Clearly, there is a trade-off between both type I error and type II error. Figure 1.4(b) shows the thresholds s_{α_0} and s_{α_1} determined by the significance levels α_0 and α_1 respectively. In our example, the null hypothesis H_0 is rejected for α_0 because $s_{\alpha_0} \leq \hat{s}$, and it is not rejected for α_1 because $s_{\alpha_1} < \hat{s}$.

Before continuing, it must be noted that different authors [Iacobucci (2005); García and Herrera (2008)] indicate that it is advisable to report the (corrected) p-values obtained with each statistical test, instead of indicating when the null hypothesis is rejected at some α value. It provides more information about the strength of rejection of the null hypothesis.

As we noted before, two groups of statistical tests could be identified: (a) parametric, and (b) non parametric. In parametric tests the probability distribution of the test statistic, s, under the null hypothesis H_0 can be only estimated by an additional assumption on the underlying probability density function of the differences $\{d^{(1)}, ..., d^{(N)}\}$. Most of the parametric tests used by the machine learning community are based on normally distributed samples (normal assumption), $D \rightsquigarrow N(d; \mu_D, \sigma_D^2)$ where D is the difference random variable. If this assumption is not true then the test loses its validity. However, in some cases the deviation of the assumption has only a minor influence on the statistical test, indicating robust procedure. A parametric test also offers greater discrimination than the corresponding distribution-free test, provided that the parametric assumption holds.

The most popular parametric test is the paired t-test, which is based on sample difference average and variance. In this section we show how to perform comparison based on hypothesis tests. In general, we suggest the use of non-parametric tests because they obtain more robust conclusions independently of the true probability density function of the difference D. In the context of comparing classifiers over multiple data sets, the parametric tests, such as t-test, are not appropriate mainly due to three reasons [Demšar (2006)]: (i) Parametric test only makes sense when the differences over data sets are commensurable. In a single data set it can be assumed that the estimated errors obtained for an induction algorithm A are commensurable, but this assumption does not hold in general with scores obtained in different data sets. So, usually, averages are meaningless. (ii) Unless the set of scores is large enough (≥ 30), the parametric tests (under normal assumption) require that the differences between the two classifiers compared are distributed normally. Again, it is possible to assume normality when using a single data set, but this assumption does not hold in general with scores obtained in different data sets. (iii) Parametric tests are based on an average value over the scores obtained in different data sets. The average allows the classifier's excellent performance on one data set to compensate for the overall bad performance, or the opposite, a total failure in one domain can prevail over the fair results on most others, and, usually, we are interested in classifiers that behave well on as many problems as possible. So, the average is susceptible to outliers and it can skew the test statistics. For the non-parametric tests, no assumption has to be made regarding the frequency distribution and the estimated differences $\{d^{(1)}, ..., d^{(N)}\}$ are not summarized in a few statistics such us the sample average and variance. When comparing two samples the non-parametric tests look at the relative ranking of the sample members. Thus, the non-parametric

tests avoid the main drawbacks of the parametric tests, at the expense of the power of the test when the parametric assumption holds.

An important issue of the estimated errors is their independence which has a direct influence on the estimated variance. When the estimated errors are dependent, the variance tends to be underestimated and the type I error of the hypothesis test could increase. When multiple data sets are available for a comparison, we can estimate independent errors $\{\hat{\epsilon}^{(1)}, ..., \hat{\epsilon}^{(N)}\}$ obtained from the data sets $\{\mathcal{S}^{(1)}, ..., \mathcal{S}^{(N)}\}$ using any of the estimators described in Section 1.8.1. We recommend those with lower variances for stable comparisons. It is also advisable to use the same partitions with each data set in order to obtain scores that can be considered paired because the variance of the differences of the estimated errors between the algorithms tends to be smaller.

Running algorithms on multiple data sets naturally gives a sample of independent measurements, but how do we proceed when we have a single data set? The most popular procedure is to perform a k-fold cross validation using the same partitions in the single data set, and use the scores obtained at each of the k pairs training-test sets. The scores are clearly not independent because, in spite of the independence of the test set used, a pair of training sets will share $k - 2/k - 1$ percentage of their cases. Nadeau and Bengio (2003) propose the corrected resampled parametric t-test which adjusts the variance based on the overlaps between subsets of examples. We suggest the use of a stratified paired k-fold cross validation estimator as suggested in Section 1.8.1. Stratification reduces the sample variance of the obtained scores for both classifiers, $var(\hat{\epsilon}_1^{(1)}, ..., \hat{\epsilon}_1^{(N)})$ and $var(\hat{\epsilon}_2^{(1)}, ..., \hat{\epsilon}_2^{(N)})$, and paired cross validations reduces the variance of the differences, $var(d^{(1)}, ..., d^{(N)})$. Besides, when it is computationally feasible we suggest the use of a repeated k-fold cross-validation [Rodríguez et al. (2009)]. All the proposed reductions of the variance are beneficial because they increase the power of the test.

In this Section we consider two types of comparisons: comparisons between two classifiers and comparisons between a set of more than two classifiers. Section 1.8.2.1 proposes two procedures in order to compare two classifiers in a single and in multiple data sets, respectively. Finally, Section 1.8.2.2 introduces a procedure in order to compare multiple classifiers using N data sets.

1.8.2.1 Comparing two classifiers or two classifier induction algorithms

Let $d^{(i)} = \hat{\epsilon}_1^{(i)} - \hat{\epsilon}_2^{(i)}$ be the difference between the estimated prediction errors of the classifiers $\phi_1^{(i)}$ and $\phi_2^{(i)}$ learned and tested using $\mathcal{S}^{(i)}$ out of N data sets, $\{\mathcal{S}^{(1)}, ..., \mathcal{S}^{(N)}\}$, using the induction algorithms A_1 and A_2, respectively, $\phi_1^{(i)} = A_1(\mathcal{S}^{(i)})$ and $\phi_2^{(i)} = A_2(\mathcal{S}^{(i)})$. When the comparison is performed over N different data sets, in general, the estimated errors are obtained using one of the procedures introduced in Section 1.8.1.3.

On the other hand, when the comparison is performed in a single data set, S, usually the comparison is based on the N intermediate errors obtained by the error estimators described in Section 1.8.1.3. For example, a comparison can be based on the $k = N$ intermediate errors computed by the k-cv estimator (see Paragraph B. in Section 1.8.1.3). A computationally intensive alternative consists of computing multiple estimations based on the estimators introduced in Section 1.8.1 using randomly generated (different) training-test pairs.

In order to reduce the sample variance of the differences $\{d^{(1)}, ..., d^{(N)}\}$, it is advisable to perform a paired estimation of the error, where the estimated errors $(\hat{\epsilon}_1^{(i)}, \hat{\epsilon}_2^{(i)})_{i=1}^N$ are obtained by the same estimator using the same training-test pairs (see Section 1.8.1.3). This reduction of the variance tends to improve the power of the hypothesis test used.

A. Over multiple data sets

In order to compare two algorithms based on multiple data sets, we propose the Wilcoxon signed-ranks paired test [Wilcoxon (1945)], following the suggestion of Demšar (2006).

The Wilcoxon signed-rank test is a non-parametric alternative to the paired t-test, which ranks the differences in performances of two classifiers for each data set, ignoring the signs, and compares the ranks for the positive and negative differences. The differences are ranked according to their absolute values (average ranks are assigned in case of ties). Let R^+ be the sum of ranks for the data sets on which the second algorithm outperformed the first, and R^- the sum of ranks for the opposite. Ranks of $d^{(i)} = 0$ are equally split among the sums.

$$R^+ = \sum_{d^{(i)}>0} rank(d^{(i)}) + \frac{1}{2} \sum_{d^{(i)}=0} rank(d^{(i)})$$
$$R^- = \sum_{d^{(i)}<0} rank(d^{(i)}) + \frac{1}{2} \sum_{d^{(i)}=0} rank(d^{(i)})$$

Let R_{min} be the smaller of the sums, $R_{min} = min(R^+, R^-)$. Then, the statistic

$$z = \frac{R_{min} - \frac{1}{2}N(N+1)}{\sqrt{\frac{1}{24}N(N+1)(2N+1)}} \tag{1.31}$$

has approximately a standard normal distribution, $\mathcal{N}(\mu = 0, \sigma = 1)$. This test is safer than t-test since it does not assume normally distributed differences. Besides, the outlier differences have less effect on the Wilcoxon test.

B. In a single data set

Following the proposal of Nadeau and Bengio (2003), we suggest the use of corrected resampled t-test for comparing classifiers in a single data set. The

samples of scores are obtained using a stratified paired k-fold cross validation, as previously suggested. The null hypothesis is "the mean difference is zero".

The statistic

$$s = \frac{\overline{d}}{\sqrt{\frac{1}{k} + \frac{\sigma^2}{k-1}}} \tag{1.32}$$

is distributed under the null hypothesis according to a t distribution with $k-1$ degrees of freedom, where $\overline{d} = \frac{1}{k}\sum_{i=1}^{k} d^{(i)}$ is the sample mean and $\sigma^2 = \frac{1}{k-1}\sum_{i=1}^{k}(d^{(i)} - 1/k\sum_{i=1}^{k} d^{(i)})^2$ is the quasivariance of $\{d^{(1)}, ..., d^{(k)}\}$ respectively.

1.8.2.2 Comparing many classifier induction algorithms

This section presents two procedures for comparing multiple classifier induction algorithms over multiple data sets. The procedure consists of two steps based on hypothesis tests: (i) to check if there are significative differences between the estimated errors of some of the classifier induction algorithms included in the study. This step is performed by means of a single test (see Paragraph A.). The null-hypothesis being tested is H_0:*all classifiers obtain the same error*. If the null hypothesis is rejected, it can be concluded that there are statistically significant differences between the classifiers, and we continue with the second step: (ii) to check for each pair of algorithms if there are significative differences by means of a post-hoc test (see Paragraphs B. and C.).

Step (ii) consists of $\binom{n}{2}$ different comparisons. The null-hypotheses being tested are $H_0^{(i,j)}$: *algorithms A_i and A_j obtain the same error*, for all $i \neq j$. In general, the probability of rejecting the null hypothesis when it is true (type I error) is the size of the test, α. If multiple hypothesis tests $(n(n-1)/2)$ are performed, the probability of rejecting some of the hypothesis $H_0^{(i)}$ provided they are true is $1-(1-\alpha)^{n(n-1)/2}$ and, clearly, it tends to 1 as n increases. In order to maintain a low type I error when multiple comparisons are performed, usually, the p-values of the individual comparisons are corrected. For example, the Bonferroni correction [Shaffer (1995)] divides the size of the individual tests by the number individual comparisons performed, but this correction usually is too conservative and it tends to have a high type II error associated.

The hypothesis being tested, belonging to a family of all pairwise comparisons, are logically interrelated so that not all combinations of accepted and rejected hypotheses are possible. Due to the incorporation of information about the interrelation of the tests, procedures with more power and lower type II error can be designed (see Paragraph C.). In García and Herrera (2008) there is an extensive overview of the most relevant procedures including a precise guideline for choosing the most appropriate test according to the number of algorithms to be tested and the number of available data sets.

Next, we introduce Friedman's procedure [Friedman (1937)] for the first step of the multiple comparison and two post-hot tests for the second step:

Nemenyi's [Nemenyi (1963)] and Shaffer's static [Shaffer (1986)] procedures. However, following the suggestion of García and Herrera (2008), we recommend the use of Friedman's plus Shaffer's static procedures because Shaffer's static procedure has more power than Nemenyi's procedure. All these tests are based on the ranks $r_i^{(j)}$ obtained by the estimated error $\hat{\epsilon}_i^{(j)}$ (ranked from the lowest to the highest) in the set of the scores obtained for the j-th data set $\{\hat{\epsilon}_1^{(j)}, ..., \hat{\epsilon}_n^{(j)}\}$. Note that, if the scores measure the goodness of the algorithms, e.g. accuracy, the scores are ranked from the highest to the lowest.

It should be noted that in order to compare multiple classifier induction algorithms in a single data set, similar procedures can be used based on the intermediate errors obtained with k-fold cross validation or bootstrap estimators (see Section 1.8.1.3).

A. Friedman's procedure

Friedman's statistics [Friedman (1937); Iman and Davenport (1980)] is based on the average ranks of each classifier induction algorithm A_i:

$$R_i = \frac{1}{N} \sum_{j=1}^{N} r_i^{(j)} \tag{1.33}$$

Friedman's statistics [Friedman (1937)]

$$\chi_F^2 = \frac{12N}{n(n+1)} [\sum_{i=1}^{n} R_i^2 - \frac{n(n+1)^2}{4}] \tag{1.34}$$

follows, under the null hypothesis, a chi-square distribution with $(n-1)$ degrees of freedom.

Iman and Davenport (1980) showed that Friedman's ϕ_F^2 statistic is undesirably conservative and derived a better statistic

$$F_F = \frac{(N-1)\chi_F^2}{N(n-1) - \chi_F^2} \tag{1.35}$$

which follows a F-distribution with $(n-1) \times (n-1)(N-1)$ degrees of freedom.

Friedman's test is an omnibus test which can be used to carry out these types of comparisons. It allows to detect differences considering the global set of classifiers. Once Friedman's test rejects the null hypothesis, we can proceed with a post-hoc test in order to find the concrete pairwise comparisons which produce differences.

B. Nemenyi's procedure

Nemenyi's procedure [Nemenyi (1963)]considers all the possible pairwise combinations of rejection of the null hypothesis and, thus, it has less power than Shaffer's static procedure; it imposes more severe corrected p-values because

it considers all the $\binom{n}{2}$ pairwise comparisons. Sometimes the Friedman test reports a significant difference but the Nemenyi post-hoc test fails to detect it due to lower power.

The test statistics for comparing the i-th and j-th classifier is

$$s = (R_i - R_j)\sqrt{6N}n(n+1) \tag{1.36}$$

where R_i is the average rank computed for the Friedman test. The z value is used to find the corresponding probability (p-value) from the table of a normal distribution which is then compared with an appropriate significance level α (Table A1 in Sheskin (2003)).

So, since the size of the test is constant for all pairwise comparisons, α, a critical rank difference CD_α can be defined as follows:

$$CD = \frac{s_\alpha}{\sqrt{6N}n(n+1)} \tag{1.37}$$

where $pval(s_\alpha) = \alpha$. Given the comparison between any pair of algorithms A_i and A_j, its null hypothesis $H_0^{(i,j)}$ can be rejected if $|R_i - R_j| \geq CD_\alpha$.

As we noted before, Nemenyi's procedure imposes severe corrected values because it considers that all the tests are independent. But the hypotheses being tested, belonging to a family of all pairwise comparisons, are logically interrelated so that not all combinations of true and false hypothesis are possible.

C. Shaffer's static procedure

Shaffer's static procedure [García and Herrera (2008)] follows a step down method, such us Holm's procedure [Holm (1979)]. Holm's procedure was also described in [Demšar (2006)] but it was used for comparisons of multiple classifiers involving a control method. It adjusts the test size α in a step down method. Let $p_1, ..., p_m$ the ordered p-values (smallest to largest) and $H_1, ..., H_m$ be the corresponding $m = \binom{n}{2}$ hypotheses. Holm's procedure rejects $H_1, ..., H_{(i-1)}$ if i is the smallest integer such that $p_i > \alpha/(m - i + 1)$. This procedure assumes at each step i that the null hypothesis $H_i, ..., H_m$ can be true. However, if the previous $i - 1$ hypothesis have been rejected this assumption is not true since the null hypotheses are interrelated.

In Shaffer's static procedure, at step i, the hypothesis H_i is rejected if $p_i \leq \alpha_i = \alpha/t_i$, where t_i is the maximum number of hypotheses which can be true given that any $(i - 1)$ hypotheses are false. For $i = 2 + \sum_{j=1}^{k-1} N - j, ..., 1 + \sum_{j=1}^{k} N - j = 2 + (2N - k)(k - 1)/2, ..., 1 + (2N - k - 1)k/2$ with $k = 1, ..., N - 2$, the correction coefficient t_i is given by

$$t_i = \binom{N-k}{2} \tag{1.38}$$

where $t_1 = \binom{N}{2}$. For example, with $N = 5$: for $i = 1$ we have that $k = 0$ and $t_i = \binom{5}{2}$, $i = 2, ..., 5$ we have $k = 1$ $t_i = \binom{4}{2}$, $i = 6, ...8$ we have $k = 2$ and $t_i = \binom{3}{2}$ and, finally, for $i = 9, 10$ we have $k = 3$ and $t_i = \binom{2}{2}$.

There are more sophisticated procedures which take into account the concrete rejected hypotheses, $\{H_1, ..., H_{i-1}\}$, in order to reduce at each step i the maximum number of null hypotheses among $H_i, ..., H_m$ that can be true, e.g. Shaffer's dynamic procedure. These procedures improve the power with respect to Shaffer's static procedure due to the more efficient use of the interrelation between the null hypotheses. For further details on these procedures the reader may consult [García and Herrera (2008)].

1.8.3 Bias plus variance analysis of the error

In this section we analyze the error of a classifier by decomposing it into different sources of error following a procedure proposed by Kohavi and Wolpert (1996).

The concept of bias-variance decomposition was introduced to machine learning for mean squared error by German et al. (1992). Later versions for zero-one-loss functions were given by Friedman (1997), Kohavi and Wolpert (1996), Domingos (2000) and James (2003). The decomposition explained in this subsection was proposed by Kohavi and Wolpert (1996) and it is the most popular among the proposed decompositions in machine learning. It decomposes the classification error. The decomposition has become popular due to its simplicity. It decomposes the error into two additive sources called bias and variance. Besides, the intuitions behind both terms are similar to the intuitions behind the bias and variance of an estimator (See Section 1.7). The bias-variance decomposition of the expected error can be useful to explain the behaviors of different algorithms [van der Putten and van Someren (2004)]. The decomposition of the classification error proposed by Friedman (1997) seems to be more appropriate for classification but it decomposes the error into multiplicative terms of harder interpretation which do not correspond to the bias and variance terms of an estimator.

Kohavi and Wolpert (1996) define the classification error as:

$$\epsilon_A = \int f(\boldsymbol{x}) \sum_{c=1}^{r} p(c|\boldsymbol{x})(1 - \hat{p}_A(c|\boldsymbol{x}))d\boldsymbol{x} \tag{1.39}$$

where $\hat{p}_A(c|\boldsymbol{x})$ is related to the classification associated to the classifiers obtained using the inductor A from data and $p(\cdot)$ is the true distribution of the domain. The definition is given in terms of a continuous multidimensional random variable $\boldsymbol{X}$, for the sake of simplicity. The generalization to multidimensional mixed random variables is straightforward. It must be noted that this error could be considered a calibration measure since it takes into consideration the estimated probabilities $\hat{p}_A(c|\boldsymbol{x})$.

But, how are the quantities $\rho(c, \boldsymbol{x}) = p(c|\boldsymbol{x})f(\boldsymbol{x})$ and $\hat{p}_A(c|\boldsymbol{x})$ estimated from data, $\mathcal{S}$? Following the proposal of Kohavi and Wolpert (1996), the data set is divided into two partitions $\mathcal{P}_1$ and $\mathcal{P}_2$, where $\mathcal{P}_1 \cup \mathcal{P}_2 = \mathcal{S}$ and $\mathcal{P}_1 \cap \mathcal{P}_2 = \emptyset$. $\mathcal{P}_1$ is used in order to generate, by random sampling with replacement, N training sets $\{\mathcal{S}^{(1)}, ..., \mathcal{S}^{(N)}\}$ and $\mathcal{P}_2$ is the single test set $\mathcal{T} = \mathcal{P}_2$. For each data set $\mathcal{S}^{(i)}$ we learn a classifier $\phi_i = A(\mathcal{S}^{(i)})$ and we classify the entire test set $\mathcal{T}$, $\{c_i^{(1)}, ..., c_i^{(n)}\}$, where n is the size of the test set $\mathcal{T}$. Then, the a posteriori probability distribution of the class at each $\boldsymbol{x}$, where for some c $(\boldsymbol{x}, c) \in \mathcal{T}$, is estimated as:

$$\hat{p}_A(c|\boldsymbol{x}^{(i)}) = \frac{1}{N} \sum_{j=1}^{N} 1(c; c_j^{(i)}) \tag{1.40}$$

where $1(c; c') = 1$ iff $c = c'$ and zero otherwise. Note that the estimation $\hat{p}_A(c|\boldsymbol{x}^{(i)})$ can be seen as the average classification of $\boldsymbol{x}^{(i)}$ using ϕ_i for $i = 1, ..., N$. On the other hand the estimation of the true a posteriori distribution $\hat{p}(c|\boldsymbol{x}) = l/m$ where l is the number of times that the instance $(c, \boldsymbol{x})$ appears in $\mathcal{T}$ and m is the number of times that for any c' $(c', \boldsymbol{x})$ appears in $\mathcal{T}$. $\hat{p}(c|\boldsymbol{x})$ is typically 1 when $(\boldsymbol{x}, c) \in \mathcal{T}$ and zero otherwise. Then, using these quantities Equation 1.39 is estimated as follows:

$$\hat{\epsilon}_A = \frac{1}{n} \sum_{i=1}^{n} \hat{p}(c|\boldsymbol{x})(1 - \hat{p}_A(c|\boldsymbol{x})) \tag{1.41}$$

The bias-variance decomposition of the expected error proposed in Kohavi and Wolpert (1996) is given by:

$$\epsilon_A = \int f(\boldsymbol{x})(\sigma_{\boldsymbol{x}}^2 + bias_{\boldsymbol{x}}^2 + var_{\boldsymbol{x}})d\boldsymbol{x} \tag{1.42}$$

where $\boldsymbol{x}$ is an instance of the test set, $\sigma_{\boldsymbol{x}}^2$ is the "intrinsic" target noise, $bias_{\boldsymbol{x}}^2$ is the square bias and $var_{\boldsymbol{x}}$ is the variance associated to the instance $\boldsymbol{x}$. Each term in Equation 1.42 is defined as:

$$\sigma_{\boldsymbol{x}}^2 = \frac{1}{2}(1 - \sum_{c=1}^{r} p(c|\boldsymbol{x})^2) \qquad bias_{\boldsymbol{x}}^2 = \frac{1}{2} \sum_{c=1}^{r} (p(c|\boldsymbol{x}) - \hat{p}_A(c|\boldsymbol{x}))^2$$
$$var_{\boldsymbol{x}} = \frac{1}{2}(1 - \sum_{c=1}^{r} \hat{p}_A(c|\boldsymbol{x})^2) \tag{1.43}$$

The averaged squared bias (or bias term of the decomposition) is defined as $bias^2 = \int f(\boldsymbol{x}) bias_{\boldsymbol{x}}^2 d\boldsymbol{x}$, and the averaged variance (or variance term) is defined as $var = \int f(\boldsymbol{x}) var_{\boldsymbol{x}} d\boldsymbol{x}$. The target noise is related with the expected error of the Bayes classifier. Therefore, it is independent of the learning algorithm given the training set, and it is characteristic of each domain. In

practice, if there are two instances in the test set with the same configuration for the predictors and a different value for the class, the estimated "intrinsic" noise is positive, otherwise it is zero [Kohavi and Wolpert (1996)]. Thus, it is considered zero given the data sets when there are not repeated instances. When the noise is zero the error of the data sets included will decompose into bias plus variance. The bias component can be seen as the error due to the incorrect fitness of the average classification of ϕ_i for $i = 1, ..., N$, $\hat{p}_A(c|\boldsymbol{x})$, to the target posterior distribution, $p(c|\boldsymbol{x})$. By contrast, the variance component measures the variability of the average classification, $\hat{p}_A(c|\boldsymbol{x})$, which is independent of $p(c|\boldsymbol{x})$. From these concepts, we can hypothesize that bias and variance terms become lower and higher, respectively, as the number of parameters needed to model the classifier grows (as classifier complexity and sensitivity to changes in the training set increases).

1.9 Performance measures

In this section we introduce some scores used to measure the performance of the classifiers (or induction algorithms). We have divided the scores in two parts: scores based on the confusion matrix and based on receiver operating characteristic (ROC) curve.

1.9.1 Confusion matrix

This subsection includes some scores based on the confusion matrix (or contingency table). Let us have a domain with a class variable C with r states, $\{c_1, ..., c_r\}$, a classifier ϕ and a test set $\mathcal{T}$, its associated contingency table M is a matrix $r \times r$ with elements $M(i, j)$, representing the number of instances $(\boldsymbol{x}, c_j) \in \mathcal{T}$ classified as $c_i = \phi(\boldsymbol{x})$.

Most of the scores based on a confusion matrix are defined for problems with two class labels, the positive p and the negative n (binary classification problems). A confusion matrix M of dimensions $r \times r$ can be understood as r confusion matrix 2×2 (bivariate confusion matrix), i.e. the i-th matrix, M^i is constructed from M considering as the positive class $c_+ = c_i$ and negative $c_- = \bigcup_{j \neq i} c^j$: $M^i(1,1) = M(i,i)$, $M^i(2,2) = \sum_{j \neq i} M(j,j)$, $M(1,2) = \sum_{j \neq i} M(i,j)$ and $M(2,1) = \sum_{j \neq i} M(j,i)$.

For the sake of simplicity, let us consider a binary classification problem $c = \{c_+, c_-\}$. Given an instance $(\boldsymbol{x}, c)$ there are four possible results depending on the values of c and $\phi(\boldsymbol{x}) = c'$:

- A *true positive* occurs when $c = c' = c_+$.
- A *true negative* occurs when $c = c' = c_-$.
- A *false positive* occurs when $c = n$ and $c' = c_+$.
- A *false negative* occurs when $c = p$ and $c' = c_-$.

Given a test set $\mathcal{T} = \{(\boldsymbol{x}^{(1)}, c^{(1)}), ..., (\boldsymbol{x}^{(n)}, c^{(n)})\}$ and a classifier ϕ the confusion matrix M contains the number of true positives (TP) $M_{1,1}$, of false positives (FP) $M_{1,2}$, of false negatives (FN) $M_{2,1}$ and the number of true negatives (TN) $M_{2,2}$. We define P as the number of positive cases, $P = TP + FN$, and N as the number of negative cases $N = TN + FP$.

Given a bivariate classification matrix M the following scores can be computed:

- True positive rate (TPR), *recall* or sensitivity: $TPR = TP/P$
- False positive rate (FPR) or *type I error*: $FPR = FP/N$
- True negative rate (TPR), *specificity* or $(1 - FPR)$: $TNR = TN/N$
- False negative rate (FNR), *type II error* or $(1 - TPR)$: $FNR = FN/P$
- *Classification error* (ϵ) or 1-accuracy: $\epsilon = (FP + FN)/(N + P)$
- Positive predictive value (PPV): $PPV = TP/(TP + FP)$
- *False discovery rate* (FDR) or $(1 - PPV)$: $FDR = FP/(TP + FP)$
- Negative predictive value (NPV): $NPV = TN/(TN + FN)$

1.9.2 Receiver operating characteristic

A receiver operating characteristic (ROC) curve is a graphical plot of the sensitivity (TPR) vs 1-specificity (FPR) for a binary classification problem as the *classification threshold* varies, $0 \leq \tau \leq 1$. The classification threshold can be understood in terms of the following loss function:

$$L_\tau = \begin{pmatrix} 0 & , \tau \\ 1 - \tau & , 0 \end{pmatrix} \tag{1.44}$$

Given the loss function L_τ, a training set S, a domain defined by $p(\boldsymbol{x}, c)$ and a learning algorithm which takes into account the loss function, $A(S; L_\tau) = \phi_\tau$, its associated ROC curve is defined as the curve $TPR(\phi_\tau)$ vs. $FPR(\phi_\tau)$ for $0 \leq \tau \leq 1$.

An estimation based on a test set $\mathcal{T}$ will be the curve $TPR(\phi_\tau; \mathcal{T})$ vs. $FPR(\phi_\tau; \mathcal{T})$ for $0 \leq \tau \leq 1$. Note that each pair $TPR(\phi_\tau; \mathcal{T})$ and $FPR(\phi_\tau; \mathcal{T})$ represents a confusion matrix. The ROC space is defined by FPR and TPR as X and Y axes respectively. The perfect classification (often impossible) point is the (0,1). A completely random guess would give a point along the diagonal line from $(0, 0)$ to $(1, 1)$ (*line of no-discrimination*). Points above and below the line of no-discrimination indicate better and worse classifications than the random guess respectively. The information contained in the ROC curve is usually summarized as a single value: the area under the ROC curve (AUC). This area can be interpreted as the probability that when we randomly pick one positive and one negative example, the classifier will assign a higher score to the positive example than to the negative.

1.10 Information theory

In this section we introduce some useful concepts of information theory [Shanon (1948); Kullback (1959); Cover and Thomas (1991); Jakulin (2002)] for the correct understanding of this document. The entropy, one of the main concepts of information theory, was originally proposed in physics in the context of equilibrium thermodynamics and later as a measure of disorder through developments in statistical mechanics [Bishop (2006)]. However, all the concepts in this introduction are presented from an information theory point of view.

We start considering a unidimensional discrete random variable X and we ask how much information is received when we observe a particular value x, also known as an *event*. The amount of information given by the event x, $i(x)$, can be understood as the degree of surprise of the value x. The amount of information $i(x)$ is inversely proportional to the probability $p(x)$ and thus, the higher $p(x)$, the less information it provides. The form of $i(x)$ can be found by noting that the information gain of observing two unrelated events $X = x$ and $Y = y$ should be the sum of the information gained from each of them separately $i(x, y) = i(x) + i(y)$. Besides, two unrelated events will be statistically independent and thus $p(x, y) = p(x)p(y)$. From these two relationships, it is easily shown that $i(x)$ should be given by the logarithm of $p(x)$ and so we have that:

$$i(x) = -\log_2 p(x) \tag{1.45}$$

where the negative sign ensures that information is positive or zero. The choice of the basis for the logarithm is arbitrary, and we will adopt the convention of using logarithms of base 2. In this case the unit of the amount of information is a bit (binary digit).

Let us suppose now that a sender wants to transmit the value of a random variable to a receiver. The average amount of information that they transmit in the process is obtained by taking the expectation of the amount of information of each possible event x:

$$H(X) = E_X[i(X)] = \sum_x p(x)i(x) = -\sum_x p(x)\log_2 p(x) \tag{1.46}$$

The generalization of the entropy to d-dimensional discrete random variables $\boldsymbol{X} = (X_1, ..., X_d)$ is straightforward by considering the entropy of the unidimensional discrete random variable X which is the cartesian product of X_i for all i, $X = X_1 \times ... \times X_d$.

The average amount of information transmitted by events x of a variable X is called the entropy of the variable X. Analogously, the entropy $H(X)$ is the information needed to describe the random variable X. The entropy has some useful properties:

- The noiseless coding theorem [Shanon (1948)] states that the entropy is a lower bound for the number of bits needed to transmit the state of a random variable. Therefore, the entropy can be interpreted as the amount of information of a variable.
- A nonuniform distributed variable has smaller entropy than its uniform counterpart. Thus, for a fixed number of states, r, the upper bound for the entropy is given by $-\log_2 r$.

The entropy can be considered as the elementary quantity of information theory and the measures of information theory presented in this section can be decomposed into sums and differences of entropies.

The definition of entropy can be extended to include distributions over continuous random variables X as follows. First, we divide the variable X into bins of size Δ. Then, assuming that $f(x)$ is continuous, the mean value theorem tells us that, for each such bin $(i\Delta, (i+1)\Delta)$, there must exist a value x_i such that:

$$\int_{i\Delta}^{(i+1)\Delta} f(x)dx = f(x_i)\Delta \tag{1.47}$$

We can now discretize the continuous variable X by assigning a density of $f(x_i)$ to any value x whenever it falls in the interval $(i\Delta, (i+1)\Delta]$. Therefore, the probability of the i-th interval is $f(x_i)\Delta = p(x_i^{\Delta})$. This gives a discrete random variable X^{Δ} for which the entropy takes the form

$$\begin{aligned} H(X^{\Delta}) &= -\sum_i f(x_i)\Delta \log_2(f(x_i)\Delta) \\ &= -\log_2 \Delta - \sum_i f(x_i)\Delta \log_2(f(x_i)) \end{aligned} \tag{1.48}$$

We now omit the first term $-\log_2 \Delta$ in Equation 1.48 and we consider the limit $\Delta \to 0$ of the second term $-\sum_i f(x_i)\Delta \log_2(f(x_i))$. This term approaches the integral of $-f(x)log_2 f(x)$ in the limit, and this quantity is known as *differential entropy.*

$$h(X) = \lim_{\Delta \to 0}\{-\sum f(x_i)\Delta \log_2 f(x_i)\} = -\int f(x)log_2 f(x)dx \tag{1.49}$$

As in the case of the entropy, the generalization of the differential entropy to multidimensional continuous random variables $\boldsymbol{X}$ is straightforward.

We see that the entropy and the differential entropy differ in $-\log_2 \Delta$, which diverges in the limit $\Delta \to 0$. This reflects that a large number of bits is required on average to specify a continuous variable very precisely. The differential entropy shares many of the mathematical properties of the entropy. The main particularities of the continuous and categorical formulation are:

- The entropy of a categorical variable is invariant to changes in the order of the states of the variable.

- The differential entropy is invariant to the translation, $h(X + a) = h(X)$, but it depends on the scale of the variable used, $h(X * a) = h(X) + \log_2 a$, for $a \in \mathbb{R}$.

In the remainder of this section we will introduce the formulation for discrete random variables indicating the main differences with respect to the continuous formulation. As we noted in Section 1.3, the continuous version of the formulation consists of replacing the sums by integrals.

1.10.1 Conditional entropy and mutual information

Given a pair of unidimensional discrete random variables X and Y, let us suppose that we have a joint distribution $p(x, y)$ from which we draw pairs of values (x, y). If a value x is known, then the additional information needed to specify the corresponding value y is given by the relative amount of information $i(y|x) = -\log_2 p(y|x)$. The conditional entropy of Y given X is defined as the average of the relative amount of information of x given y, $i(y|x)$

$$H(Y|X) = E_X[H(Y|X)] = E_{(X,Y)}[i(Y|X)] = -\sum_{x,y} p(x, y) \log_2 p(y|x) \tag{1.50}$$

The conditional entropy $H(Y|X)$ measures the average amount of information transmitted by events y of a variable Y when the events x of X are known. It also can be defined as the information needed to describe Y given X. It can be easily seen, using the product rule, that the conditional entropy satisfies the relation

$$H(X, Y) = H(X) + H(Y|X) \tag{1.51}$$

One interpretation is that the information needed to describe X and Y is given by the sum of the information needed to describe X alone plus the information needed to describe Y given X. This is known as the *chain rule of the entropy*. The *differential conditional entropy* $h(Y|X)$ is defined analogously and inherits some of the properties of the conditional entropy.

Let us suppose that we have a discrete random variable X with an unknown probability distribution $p(x)$ and that we have modeled $p(x)$ using the probability distribution $q(x)$. If we use $q(x)$ to construct a coding scheme for the purpose of transmitting values of X to a receiver, then the average additional amount of information required to specify the value of X as a result of using $q(x)$ instead of the true distribution $p(x)$ is given by:

$$\begin{aligned} KL(p; q) &= -\sum_x p(x) \log_2 q(x) - \left(-\sum_x p(x) \log_2 p(x) \right) \\ &= -\sum_x p(x) \log_2 \frac{q(x)}{p(x)} \end{aligned} \tag{1.52}$$

This is known as *Kullback-Leibler divergence* or *relative entropy* [Kullback and Leibler (1951)], between the distributions $p(x)$ and $q(x)$. Note that this quantity is not symmetrical, that is, $KL(p;q) \neq KL(q;p)$. The Kullback-Leibler divergence satisfies that $KL(p;q) \geq 0$ with equality if, and only if, $p(x) = q(x)$. The Kullback-Leibler divergence can be analogously defined for probability density functions replacing sums by integrals.

Let us now consider a joint probability distribution for two variables X and Y given by $p(x, y)$. If the variables are independent, then their joint distribution will factorize into the product of their marginal $p(x, y) = p(x)p(y)$. If the variables are not independent, we can gain some idea of whether they are 'close' to being independent by considering the Kullback-Leibler divergence between the joint distribution and the product of the marginal distributions. This quantity is known as *mutual information* (or information gain) between X and Y [Cover and Thomas (1991)]:

$$I(X;Y) = KL(p(X,Y);p(X)p(Y)) \tag{1.53}$$

Since $I(X;Y)$ is a Kullback-Leibler divergence, it inherits the property $I(X;Y) \geq 0$ with equality if, an only if, X and Y are independent ($X \perp Y$). So the mutual information can be decomposed as follows:

$$\begin{aligned} I(X;Y) &= H(X) - H(X|Y) && (1.54)\\ &= H(Y) - H(Y|X) \\ &= H(X) + H(Y) - H(X,Y) && (1.55) \end{aligned}$$

Note that the mutual information is symmetric $I(X;Y) = I(Y;X)$. So, we can view the mutual information as the reduction on the uncertainty about X when Y is known (and vice versa). In contrast to the differential entropy and the conditional differential entropy, the continuous version of the mutual information (differential mutual information) is invariant to changes in the scale, i.e., given two unidimensional continuous variables X and Y and for any $a, b \neq 0$, $I(X;Y) = I(a * X; b * Y)$.

An interesting feature of the mutual information is that it measures the degree of (non-linear) dependence between two variables. Now the question is, is it possible to measure the degree of conditional dependence based on a statistic similar to the mutual information? Let us consider a conditional distribution of two variables X and Y given a third variable Z, $p(x, y|z)$. The mutual information of X and Y given a value z is defined as:

$$I(X;Y|Z=z) = KL(p(X,Y|z);p(X|z)p(Y|z)) \tag{1.56}$$

The conditioned mutual information of X and Y given Z is defined as:

$$I(X;Y|Z) = E_Z[I(X;Y|Z)] \tag{1.57}$$

The conditioned mutual information measures the degree of (non-linear) conditional dependence between X and Y given Z. Since it is the expected value

of a Kullback-Leibler divergence, it inherits the property $I(X;Y|Z) \geq 0$ with equality if, an only if, X and Y are conditionally independent given Z ($X \perp Y|Z$). But, is it possible to generalize the dependence between a pair of variables to an arbitrary set of variables? To answer this question we have to introduce the concept of interaction information.

1.10.2 Interaction information

Interaction information , [Jakulin (2002)] is a metric based on the information-theoretic concept of entropy. It reflects the degree of (non-linear) dependence (or interaction) between a set of variables. It is based on information-theoretic notion of entropy. Given a set of discrete random variables $\boldsymbol{X} = (X_1, ..., X_k)$ the *k-way interaction information* is defined as:

$$I(X_1; ...; X_k) = -\sum_{\boldsymbol{Y} \subseteq \boldsymbol{X}} (-1)^{k-|\boldsymbol{Y}|} H(\boldsymbol{Y}) \tag{1.58}$$

Note that Equation 1.58 is easily generalizable to continuous random variables, replacing the entropy term $H(\boldsymbol{Y})$ by the differential entropy $h(\boldsymbol{Y})$.

Interaction information reflects the degree of interaction between a set of variables, $\boldsymbol{X}$. It can be seen as a generalization of the mutual information for measuring interactions between a set of variables $\boldsymbol{X}$ rather than between two univariate random variables X_i, X_j. Mutual information and 2-way interaction information are equivalent, i.e. given two univariate discrete random variables X and Y, $I(X;Y) = -\sum_{\boldsymbol{Z} \subseteq (X,Y)} (-1)^{k-|\boldsymbol{Z}|} H(\boldsymbol{Z}) = H(X)+H(Y)-H(X,Y)$.

Now we will focus our attention on the *3-way interaction information* between three unidimensional random variables X, Y and Z. The 3-way interaction information can be defined in terms of mutual information [Jakulin and Bratko (2004)]:

$$\begin{aligned} I(X;Y;Z) &= -H(X;Y;Z) + H(X;Y) + H(X;Z) + H(Y;Z) \\ &\quad -H(X) - H(Y) - H(Z) \\ &= I(X;Y|Z) - I(X;Y) \\ &= I(X;Z|Y) - I(X;Z) \\ &= I(Y;Z|X) - I(Y;Z) \end{aligned} \tag{1.59}$$

If we analyze the interplay of the attributes X and Y in Equation 1.59 we differentiate two parts. The mutual information $I(X;Y)$ measures strength of the dependence between X and Y. The conditioned mutual information $I(X;Y|Z))$ measures the strength of the conditional dependence of X and Y given Z. The interaction information measures the difference between the strength of the dependence and the strength of the conditional dependence. Therefore, the higher the interaction information, the stronger is the conditional dependence assumption with respect to the dependence assumption.

Now we will briefly introduce an intuition which relates interaction information with the error of classification models. This intuition is explained in more detail in Chapter 5. Let $(\boldsymbol{X}, C) = (X_1, ..., X_d, C)$ be a multidimensional discrete random variable with joint probability distribution $p(X_1, ..., X_d, C)$. Consider the sets of classifiers $\mathcal{M}_{ind} = \{\forall i \neq j, \phi^{i,j}_{ind}\}$ and $\mathcal{M}_{dep} = \{\forall i \neq j, \phi^{i,j}_{dep}\}$ where $\phi^{i,j}_{ind}$ and $\phi^{i,j}_{dep}$ are based on the probability functions $p(X_i, X_j|C)$ $p(C)$ and $p(X_i|C)p(X_j|C)p(C)$ respectively. The classifier $\phi^{i,j}_{dep}$ is taking into account the dependence between X_i and X_j while $\phi^{i,j}_{ind}$ considers that X and Y are independent given the class C. The difference of the classification error between each pair of model $\phi^{i,j}_{dep}$ and $\phi^{i,j}_{ind}$ tends to increase with the increase in the 3-way interaction information $I(X_i; X_j; C)$. This fact is well illustrated by artificial domains in Pérez et al. (2006a) (see Chapter 5 for further details). The 3-way interaction information has been recently used in Pernkopf and Bilmes (2005) for the induction of Bayesian classifiers [Bilmes (2000); Pernkopf and Bilmes (2005); Pérez et al. (2006a)]. For a detailed literature review in interaction information since 1954 see Section 2.5 of Jakulin and Bratko (2004).

1.11 Curse of dimensionality and feature subset selection

An important issue of the *preprocessing* of the data consists of transforming the variable space or selecting the subset of relevant variables which will take part in the classifier induction process. Reducing the dimensionality of the domain (number of features) gives some advantages in a classifier induction process: reduction of the search space, easy explanation capacity, improvement of the classification accuracy, and enhancement of the reliability of its estimation. *Feature subset selection* [Liu and Motoda (1998, 2008); Saeys et al. (2007)] consists of selecting an appropriate subset of predictor variables for supervised classification. On the other hand, the *transformation of the space of variables* tries to construct a set of new artificial variables with interesting properties for classification. For example, principal component analysis [Jolliffe (1986)] obtains independent (orthogonal) transformed features which captures most of the information of the original feature space by condensing the variance of the original feature space (under Gaussian assumption). As we noted previously, feature subset selection and transformation of the space are usually considered a part of the preprocessing step, but it can also be considered a part of the classifier induction algorithm (learning process) because the use of different feature inevitably imposes different models [Egmont-Peterson (2004)].

For practical supervised classification applications in many real-world problems we will have to deal with domains (or spaces) of high dimensionality, e.g. $\mathcal{O}(10^4)$ predictor variables in DNA microarray domains. This poses some serious challenges, inherited from density estimation field, and it is an im-

portant factor influencing the design of classifiers. Let us see three intuitions taken from Bishop (2006):

- If we divide a region of a n-dimensional space into regular cells, then the number of such cells grows exponentially with the dimensionality of the space, n. For example, if for $n = 1$, the number of cells is k, then for any $n \geq 1$ is k^n. The problem with an exponentially large number of cells is that we would need an exponentially large training set in order to ensure that the cells are not empty. This problem is closely related with data sparsity and the estimation of parameters in multinomial distributions (or probability tables) and Bayesian multinomial networks –see Chapter 2.
- Consider concentric spheres of two radiuses $r = \{1, 1 - a\}$ in a space of n dimensions, and ask what is the fraction of the volume that lies between both spheres with respect to the biggest sphere. The fraction tends to one as n increases even for small values of a. Thus, in spaces of high dimensionality, most of the volume of a sphere is concentrated in a thin shell near the surface. So, in spaces of high dimensionality, the density estimation techniques should be concentrated in obtaining estimators with good fitting in the tails of the probability density functions.
- In a domain with n variables the number of different pairs of variables is $\Theta(n^2)$. The different number of sets of o variables is $\Theta(n^o)$. Thus, the number of different statistics of order o to be computed is $\Theta(n^o)$.

Clearly some intuitions developed in spaces of low dimensionality are not useful for high dimensional spaces.

Fortunately, real-world data will often be confined to a region of the space having lower effective dimensionality, and in particular the directions over which important variations in the target variables occur may be too confined. In other words, real-world domains usually contain many irrelevant or redundant variables. In this dissertation we introduce the most popular approach to avoid the curse of dimensionality in supervised classification: feature subset selection.

Following, in Section 1.11.1, we introduce the feature subset selection problem. It is divided into three main parts. The section starts by introducing FSS problem, and filter and wrapper approaches. Then, due to the importance of relevance concept, Section 1.11.1.1 formally presents the concept of relevance for predictor random variables in supervised classification problems. Section 1.11.1.2 presents the two main feature subset selection problems with different goals: minimal-optimal and all relevant problems. Then, we present four representative feature subset selection algorithms: ranking of mutual information plus threshold (MIT), correlation based feature selection (CFS), Phy algorithm and wrapper approaches for Bayesian networks.

1.11.1 Feature subset selection

Feature subset selection (FSS) [John et al. (1994); Liu and Motoda (1998); Bell and Wang (2000); Nilsson et al. (2007)] is the process of reducing input data dimension by selecting a subset $\boldsymbol{X}_s$ of the predictor variables $\boldsymbol{X}$, where $\boldsymbol{X}_s \subseteq \boldsymbol{X}$. The most popular purpose of FSS in supervised classification consists of selecting a subset of features from the feature space which is good enough to predict the class of an unlabeled sample, ignoring its ability to describe the training set. By reducing dimensionality, FSS attempts to solve two important problems: facilitate learning (inducing) accurate classifiers, and discover the most interesting features, which may provide for better understanding of the problem itself [Guyon and Elisseeff (2003)].

With regard to how to evaluate the goodness (or quality) of a subset of features, the FSS methods fall into two broad categories: filter [Liu and Motoda (1998)] and wrapper approaches [Kohavi and John (1997)]. In the *filter approach* a good feature set is selected as a result of pre-processing based on properties of the data itself and independent of the learning algorithm [Bell and Wang (2000)]. In other words, the scores used in the filter approaches are based on intrinsic characteristics of the data [Liu and Motoda (1998)]. The advantages of filter approaches are usually related to the time complexity needed to make the selection and to the independence from the induction algorithm. A score based on correlation used to select variables in a filter manner is the *correlation based feature selection* score [Hall and Smith (1997); Yu and Liu (2004)] (see Paragraph B. in Section 1.11.1.3). More examples based on information theory are the approaches based on relevance concepts [Wang (1996); Wang et al. (1999)].

In the wrapper approach [John et al. (1994)], FSS is done with the help of the classifier induction algorithm selected for the problem. The FSS algorithm conducts the search for a good feature set using the selected learning algorithm itself as a part of the evaluation function [Bell and Wang (2000)]. Wrapper approaches use an estimation of the classification performance as the search score [Kohavi and John (1997)]. Thus, they depend on the specific classifier used to estimate the classification goodness. In comparison with filter approach, wrapper approach can be thought of as being based on a different notion of relevance: relevance to a classifier (see Section 1.11.1.1). However, Kohavi (1995a) shows that the optimal feature subset for classification obtained in this way must be from relevant feature subset as defined in the filter approach [John et al. (1994)].

Most of the FSS algorithms perform a search in the space of the possible subsets of features. This process is usually based on a heuristic search guided by a score (filter or wrapper). The search process depends on the score and search strategy used. Two kinds of search processes can be identified, depending on the process of constructing the solution. The solution can be constructed incrementally by adding, removing or merging subsets of features, or the complete solution can be given at each step, like in some population

based meta-heuristic search algorithms such as genetic algorithms [Goldberg (1989)] of estimation distribution algorithms [Larrañaga and Lozano (2002); Inza et al. (2000)]. For a review of different search strategies, see Kudo (2000).

Since FSS has a traditional close link with the notion of relevance [Kinja and Rendell (1992); John et al. (1994); Hall and Smith (1997); Bell and Wang (2000); Nilsson et al. (2007)] the following subsection is focused on the concept of relevance.

1.11.1.1 Relevance

This section gives some definitions of variable relevance [John et al. (1994); Wang (1996); Bell and Wang (2000); Nilsson et al. (2007)] for supervised classification (and regression) problems. These definitions will be useful in order to understand the variable reduction and selection techniques. The definitions of relevance considered in this section are rooted in the well-known concept of (probabilistic) conditional independence (See Section 1.3). There exists an alternative concept of relevance which is concerned with the relevance between instances of variables, but this is focused on sample reduction (condensation) rather than space dimensionality reduction. However, both relevance concepts are related and are complementary aspects of the relevance concept. For a unified framework of relevance of variables and relevance of instances see Wang (1996). From here on, we will focus on variable relevance concept with dimensionality reduction purposes for supervised classification.

The following definitions of relevance are based on the predictor random variables $\boldsymbol{X} = (X_1, ..., X_d)$ and $\boldsymbol{X}_{\neg i} = \boldsymbol{X} \setminus X_i$, and on the target random variable C. The nature of the random variables, i.e. continuous, discrete or mixed, is not specified because the definitions are independent from their nature. Thus, the same definitions can be used in a regression problem, i.e. when the variable C is continuous, and for mixed domains, i.e. when $\boldsymbol{X}$ is a multidimensional mixed random variable. Moreover, the relevance definitions are given for univariate random variables, but they can be generalized to sets of variables by replacing the variable X_i by the subset of variables (or multidimensional random variable) $\boldsymbol{X}_s \subseteq \boldsymbol{X}$.

Most of the methodological contributions of this document are based on probabilistic graphical models [Pearl (1988); Castillo et al. (1997)]. Probabilistic graphical models are conditional independence maps (see Chapter 2, Section 2.2.1 and 2.3) and, thus, we are more interested in conditional independence statements. As we will see in Chapter 2, conditional independence statements enable us to graphically factorize the generalized joint probability distribution for $\boldsymbol{X}$, resulting in a reduction in the number of parameters that are needed to completely specify this distribution. In probabilistic graphical models, irrelevance is identified with conditional independence, and relevance is defined as a negation of the irrelevance. On the other hand, we are more interested in relevance rather than in irrelevance for feature subset selection. In order to take into account both aspects, and following Pearl (1988) and

John et al. (1994), we introduce the concept of relevance based on conditional dependence. Moreover, we will quantify the relevance of one variable with respect to a target variable in terms of the strength of their conditional dependence. For alternative definitions and a rigorous analysis of relevance concepts see Wang (1996).

A feature X_i can be identified as relevant to a target concept, such us the class C, if its value varies systematically with the target.

Definition 1.22 (Relevance) *A feature X_i is relevant to C iff there exist some x_i and c such that $p(X_i = x_i) > 0$ and $p(C = c|X_i = x_i) \neq p(C = c)$.*

Under this definition, X_i is relevant to C if knowing its value can change the estimates for C, or in other words, if C is dependent on X_i, $CD(C; X_i|\emptyset)$. However, this definition is not useful for our purposes since it is stated in terms of (in)dependence and not in terms of conditional (in)dependence. Moreover, John et al. (1994) showed by an example (see XOR example in Section 1.11.1.2), that the above definition gives unexpected results, and that the dichotomy (relevance/irrelevance) in general is not enough. Then, they propose an alternative formulation of relevance, which distinguishes between strong relevance and weak relevance.

Two well-known relevance definitions coping with this problem were proposed by John et al. (1994).

Definition 1.23 (Strong relevance) *A feature X_i is strongly relevant to C iff $CD(C; X_i|\boldsymbol{X}_{\neg i})$.*

Definition 1.24 (Weak relevance) *A feature X_i is weak relevant to C iff it is not strongly relevant to C, but satisfies $CD(C; X_i|\boldsymbol{X}_s)$ for some $\boldsymbol{X}_s \subseteq \boldsymbol{X}_{\neg i}$.*

A *strongly relevant* feature X_i provides information about C that cannot be obtained from any other feature $\boldsymbol{X}_{\neg i}$. On the other hand, a *weakly relevant* feature X_i also provides information about C but this information is *redundant* –it can also be obtained from other features $\boldsymbol{X}_{\neg i}$. An illustrative example of strong and weak relevances is given in the following subsection (Section 1.11.1.2).

Definition 1.25 (Relevance) *A feature X_i is relevant to C iff it is strongly relevant or weakly relevant to C. A feature X_i is irrelevant to C iff it is not relevant to C.*

Considering Definition 1.25 [John et al. (1994)] an intuitive measure for the relevance of a feature X is $I(X; C)$ and for the relevance of a subset of variables $\boldsymbol{X}_s \in \boldsymbol{X}$ is the mutual information $I(\boldsymbol{X}_s; C)$. Unfortunately, the number of cases required for obtaining a reliable estimator of $I(\boldsymbol{X}_s; C)$ increases exponentially with the cardinality of the subset $\boldsymbol{X}_s$.

The following definition considers the feature relevance concept with respect to a classifier, $\phi_{\boldsymbol{X}}$ [Nilsson et al. (2007)].

Definition 1.26 *A feature X_i is relevant to a classifier $\phi_{\boldsymbol{X}}$ iff $p(\phi_{\boldsymbol{X}}(x_i, \boldsymbol{x}_{\neg i}) \neq \phi_{\boldsymbol{X}}(x'_i, \boldsymbol{x}_{\neg i})) > 0$ for some $\boldsymbol{X}_{\neg i} = \boldsymbol{x}_{\neg i}$, where x_i and x'_i are independent and identically distributed samples.*

This definition states that in order to be considered relevant to a classifier ϕ, a feature X_i must influence the value of $\phi(\boldsymbol{x})$ for some $\boldsymbol{x}$ with non-zero probability. The relevance for a classifier is closely related to the relevance for the class variable. The definition can be reformulated as follows for generative and conditional classifiers: a feature X_i is relevant to a classifier ϕ based on the distributions $p_\phi(c|\boldsymbol{x})$ iff $p(\arg_c \max p_\phi(c|x_i, \boldsymbol{x}_{\neg i}) \neq \arg_c \max p_\phi(c|x'_i, \boldsymbol{x}_{\neg i})) > 0$ for some $\boldsymbol{x}_{\neg i}$. Thus, in general, a variable can be strongly relevant for a class variable being at the same time not relevant for a classifier. Besides, one feature can be relevant for a classifier, being at the same time irrelevant for another. Moreover, a variable could be strongly relevant for the class variable, being at the same time irrelevant for Bayes classifier (for the definition of Bayes classifier see Section 1.5), that is, a value of X_i can affect the a posteriori $p(c|x_i, \boldsymbol{x}_{\neg i})$ for some c and any $\boldsymbol{x}_{\neg i}$ without effecting the Bayes classifier.

In practise, the data distribution is unknown, and a classifier ϕ must be induced from the training data $S = \{(\boldsymbol{x}^{(1)}, c^{(1)}), ..., (\boldsymbol{x}^{(N)}, c^{(N)})\}$ by an induction algorithm A, $A(S) = \phi$ (see Section 1). As we have seen before, the relevance can be defined with respect to some classifier, ϕ. It is useful to consider the relevance with respect to the Bayes classifier, ϕ_B, which is optimal for classification and independent from the training set. Unfortunately, this classifier is unknown in real-world domains, but there are some classifiers for which the consistency is proved, such as k-nearest neighbor for appropriate values of k [Devroye et al. (1996)] or Parzen window classifier (see Chapter 4). We say that an inducer is consistent if the induced classifier $A(S) = \phi$ converges in probability to Bayes classifier ϕ_B as the sample size tends to infinity. Thus we can use a consistent classifier $\phi = A(S)$ based on a finite sample set S as an estimate of the Bayes classifier and consider the relevance to ϕ as an estimate of the relevance to the Bayes classifier.

1.11.1.2 Minimal-optimal and all-relevant feature subset selection

The problem of feature subset selection can be approached from two different points of view [Nilsson et al. (2007)]: finding a subset of features optimal for classification and finding all the features relevant to the target variable. The latter problem is motivated by recent applications within bioinformatics, particularly gene expression analysis [Nilsson et al. (2007)]. However, most of the works in the literature are rather interested in obtaining good features for classification [John and Langley (1995); Hall (1999)].

The goal of minimal-optimal feature subset selection is to obtain an optimal classifier using the minimal number of features. We sill now consider two related minimal-optimal problem definition. The first one is based on the concept of relevance as given by Definition 1.25 and the latter is based on the idea of relevance to the Bayes classifier.

Let us consider one illustrative example taken from John et al. (1994) in order to understand the nature of the minimal-optimal FSS. Let predictors $X_1, ..., X_5$ be boolean, where $X_4 = \neg X_2$ and $X_5 = \neg X_3$, and let $g((x_1, ..., x_5)) = x_1 \bigoplus x_2 = x_1 \bigoplus x_4 = c$ be the function that determines the class, where $\bigoplus$ represents the XOR operator. Clearly, two minimal optimal subsets of features exist, $\{X_1, X_2\}$ and $\{X_1, X_4\}$. Given the definition of strong and weak relevance of Equations 1.23 and 1.24 we can say that X_1 is strongly relevant, X_2 and X_4 are weakly relevant and X_3 and X_5 irrelevant. Under these definitions of relevance, X_i is relevant to C if the probability of the outcome can change when we eliminate knowledge about the value of X_i. Strong relevance implies that the feature is indispensable [John and Langley (1995)], in the sense that it cannot be removed without loss of accuracy in the estimate $\hat{p}(c|\boldsymbol{x})$. On the other hand, weak relevance implies that the attribute can sometimes contribute to improve the accuracy of the estimated a posteriori probabilities $\hat{p}(c|\boldsymbol{x})$ [John and Langley (1995)]. Moreover, under some mild conditions related with the uniqueness of the Bayes classifier, Nilsson et al. (2007) demonstrates that the minimal-optimal set can be defined as the set of features strongly relevant to the Bayes classifier ϕ_B. Under these mild conditions Nilsson et al. (2007) proves that minimal-optimal subset only contains strongly relevant features in the sense of Definition 1.23. It must be highlighted that in certain domains a strongly relevant feature X_i could be not included in the set of relevant features to the Bayes classifier. In other words, the value x_i can affect the probability of the class $p(c|\boldsymbol{x})$ but not the Bayes classifier because the change in $p(c|\boldsymbol{x})$ is not large enough to alter the decision $\arg_c \max p(c|\boldsymbol{x})$. In this sense, relevance to Bayes classifier is stronger than strong relevance to the class variable. An illustrative example of this fact is given in Nilsson et al. (2007).

A recent study by Yu and Liu (2004) examines the role of weakly relevant features in more detail and subdivides these further into redundant and non-redundant weakly relevant features, of which the latter is deemed to be important for the Bayes classifier. However, Yu and Liu (2004) consider arbitrary distributions; for strictly positive distributions however, it can be seen that all weakly relevant features are also redundant in their terminology, so that their distinction is not useful in this case [Nilsson et al. (2007)].

On the other hand, the goal of all-relevant FSS is to obtain the entire set of features relevant to the class variable C. We can define the all-relevant subset in the light of the relevance given by Definition 1.25 [Nilsson et al. (2007)], but it can be also defined in terms of the goal of the study; for example, in bioinformatics many researchers are more interested in the biological significance of the features (genes) that depend on the class C rather than in obtaining a high discriminative subset of features. However, in practice, a minimal-optimal FSS is usually performed to optimize some classification criteria and then the chosen features are examined for biological significance [Golub et al. (1999); Guyon et al. (2002)]. Unfortunately, this strategy ignores the distinction between biological significance and prediction. A characteriza-

tion of the problem in the sense of Definition 1.25 is given in Nilsson et al. (2007). Moreover Nilsson et al. (2007) presents, under some mild conditions related with the uniqueness of the Bayes classifier, a consistent algorithm in polynomial time with the number of features. In the remainder of this section, we will focus on feature subset selection for the minimal-optimal problem.

1.11.1.3 Examples of FSS algorithms

Now we will present a representative set of feature selection algorithms, highlighting their main differences taking into account the heuristic, the dimensionality of the used statistics and their merits and problems.

A. Mutual information plus threshold

This algorithm is representative of the family of univariate algorithms. We call this family of FSS algorithm univariate because they measure the relevance of each individual predictor X_i for $i = 1, ..., n$ to the class variable C independently from the rest of predictors $\boldsymbol{X}_s \in \boldsymbol{X}_{\neg i} = (X_1, ..., X_{i-1}, X_{i+1}, ..., X_n)$. This algorithm quantifies the relevance of each variable X_i for $i = 1, ..., n$ in terms of its dependence with the class $CD(X_i; C|\emptyset)$ by means of the mutual information $I(X_i; C)$. Then, by means of the following (in)dependence test presented which determines if the relevance of each predictor variable is statistically significant or not.

This test is based on a theorem introduced in [Kullback (1959)] considering the discrete mutual information estimated using the empirical probability distribution obtained from $\mathcal{S}$, $\hat{I}(X_i; C)$. The test is based on the following theorem:

Theorem 5. *Let X_i and C be two discrete random variables which take r_i and r different values respectively. Let $\mathcal{S} = \{(x_i^{(1)}, c^{(1)}), ..., (x_i^{(N)}, c^{(N)})\}$ be a bivariate sample of iid cases for (X_i, C). Let $\hat{p}(x_i, c)$, $\hat{p}(x_i)$ and $\hat{p}(c)$ be the empirical probability distributions obtained from $\mathcal{S}$ for (X_i, C), X_i and C respectively, and let $\hat{I}(X_i; C)$ be the estimator of $I(X_i; C)$ based on $\hat{p}(x_i, c)$, $\hat{p}(x_i)$ and $\hat{p}(c)$, $\hat{I}(X_i; C) = \sum_{x_i} \sum_c \ln \hat{p}(x_i, c) \frac{\hat{p}(x_i, c)}{\hat{p}(x_i)\hat{p}(c)}$. If $p(x_i, c) = p(x_i)p(c)$, i.e. $CI(X_i; C|\emptyset)$, then*

$$2N \cdot \hat{I}(X_i; C) \underset{N \to \infty}{\overset{d}{\to}} \chi^2(t; \theta = (r_i - 1)(r - 1))$$

where d indicates convergence in distribution and θ are the degrees of freedom of the χ^2 distribution. ■

The proof of this theorem can be found in [Kullback (1959)]. An alternative proof in a more general form (φ-divergences [Ali and Silvey (1966)]) can be found in [Pardo et al. (1993)] (Theorem 7.4, p.307). Note that the logarithm used to compute the mutual information is the natural logarithm (base e).

Based on Theorem 5, the following (in)dependence test can be proposed [Kullback (1959); Pardo et al. (1993)]: Let X_i and C be two discrete random

variables, where $p(x_i, c)$ is their joint probability distribution and, $p(x_i)$ and $p(c)$ their marginal probability distributions respectively. We are interested in contrasting the null hypothesis H_0: *X_i and C are independent*, which can be stated as:

$$H_0 : p(x_i, c) = p(x_i)p(c) \text{ for all } (x_i, c)$$

being the alternative hypothesis $H_1 : p(x_i, c) \neq p(x_i)p(c)$ for some (x_i, c).

Then, the following statistic is considered:

$$t = 2N \cdot \hat{I}(X_i; C)$$

Note that, by Theorem 5, we have that $T \rightsquigarrow \chi^2(t; (r_i - 1)(r - 1))$. If the null hypothesis is true, t should be small. Then, a high t indicates a low compatibility of the null hypothesis with the data. So, by Theorem 5, the statistical test of size α consists of rejecting H_0 if $t > \tau$, where $\alpha = pval(\tau) = \int_\tau^\infty \chi^2(t; (r_i - 1)(r - 1))dt$.

One of the main advantages of the family of univariate FSS algorithms based on relevance is that they only require to compute a number of statistics $\mathcal{O}(n)$, e.g. $\hat{I}(X_i; C)$ for all i. The univariate FSS algorithms are not suitable for minimal-optimal problems because they consider the relevance individually for each predictor independently from the rest of predictors and, thus, they tend to obtain features highly correlated with the class but they could share a lot of redundant information. On the other hand, univariate FSS algorithms can be useful to provide a first approach to the all-relevant problem selecting the relevant variables X_i which verify $CD(X_i; C|\emptyset)$, but they do not not take into account variables given by $CD(X_i; C|\boldsymbol{X}_s)$ with $\boldsymbol{X}_s \in \boldsymbol{X}_{\neg i}$.

B. Correlation-based feature selection

This paragraph presents an score rather than a complete FSS algorithm. This score is called correlation based feature selection (CFS) [Hall (1999)] and it is based on the following heuristic: *a good set for supervised classification includes predictor variables X_i highly correlated with the class C being, at the same time, lowly correlated between them.* The maximization of the score can be done by means of different search strategies and the most popular is a forward greedy approach. Given a set of features $\boldsymbol{X} = (X_1, ..., X_n)$, its CFS score is given by:

$$CFS(\boldsymbol{X}) = \frac{n\overline{cor}_C}{\sqrt{n + n(n-1)\overline{cor}_{\boldsymbol{X}}}} = \frac{\sum_{i=1}^n cor(X_i; C)}{\sqrt{n + 2\sum_{i=1}^n \sum_{j=i+1}^n cor(X_i; X_j)}} \tag{1.60}$$

where $\overline{cor}_C = \frac{1}{n}\sum_{i=1}^n cor(X_i; C)$ is the average correlation with the class and $\overline{cor}_{\boldsymbol{X}} = \frac{2}{n(n-1)}\sum_{i=1}^n \sum_{j=i+1}^n cor(X_i; X_j)$ is the average correlation between the predictors. We can conclude directly from Equation 1.60 that the higher the correlations between the features and the class, $\sum_{i=1}^n cor(X_i; C)$, the higher the CFS score. Moreover, the lower the correlations among the

features, $\sum_{i=1}^{n}\sum_{j=i+1}^{n} cor(X_i; X_j)$, the higher the CFS score. However, it is unlikely that a group of features that are all highly correlated with the class variable will at the same time bear low correlations among them. Furthermore, when a new feature is added to the selected subset, low inter correlation with the already selected features may well predominate over high correlation with the class [Hall (1999)].

In Hall (1999) different correlation measures are proposed and tested in order to obtain different implementations of the CFS score: *symmetrical uncertainty* (SU), minimum description length and relief. The three correlation scores are appropriate for CFS because (i) they are symmetrical $cor(X_i; X_j) = cor(X_j; X_i)$, (ii) they prefer predictive features with fewer values over those with more values. Now we will focus on CFS based on symmetrical uncertainty correlation score. It seems to perform slightly better than minimum description length score, specially with small training set sizes, and it performs better than relief [Hall (1999)]. The symmetrical uncertainty correlation coefficient (SU) between X and Y discrete random variables is defined as:

$$su(X; Y) = \frac{2I(X; Y)}{H(X) + H(Y)} \tag{1.61}$$

SU is a symmetric and normalized quantity based on mutual information $I(X; Y)$ and entropy $H(X), H(Y), 0 \leq su(X; Y) = su(Y; X) \leq 1$. The mutual information can be used to measure the correlation between variables X and Y in terms of the level of information shared by them (see Section 1.10 for more details). SU inherits the symmetry from $I(X; Y)$, which is desirable to quantify the correlation among the predictors. On the other hand, the entropy is used in order to normalize the mutual information. Mutual information can be problematic because it tends to give higher values to variables with more states. Entropy also tends to increase with the number of values of the discrete random variable, and it gives a high bound to the mutual information $0 \leq I(X; Y) \leq \min(H(X), H(Y))$. Thus, it can be considered comparable to the mutual information. So, SU compensates the bias of mutual information towards attributes with many values using the entropy. Due to normalization, the obtained individual correlation coefficients can be considered comparable.

Experimentation on artificial domains shows CFS to be effective at screening both irrelevant (low correlated with the class) and redundant features (high correlated with the included features) and, as long as there are no extreme feature interactions, CFS is able to quickly identify relevant features. Using CFS the accuracy can be dramatically improved in artificial domains with the presence of irrelevant or redundant variables [Hall (1999)].

But CFS score is too heavily biased in favor of small feature subsets and this aggressive bias may result in some loss of accuracy. Because correlations are estimated globally using the training set, CFS with forward greedy selection tends to select a core subset of features, but may fail to include subsidiary features that are locally predictive in a small area of the space of instances.

C. Phi

Phi is a consistent FSS algorithm which solves minimal-optimal problem under some mild assumptions related to the uniqueness of the Bayes classifier. The consistency of the algorithm is proved by Nilsson et al. (2007). The pseudo-code of the Phi backward algorithm is given in Figure 1

Algorithm 1: **Phi feature subset selection algorithm.**

1 Estimate the error of the classifier with all predictor variables, $\hat{\epsilon}_{\boldsymbol{X}}$
2 Estimate the errors of the classifiers with the predictors $\boldsymbol{X}_{\neg i}$ for $i = 1, ..., n$, $\hat{\epsilon}_{\boldsymbol{X}_{\neg i}}$
3 Select the random variables X_i for which $\hat{\epsilon}_{\boldsymbol{X}_{\neg i}} > \hat{\epsilon}_{\boldsymbol{X}}$ for $i = 1, ..., n$

The error can be estimated using the estimators introduced in 1.8.1. This algorithm can be implemented for different choices of classifier induction algorithms, provided that they are consistent. For example, it can be implemented with k-nearest neighbor with an appropriate strategy for fixing k value: $k = \log N$ or $k = \sqrt{N}$ [Devroye et al. (1996)], where N is the number of cases in the training set. An implementation of Phi algorithm with support vector machines was proposed in Guyon et al. (2002) and it was applied to gene selection for cancer classification.

D. Wrapper feature subset selection using Bayesian networks

In the literature several classifier induction algorithms have been proposed based on Bayesian networks which perform a feature subset selection implicitly in the learning process. Some of these methods are wrapper [Kohavi (1995b); Langley and Sage (1994); Pazzani (1997); Keogh and Pazzani (1999)] and they relate the relevance of a feature X_i with respect to the class variable C with the improvement of the performance when the feature is X_i included in the Markov blanket of C. The inclusion in the Markov blanket of C can be performed in different ways: by considering the addition of an arc from C to X_i, $C \rightarrow X_i$, such as in selective naïve Bayes [Langley and Sage (1994)] and in wrapper tree-augmented naïve Bayes [Keogh and Pazzani (1999)], by considering the addition of the arc $X_i \rightarrow C$ [Cheng and Greiner (1999)], or by considering to join the feature X_i with a variable X_j relevant to C [Pazzani (1997)], among others. Some of the mentioned approaches are presented in detail in Chapter 2, Section 2.5.3. Moreover, some of the mentioned algorithms are adapted to conditional Gaussian networks in Chapter 3.

1.12 Discretization

In this section we briefly explain the discretization process of a continuous variable.

Often data sets are described by continuous variables. If the number of continuous variables is large, model building for such data can be difficult and/or highly inefficient. Moreover, many data mining algorithms can only handle discrete attributes.

The goal of discretization is to reduce the number of values a continuous variable assumes by grouping them into a number, r, of intervals or bins. A discretization algorithm, used as a preprocessing step, should generate as few discrete intervals as possible. On the other hand, too few intervals may hide information about the relationship between the class variable and the discretized variable. Thus, two key problems associated with discretization are how to choose the number of intervals, r, and how to decide what the intervals are. In summary there is a trade off between the loss of information and the number of intervals of the discretized variable.

Discretization can be performed with or without class information. Analogously to unsupervised and supervised learning methods, discretization algorithms can be divided into two main categories: unsupervised and supervised. If class information exists, a discretization algorithm should take advantage of it, especially if obtained data is used to learn a classifier. In this case, a discretization algorithm should maximize the dependence between the variable X and the class C. An additional benefit of using supervised discretization is that it tends to minimize the loss of information of the class variable.

Discretization of continuous variables is most often performed one attribute at a time, independent of other attributes. This approach is known as univariate or static attribute discretization. On the other hand, when all continuous variables are discretized simultaneously, taking into account the interdependencies among them, the approach is known as multivariate or dynamic attribute discretization. For a more general and complete taxonomy of the discretization algorithms the reader should consult [Yang (2003)].

Following, we present three univariate classic and popular discretization algorithms: equal width, equal frequency and entropy [Fayyad and Irani (1993)].

1.12.1 Equal width

Equal width algorithm is a non-supervised univariate discretization algorithm. It is considered an intuitive discretization because it divides the range of X in the data set $\mathcal{S} = \{x^{(1)}, ..., x^{(N)}\}$, $[x^{min}, ..., x^{max}]$, into r partitions of equal width:

$$P^i = [x^{min} + (i-1)w, x^{min} + iw) \tag{1.62}$$

for $i = 2, ..., r-1$ where $w = (x^{max} - x^{min})/r$, the first partition is given by $P^1 = (-\infty, x^{min} + w)$ and the last partition $P^r = [x^{min} + (r-1)w, \infty)$.

1.12.2 Equal frequency

Equal frequency algorithm is a non-supervised univariate discretization algorithm. It divides the range of X into r partitions $[-\infty, x^1), [x^1, x^2), ..., [x^{r-2},$

$x^{r-1}), [x^{r-1}, \infty)$ where x^i is the i-th cut point. This algorithm chooses the cut points x^i for $i = 1, ..., r-1$ such that the same number of instances falls into each partition. That is, $N_i \simeq N/r$, where N_i is the number of instances of $\mathcal{S}$ falling in the i-th.

1.12.3 Entropy

The method proposed by Fayyad and Irani (1993), referred to as entropy discretization algorithm in this section, is based on a minimal entropy heuristic. It is a supervised univariate discretization algorithm. This method uses the class information entropy of candidate partitions to select bin boundaries for discretization. If we are given a data set $\mathcal{S} = \{(x^{(1)}, c^{(1)}), ..., (x^{(N)}, c^{(N)})\}$, and a partition boundary t. Then t partitions the set $\mathcal{S}$ of instances into the subsets $\mathcal{S}_1$ and $\mathcal{S}_2$ of sizes N_1 and N_2 respectively. Let there be r_C classes, $c \in \{c^1, ..., c^r\}$, and let $Pr(c^i|\mathcal{S})$ the proportion of examples in $\mathcal{S}$ that have the class c^i. The class entropy of a subset $\mathcal{S}$ is defined as:

$$H(C|\mathcal{S}) = -\sum_{i=1}^{r_C} Pr(c^i|\mathcal{S}) \log Pr(c^i|\mathcal{S}) \tag{1.63}$$

Then the class information entropy of the partition induced by t is given by

$$H(C|\mathcal{S}, t) = \frac{N_1}{N} H(C|\mathcal{S}_1) + \frac{N_2}{N} H(C|\mathcal{S}_2) \tag{1.64}$$

For a given variable, X, the boundary t_{min} which minimizes Equation 1.64 over all possible partitions, a boundary is selected as a binary discretization boundary. This method can then be applied recursively to both of the partitions induced by t_{min} until some stopping condition is achieved, thus creating multiple intervals on the feature X.

Entropy discretization make use of the *Minimal Description Length Principle* to determine a stopping criterion for their recursive discretization algorithm. Recursive partition within a data set $\mathcal{S}$ stops when

$$Gain(X, t|\mathcal{S}) < \frac{\log_2 N - 1}{N} + \frac{\Delta(X, t|\mathcal{S})}{N} \tag{1.65}$$

where,

$$Gain(X, t|\mathcal{S}) = H(C|\mathcal{S}) - H(C|\mathcal{S}, t)$$
$$\Delta(X, t|\mathcal{S}) = \log_2(3^{r_C} - 2) - [r_C H(C|\mathcal{S}) - r_C^1 H(C|\mathcal{S}_1) - r_C^2 H(C|\mathcal{S}_2)]$$

and r_C^i is the number of class labels represented in the set $\mathcal{S}_i$. Since the partitions along each branch of the recursive discretization are evaluated independently using this criteria, some areas in the continuous range of X will be partitioned very finely, whereas others will be partitioned coarsely.

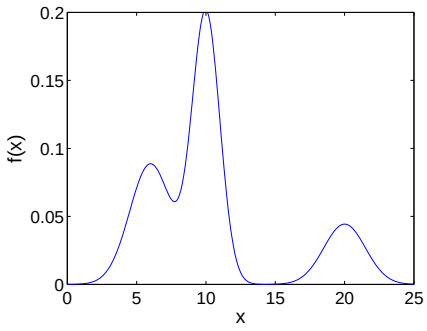


Fig. 1.5. This figure shows a finite Gaussian mixture density, $f(\boldsymbol{x})$ This density is sampled to create three data sets, $\mathcal{S}_{10}$, $\mathcal{S}_{100}$ and $\mathcal{S}_{1000}$ with 10, 100 and 1000 instances respectively, where $\mathcal{S}_{10} \subset \mathcal{S}_{100} \subset \mathcal{S}_{1000}$.

1.13 Non-parametric density estimation

The section starts with a brief introduction to non-parametric density estimation focused on unidimensional continuous random variables, from the simplest histograms to variable kernel density estimation technique. This section has been adapted from [Silverman (1986)].

The estimated density is denoted as $\hat{f}_X(x; S)$, or simply as $\hat{f}(x)$ when it is clear from the context, where $\mathcal{S}$ is a set of idd samples (training set) of the univariate continuous random variable X, $\mathcal{S} = \{x^{(1)}, ..., x^{(N)}\}$. Figure shows an example of an arbitrary density function for X from the family of finite Gaussian mixture densities [Titterington et al. (1985)], $X \rightsquigarrow 2/6\mathcal{N}(x; \mu = 5, \sigma = 1.5) + 3/6\mathcal{N}(x; 6, 1) + 1/6\mathcal{N}(x; 10, 1.5)$.

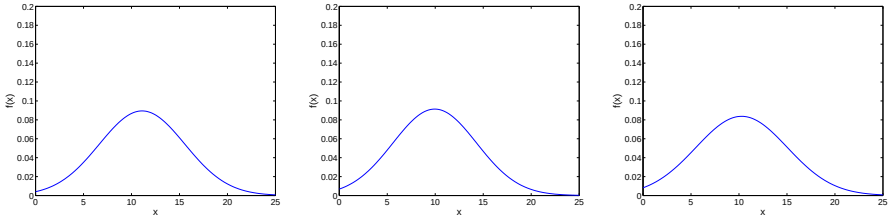
Often, parametric densities, such as Gaussian density function (see Section 3.2 for further details), are not able to properly represent the density function of a continuous random variable X, $f(X)$. In other words, the error of a parametric density with respect to an arbitrary density function can be too high due to restrictive function shapes. Intuitively, non-parametric density estimators try to break with the restrictive function shapes imposed by the family of parametric densities.

1.13.1 Histograms

The oldest and most popular density estimator is the histogram. Given an origin x^0 and a bin width h, we define the bins (or cells) of the histogram to be the intervals $[x^0 + ih, x^0(i + 1)h)$ for all $i \in \mathbb{N}$. The intervals have been chosen closed on the left and open on the right for definiteness.

The histogram is then defined by

$$\hat{f}(x) == N_i/N \tag{1.66}$$



(a) $\hat{f}(x;\mathcal{S}_{10})$, $\epsilon = 0.78$. (b) $\hat{f}(x;\mathcal{S}_{100})$, $\epsilon = 0.70$. (c) $\hat{f}(x;\mathcal{S}_{1000})$, $\epsilon = 0.73$.

Fig. 1.6. These figures show the densities modeled under Gaussian assumption with maximum likelihood parameters based on the data sets $\mathcal{S}_{10}$, $\mathcal{S}_{100}$ and $\mathcal{S}_{1000}$. The absolute error of the estimated density, ϵ, is shown in each figure.

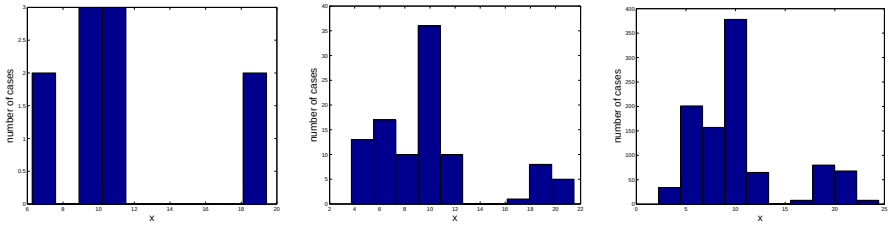
where x is in the i-th bin, $x \in [x^0 + ih, x^0 + (i+1)h)$, and N_i is the number of samples $x' \in \mathcal{S}$ in the same bin as x, $N_i = |\mathcal{S}_i| = |\{x'/x' \in [x^0 + ih, x^0 + (i+1)h)\}|$. Therefore, in order to construct the histogram, we have to specify both an origin x_0 and a bin width h. Note that, histogram is equivalent to first discretize the random variable X using the equal width procedure and, then, estimate the multinomial probability distribution (probability table or contingency table) of the obtained discretized variable.

The histogram can be generalized by allowing the bin widths to vary. So we have $\boldsymbol{h} = (h_0, ..., h_{r-1})$ one for each bin (or cell) i. Then the estimate is analogously given by

$$\hat{f}_h(x) = N_i/N \tag{1.67}$$

where x is in the i-th bin, and N_i is the number of samples $x' \in \mathcal{S}$ in the same bin as x, $N_i = |\mathcal{S}_i| = |\{x'/x' \in [x^0 + \sum_{j=1}^{i} h_j, x^0 + \sum_{j=1}^{i+1} h_j)\}|$. This approach is also equivalent to, firstly discretizing the random variable (by using an appropriate discretization algorithm) and, then, estimating the multinomial probability distribution of the obtained discretized variable.

Those who are sceptical about density estimation often ask why it ever is necessary to use methods more sophisticated than the simple histogram. The case for such methods and the drawbacks of the histograms depend quite substantially on the context, i.e. there is not a best density estimation technique suitable for all domains in general. In terms of various mathematical descriptions of accuracy in density estimation, the histogram can be quite substantially improved upon, and this mathematical drawback translates itself into inefficient use of the data if histograms are used as density estimates in procedures such as cluster analysis and nonparametric discriminant analysis. The discontinuities of histograms causes extreme difficulty if derivatives of the estimates are required. When density estimates are needed as intermediate components of other methods, e.g. classifiers based on kernel based Bayesian networks, the case for using alternatives to histograms is quite strong (see Silverman (1986)).



(a) Histogram based on $\mathcal{S}_{10}$. (b) Histogram based on $\mathcal{S}_{100}$. (c) Histogram based on $\mathcal{S}_{1000}$.

Fig. 1.7. These figures show histograms based on the data sets $\mathcal{S}_{10}$, $\mathcal{S}_{100}$ and $\mathcal{S}_{1000}$. The obtained histogram clearly improves as the size N of the data set $\mathcal{S}_N$ increases.

For the presentation and exploration of the data, histograms are an extremely useful class of density estimates, particularly in the univariate case. However, even in one dimension, the choice of the origin can have quite an effect, e.g. the occurrence or absence of peaks in the probability distribution obtained depends on the choice of the origin. Finally, it should be stressed that, in all cases, the histogram requires a choice of the amount of smoothing, i.e. r bin widths $\boldsymbol{h} = (h_0, ..., h_{r-1})$ for the case of histograms with variable bin widths.

1.13.2 Naive estimator

From the definition of probability density function 1.8 of the random variable X, we have that

$$f(x) = \lim_{h \to 0} \frac{1}{2h} Pr(x' \in (x - h, x + h)) \tag{1.68}$$

For any given h, we can of course estimate $Pr(x' \in (x - h, x + h))$ by the proportion of the sample falling in the interval $(x - h, x + h)$. Following this intuition, a natural estimator of the density is the naïve estimator which is given by

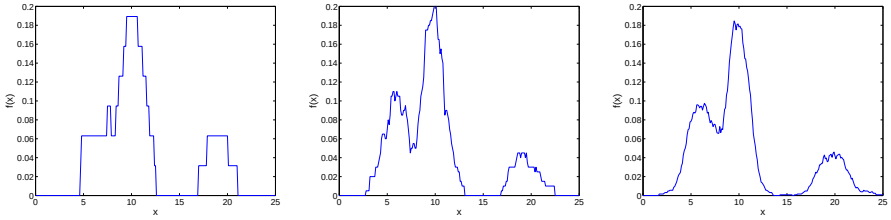
$$\hat{f}_n(x; \mathcal{S}_N) = \frac{1}{2h} N_x / N \tag{1.69}$$

where N_x is the number of cases falling in the interval $(x - h, x + h)$, $N_x = |\{x'/x' \in \mathcal{S} \wedge x' \in (x - h, x + h)\}|$. Note that it is advisable to select a small number h. In order to express the estimator more transparently, we define the weight function w

$$w(x) = \begin{cases} 1/2 & \text{if } |x| < 1 \\ 0 & \text{otherwise} \end{cases} \tag{1.70}$$

Then it is easy to see that the naïve estimator can be rewritten as

$$\hat{f}_n(x; h, \mathcal{S}_N) = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{h} w(\frac{x - x^{(i)}}{h}) \tag{1.71}$$



(a) $\hat{f}_n(x; h = 1.58, \mathcal{S}_{10})$, $\epsilon = 0.35$.

(b) $\hat{f}_n(x; h = 1.00, \mathcal{S}_{100})$, $\epsilon = 0.19$.

(c) $\hat{f}_n(x; h = 0.63, \mathcal{S}_{1000})$, $\epsilon = 0.10$.

Fig. 1.8. These figures show the densities modeled using naïve estimator based on the data sets $\mathcal{S}_{10}$, $\mathcal{S}_{100}$ and $\mathcal{S}_{1000}$. In each estimated density we have select the parameter h which minimizes the absolute error. The value of the parameter h and the absolute error of the estimated densities, ϵ, are shown in each figure. The densities modeled with naïve estimator clearly improve the estimations under Gaussian assumption shown in Figure 1.6. The modeled density clearly improves as the size of the data set increases.

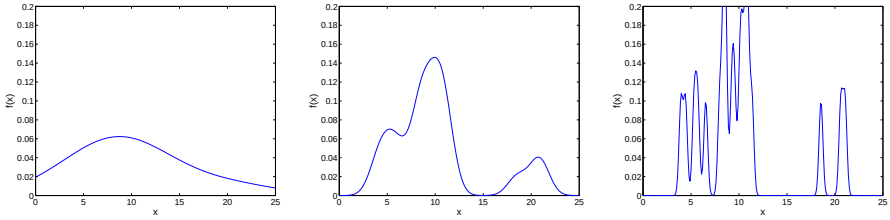
It follows from Equation 1.70 that the estimate is constructed by placing a box of width $2h$ and height $(2nh)^{-1}$ on each observation and then summing them to obtain the estimate. We shall return to this interpretation below, but it is instructive first to consider a connection with histograms.

Let us consider the histogram constructed from the data using bins of size (or width) $2h$ assuming that no observation of $\mathcal{S}_N$ lies exactly at the edge of a bin. If x happens to be at the center of one of the histogram bins, it follows at once from Equation 1.70 that the naïve estimate $\hat{f}_n(x; h, \mathcal{S}_N)$ will be exactly the ordinate of the histogram at x, $\hat{f}_h(x; h, \mathcal{S}_N)$. Therefore, the naïve estimator can be seen to be an attempt to construct a histogram where every point is the center of a sampling interval, and thus, freeing the histogram from a particular choice of bin position. But, the choice of bin width still remains and is governed by the parameter h, which controls the amount by which the data sample is smoothed to produce the estimate. Examples of naïve density estimations are shown in Figure 1.8.

The naïve estimator is not wholly satisfactory from the point of view of using density estimates for presentation. It follows from Equation 1.68 that $\hat{f}_n(x; h, \mathcal{S}_N)$ is not a continuous function. It has jumps at the points $x^{(i)} \pm h$ (see Figure 1.8). In order to overcome this difficulty, it is of interest to consider kernel density estimation, which can be considered a generalization of naïve estimator.

1.13.3 Kernel density estimation

It is easy to generalize the naïve estimator to overcome some of the difficulties discussed above. Replace the weight function $w(x)$ (see Equation 1.70) by a kernel function $K(x)$ which satisfies the condition



(a) Oversmooth: $\hat{f}_k(x; h = 5.0, \mathcal{S}_{20})$, $\epsilon = 0.73$.

(b) Optimum: $\hat{f}_k(x; h = 1.0, \mathcal{S}_{20})$, $\epsilon = 0.25$.

(c) Undersmooth: $\hat{f}_k(x; h = 0.2, \mathcal{S}_{20})$, $\epsilon = 0.62$.

Fig. 1.9. These figures show the densities modeled using kernel density estimation based on a data set with 20 samples, $\mathcal{S}_{20}$, and using different values for the window width, h. Figures 1.9(a), 1.9(b) and 1.9(c) show oversmooth, optimum and undersmooth density estimations based on $\mathcal{S}_{20}$, respectively.

$$\int_{-\infty}^{\infty} K(x)dx = 1$$

Usually, $K(x)$ will be symmetric probability density function, e.g. Gaussian density function (see Definition 3.1).

By analogy with naïve estimator (see Equation 1.68), the kernel estimator with the kernel function $K(x)$ is defined as

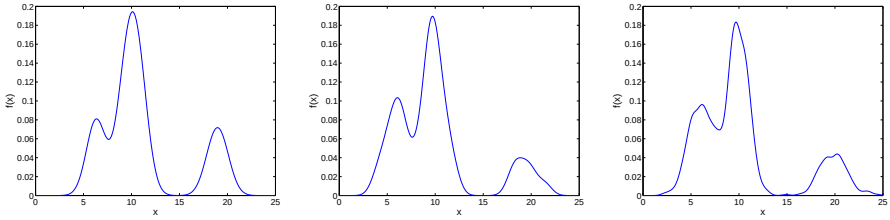
$$\hat{f}_k(x; K, h, \mathcal{S}_N) = \frac{1}{N} \sum_{i=1}^{N} K(\frac{x - x^{(i)}}{h}) \tag{1.72}$$

where h is the smoothing parameter (window width or bandwidth).

Just as the naïve estimator can be considered as a sum of "boxes" centered at the observations, the kernel estimator is a sum of "bumps" placed at the observations. The kernel function $K(x)$ determines the shape of the "bumps" while the window width h determines their width.

The effect of varying the window width is illustrated in Figure 1.9. A Gaussian kernel has been used to construct the density estimation. The estimation has been constructed from 20 idd samples, $\mathcal{S}_{20}$, taken from the density shown in Figure 1.5. If h is chosen too large then the shape of the distribution is obscured (see Figure 1.9(a)), while if h is chosen too small then spurious fine structure becomes visible (see Figure 1.9(c)). So, in the limit, as h tends to zero kernel estimator is (in a sense) a sum of Dirac delta function spikes at the observations. On the other hand, while as h becomes large kernel estimator is a sum of uniform delta function in all x and, thus, all detail, spurious or otherwise, is obscured. Figure 1.9(b) shows the optimum estimation based on $\mathcal{S}_{20}$.

Some of the elementary properties of kernel estimates follow at once from Equation 1.72. Provided the kernel $K(x)$ is a probability density function, i.e. everywhere non-negative and integrates to one, it will follow at once



(a) $\hat{f}_k(x; h = 1.0, \mathcal{S}_{10})$, $\epsilon = 0.31$. (b) $\hat{f}_k(x; h = 0.7, \mathcal{S}_{100})$, $\epsilon = 0.16$. (c) $\hat{f}_k(x; h = 0.4, \mathcal{S}_{1000})$, $\epsilon = 0.09$.

Fig. 1.10. These figures show the densities modeled using kernel density estimation using the data sets $\mathcal{S}_{10}$, $\mathcal{S}_{100}$ and $\mathcal{S}_{1000}$. In each estimated density we have select the parameter h which minimizes the absolute error. The value of the parameter h and the absolute error of the estimated densities, ϵ, are shown in each figure. The densities modeled with kernel density estimator clearly improve the estimations under Gaussian assumption shown in Figure 1.6. The modeled density clearly improves as the size of the data set increases. Note that the optimum window width, h, decreases as the size of the data set increases.

from Equation 1.72 that $\hat{f}_k(x; K, h, \mathcal{S}_N)$ is a density function. Furthermore, $\hat{f}_k(x; K, h, \mathcal{S}_N)$ will inherit all the continuity and differentiability properties of the kernel $K(x)$. For example if $K(x)$ is the Gaussian density function, then $\hat{f}(x; K, h, \mathcal{S}_N)$ will be a smooth density function having derivatives of all orders. Examples of densities modeled using kernel density estimator are given in Figure 1.10.

Apart from the histogram, the kernel estimator is probably the most popular and is certainly the most studied mathematically. It does, however, suffer from a slight drawback when applied to data from-long tailed distributions. Because the window width is fixed for all $x \in \mathcal{S}$, there is a tendency for spurious noise to appear in regions which are large but with low probability, e.g. tails of long tailed distributions. If the estimates are smoothed sufficiently to deal with this, then essential detail in the main part of the distribution is masked. In order to deal with this difficulty, the next section presents a kernel estimator with a variable window width, which can have advantages with respect to fixed kernel estimator by considering the local density of samples, i.e. a function of the points in $\mathcal{S}$ which are in the neighborhood of each $x \in \mathcal{S}$.

1.13.4 *k*-nearest neighbor estimator

The nearest neighbor class of estimators represents an attempt to adapt the amount of smoothing to the "local" density of data. The degree of smoothing is controlled by an integer k, chosen to be considerably smaller that the sample size, $|\mathcal{S}_N| = N$, e.g. $k = \log N$ or $k = \sqrt{N}$ [Devroye et al. (1996)].

The k-th nearest neighbor estimator depends on the distance measure selected between two points $d(x, x')$, e.g. Euclidean distance $|x - x'|$, and for

any x define

$$d_1(x) \leq d_2(x) \leq ... \leq d_N(x) \tag{1.73}$$

to be the distances, shorted in ascending order, from x to the points of the sample, $x' \in \mathcal{S}$. From here on we assume that the selected distance measure is Euclidean.

The k-th nearest neighbor estimator is given by

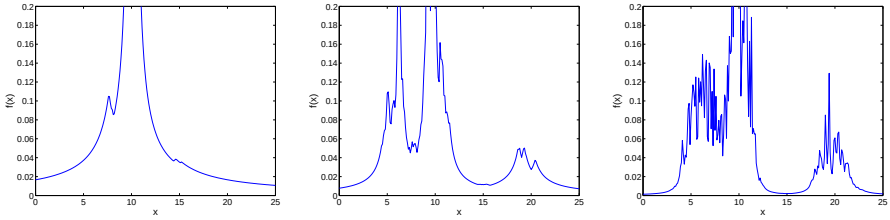
$$\hat{f}_{NN}(x; k, \mathcal{S}_N) = \frac{k}{2Nd_k(x)} \tag{1.74}$$

In order to intuitively understand this definition, suppose that the density at x is $f(x)$. Then, of a sample of size N, one would expect about $2rNf(x)$ observations to fall in the interval $[x - r, ..., x + r]$ for each $r > 0$. Since, by definition, exactly k observations fall in the interval $[x - d_k(x), x + d_k(x)]$, an estimate of the density at x may be obtained by putting $k = 2d_k(x)N\hat{f}(x)$ and this can be rearranged to give the definition of the k-th nearest neighbor estimate.

While the naïve estimator is based on the number of observations falling in a box of fixed width centered at the point of interest, the nearest neighbor is inversely proportional to the size of the box needed to contain a given number of observations. Thus, in the tail of the distribution, the distance $d_k(x)$ will be larger than in the main part of the distribution, i.e. the part with the highest density function values, and so the problem of undersmoothing in the tails should be reduced.

Like the naïve estimator, to which it is related, the nearest neighbor estimate as defined in Equation 1.74 is not a smooth curve. The function $d_k(x)$ can easily be seen to be continuous, but its derivative will have a discontinuity at every point of the form $1/2(x^{(i)} + x^{(i+k)})$, where $x^{(i)} \in \mathcal{S}$ and $\mathcal{S}$ is ordered in ascending order. It follows at once from these remarks and from the definition in Equation 1.74 that $\hat{f}_{NN}(x; k, \mathcal{S}_N)$ will have discontinuous derivative at all the same points as $d_k(x)$ for all x. In contrast to the kernel estimate, the nearest neighbor estimate will not itself be a probability density, since it will not integrate to unity. For x less than smallest data point $x < x^{(1)}$, we will have $d_k(x) = d(x, x^{(k)}) = |x^{(k)} - x|$ and for $x > x^{(N)}$ we will have $d_k(x) = d(x^{(k)}, x) = |x - x^{(k)}|$. Substituting into 1.74, it follows that $\int_{-\infty}^{\infty} \hat{f}_{NN}(x; k, \mathcal{S}_N)dx$ is infinite and that the tails of $\hat{f}_k(x; k, \mathcal{S}_N)$ die away at rate x^{-1}, in other words extremely slowly. Thus, the nearest neighbor estimate is unlikely to be appropriate if an estimate of the entire density is required. Examples of densities modeled using k-nearest neighbor estimator are given in Figure 1.11. The heavy tails and the discontinuities in the derivative are clear.

It is possible to generalize the intuition of the k-th nearest neighbor estimate to provide a kernel estimator which takes into account the local density of the sampled data $\mathcal{S}$. As in Section 1.13.3, let $K(x)$ be a kernel function



(a) $\hat{f}_{NN}(x; k = 3, \mathcal{S}_{10})$, $\epsilon = 0.77$. (b) $\hat{f}_k(x; k = 8, \mathcal{S}_{100})$, $\epsilon = 0.44$. (c) $\hat{f}_k(x; k = 10, \mathcal{S}_{1000})$, $\epsilon = 0.32$.

Fig. 1.11. These figures show the densities modeled using k-nearest neighbor estimator using the data sets $\mathcal{S}_{10}$, $\mathcal{S}_{100}$ and $\mathcal{S}_{1000}$. In each estimated density we have select the parameter k which minimizes the absolute error. The value of the parameter k and the absolute error of the estimated functions, ϵ, are shown in each figure. The densities modeled with k-nearest neighbor estimator for $N = \{100, 1000\}$ clearly improve the estimations under Gaussian assumption shown in Figure 1.6. The modeled density clearly improves as the size of the data set increases. Note that the optimum k increases as the size of the data set increases. The heavy tails and the discontinuities in the derivative are clear

integrating to one. Then the generalized k-th nearest neighbor estimate is defined by

$$\hat{f}_{k,K}(x) = \frac{1}{N d_k(x)} \sum_{i=1}^{N} K(\frac{x - x^{(i)}}{d_k(x)}) \tag{1.75}$$

It can be seen at once that $\hat{f}_{k,K}(x)$ is precisely the kernel estimate evaluated at x with window width $h = d_k(x)$. Thus, the overall amount of smoothing is governed by the choice of the integer k, but the window width used at any particular point depends on the density of observations near that point.

The ordinary k-th nearest neighbor estimate, based on Euclidean distance, is the special case of Equation 1.75 when K is the uniform kernel $K(x) = w(x)$ of Equation 1.68. Thus the definition given by Equation 1.75 is related in the same way to Equation 1.74 as the kernel estimator does to the naïve estimator. However, the derivatives of the generalized nearest neighbor estimate will be discontinuous at all the points where the function $d_k(x)$ has discontinuous derivative.

1.13.5 Variable kernel estimator

The variable kernel estimator is somewhat related to the nearest neighbor approach and is another method which adapts the amount of smoothing to the local density of data sample $\mathcal{S}_N$. The estimate is constructed similarly to the classical kernel estimate, but the scale parameter of the "bumps" placed on the sampled data points is allowed to vary from one point to another.

Let $K(x)$ be a kernel function and k a positive integer. Define $d_{i,k} = d_k(x^{(i)})$ to be the distance from $x^{(i)}$ to the k-th nearest point in the set

$\mathcal{S}_N^i = \mathcal{S}_N \backslash \{x^{(i)}\}$. Then the variable kernel estimate with smoothing parameter h is defined by

$$\hat{f}_{vk}(x; k, K, h, \mathcal{S}_N) = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{h d_{i,k}} K(\frac{x - x^{(i)}}{h d_{i,k}}) \tag{1.76}$$

The window width of the kernel placed on the point $x^{(i)}$ is proportional to $d_{i,k}$, so that sample points in regions where the data are sparse will have flatter kernels associated with them. For any fixed k, the overall degree of smoothing will depend on the parameter h. The choice of k determines how responsive the window width choice will be to very local detail.

Some comparison of the variable kernel estimate with the generalized k-th nearest neighbor estimate may be instructive. In generalized k-th nearest neighbor estimate (see Equation 1.75) the window width used to construct the estimate at point x depends on the distance criteria used and the distances from x to the data points $x' \in \mathcal{S}$, $d_k(x, x')$; in Equation 1.76 the window widths are independent of the point x at which the density is being estimated, and depend only on the distances between data points $x' \in \mathcal{S}$.

In contrast to generalized nearest neighbor estimate, if the kernel function used $K(x)$ is a probability density function, e.g. Gaussian density function, it follows from Equation 1.76 that variable kernel estimate will itself be a probability density function. Furthermore, as with the ordinary kernel estimator, all the local smoothness properties of the kernel will be inherited by the estimate.

1.14 Conclusions

This chapter is an introduction to the main concepts and procedures related to supervised classification task. We have formally introduced some concepts of probability theory indispensable for the understanding of the rest of the document. Based on these concepts, we have formally introduced the supervised classification task. Then, we introduced the decision theory in order to properly understand the classification process with probabilistic classifiers (generative and conditional classifiers). Besides, we have presented the three families of classifier induction algorithms in terms of the goal of the learning process: generative, conditional and discriminative learning. Moreover, we have provided tools for estimating the performance of a classifier and for comparing two or many classifiers based on the estimated performances. Focusing on classification error, we have shown a methodology in order to analyze the sources of the error associated to a classifier induction algorithm. We also have introduced a set of performance measures based on the confusion matrix and the ROC curve. In addition, this chapter introduces a set of quantities of information theory useful to measure the uncertainty surrounding the random variables. We have presented the curse of dimensionality concept providing

some intuitions and, in order to avoid this problem, some feature subset selection algorithms are explained, highlighting their merits and caveats. We have explained the discretization process of the continuous random variables including some of the most popular discretization algorithms. And finally, we have introduced a set of non-parametric density estimators.

2

Supervised classification with Bayesian multinomial networks

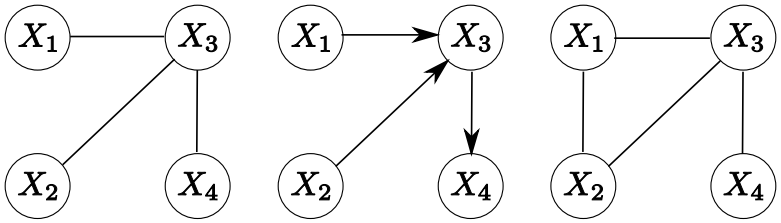
2.1 Introduction

This chapter formally introduces Bayesian multinomial networks in general and Bayesian multinomial networks with classification purposes in particular. The chapter is divided in four sections. Section 2.2 formally introduces graph theory in order to correctly interpret the conditional (in)dependence assertions encoded by a graph. Since we are interested in Bayesian networks, the section is mainly focused on directed acyclic graphs. Then, Section 2.3 formally introduces the probabilistic graphical models which represent a factorization of the generalized joint probability of the random variables included in the model. Bayesian multinomial networks are formally presented in Section 2.4. Then, Section 2.5 presents a family of classifiers based on Bayesian networks: *augmented naïve Bayes classifiers*. Section 2.6 shows the main approaches in order to deal with continuous features with Bayesian networks indicating their main drawbacks. This section motivates the main methodological contributions summarized in this work (see Chapters 3 and 4). Finally, Section 2.7 enumerates the main concepts introduced in this chapter and, besides, it indicates the main future work lines considered by the author.

2.2 Graph theory

Graphs provide us with a powerful formalism that is widely accepted as the representation of probabilistic graphical models. This section introduces some useful definitions as well as the notation concerning the graph theory that we will use in the remainder of this dissertation. The theoretical results provided in this section have been taken from [Peña (2001)].

Definition 2.1 *A* graph *is an ordered pair* $\boldsymbol{G} = (\boldsymbol{X}, \boldsymbol{E})$ *where* $\boldsymbol{X}$ *is a non-empty finite set of vertices (also called nodes) and* $\boldsymbol{E}$ *is a set of ordered pairs of distinct nodes of* $\boldsymbol{X}$ *called edges.*



(a) Undirected graph $\boldsymbol{G}_U$. (b) Directed graph $\boldsymbol{G}_D$. (c) Moral graph $\boldsymbol{G}_M$.

Fig. 2.1. This figure shows three graphs in $\boldsymbol{X} = (X_1, ..., X_4)$ which are related among them: the undirected graph $\boldsymbol{G}_U = (\boldsymbol{X}, \boldsymbol{E}_U)$, the DAG $\boldsymbol{G}_D = (\boldsymbol{X}, \boldsymbol{E}_D)$ and the moral graph $\boldsymbol{G}_M = (\boldsymbol{X}, \boldsymbol{E}_M)$, where $\boldsymbol{E}_U = \{(X_1, X_3), (X_3, X_1), (X_2, X_3), (X_3, X_2), (X_3, X_4), (X_4, X_3)\}$, $\boldsymbol{E}_D = \{(X_1, X_3), (X_2, X_3), (X_3, X_4)\}$ and $\boldsymbol{E}_U = \{(X_1, X_2), (X_2, X_1), (X_1, X_3), (X_3, X_1), (X_2, X_3), (X_3, X_2), (X_3, X_4), (X_4, X_3)\}$. The undirected graphs $\boldsymbol{G}_U$ and $\boldsymbol{G}_M$ are the underlaying and the moral graphs of $\boldsymbol{G}_D$, respectively, and they differ in the edges associated to variables with at least one common child, X_1 and X_2, $\boldsymbol{E}_M \setminus \boldsymbol{E}_U = \{(X_1, X_2), (X_2, X_1)\}$. The following parental relations can be identified in $\boldsymbol{G}_D$: $\boldsymbol{Pa}_1 = \emptyset$, $\boldsymbol{Pa}_2 = \emptyset$, $\boldsymbol{Pa}_3 = \{X_1, X_2\}$, $\boldsymbol{Pa}_4 = \{X_3\}$, $\boldsymbol{Ch}_1 = \{X_3\}$, $\boldsymbol{Ch}_2 = \{X_3\}$, $\boldsymbol{Ch}_3 = \{X_4\}$ and $\boldsymbol{Ch}_4 = \emptyset$.

Definition 2.2 *Given a graph $\boldsymbol{G} = (\boldsymbol{X}, \boldsymbol{E})$, if $(X_i, X_j) \in \boldsymbol{E}$ and $(X_j, X_i) \in \boldsymbol{E}$, then the edge between X_i and X_j is called an* undirected edge *for all i and j.*

The graphical representation of a graph $\boldsymbol{G} = (\boldsymbol{X}, \boldsymbol{E})$ reflects the existence of an undirected edge between X_i and X_j with a line between X_i and X_j, i.e. $X_i - X_j$, for all i and j. An example of the graphical representation of an undirected graph is shown in Figure 2.1(a).

Definition 2.3 *Given a graph $\boldsymbol{G} = (\boldsymbol{X}, \boldsymbol{E})$, if $(X_i, X_j) \in \boldsymbol{E}$ but $(X_j, X_i) \notin \boldsymbol{E}$, then the edge from X_i to X_j is called a* directed edge*, and X_i is parent of X_j and X_j is a child of X_i for all i and j.*

The graphical representation of a graph $\boldsymbol{G} = (\boldsymbol{X}, \boldsymbol{E})$ reflects the existence of a directed edge from X_i to X_j with an arrow from X_i to X_j, i.e. $X_i \rightarrow X_j$. In this dissertation, the set of all the parents of a given node X_i in $\boldsymbol{G}$ is represented by $\boldsymbol{Pa}(\boldsymbol{G})_i$ for all i. On the other hand, the set of all the children of a given node X_i in $\boldsymbol{G}$ is represented by $\boldsymbol{Ch}(\boldsymbol{G})_i$ for all i. Both representations, $\boldsymbol{Pa}(\boldsymbol{G})_i$ and $\boldsymbol{Ch}(\boldsymbol{G})_i$, can be replaced by $\boldsymbol{Pa}_i$ and $\boldsymbol{Ch}_i$ when the context makes clear the graph $\boldsymbol{G}$ that is being considered. An example of a directed graph is shown in Figure 2.1(b).

Definition 2.4 *Let $\boldsymbol{G} = (\boldsymbol{X}, \boldsymbol{E})$ be a graph. A* path *from X_i to X_j in $\boldsymbol{G}$ is a sequence of distinct nodes $X_{i_1}, ...X_{i_m}$ with $m > 1$ such that $X_{i_1} = X_i$, $X_{i_m} = X_j$, and $(X_{i_k}, X_{i_{k+1}}) \in \boldsymbol{E}$ for all k such that $1 \leq k < m$. The* length

of the path *is the number of nodes in the path minus one, i.e.* $m-1$. *A cycle is a path with the modification that* $X_i = X_j$.

Definition 2.5 *A graph* $\boldsymbol{G} = (\boldsymbol{X}, \boldsymbol{E})$ *is called a* complete graph *if there is an undirected or directed edge between every pair of distinct nodes of* $\boldsymbol{X}$.

The definition of complete graph is adapted to the family of augmented naïve Bayes structures (complete structures) in Section 2.5.2.

Definition 2.6 *Let* $\boldsymbol{G} = (\boldsymbol{X}, \boldsymbol{E})$ *be a graph.* $\boldsymbol{G}$ *is an* undirected graph *if there are only undirected edges in* $\boldsymbol{E}$.

When it is clear from the context that the graph used is either undirected or directed, its edges (undirected and directed, respectively) are usually called arcs. Figure 2.1(a) shows an example of an undirected graph.

Definition 2.7 *Let* $\boldsymbol{G} = (\boldsymbol{X}, \boldsymbol{E})$ *be a graph.* $\boldsymbol{G}$ *is a* directed graph *if there are only directed edges in* $\boldsymbol{E}$.

Definition 2.8 *Let* $\boldsymbol{G} = (\boldsymbol{X}, \boldsymbol{E})$ *be a graph.* $\boldsymbol{G}$ *is a* directed acyclic graph *(DAG) if* $\boldsymbol{G}$ *is a directed graph and it does not contain cycles.*

DAGs play a basic role in the field of probabilistic graphical models. Figure 2.1(b) shows an example of a DAG.

Definition 2.9 *Let* $\boldsymbol{G} = (\boldsymbol{X}, \boldsymbol{E})$ *be a directed graph. If there exists a path from* X_i *to* X_j *in* $\boldsymbol{G}$*, then* X_i *is an* ascendant *of* X_j *and* X_j *is a* descendant *of* X_i *for all* i *and* j. *A set of nodes* $\boldsymbol{Y} \subseteq \boldsymbol{X}$ *that contains all the ascendants of* X_i *is referred to as an* ancestral set.

Definition 2.10 *Let* $\boldsymbol{G} = (\boldsymbol{X}, \boldsymbol{E})$ *be a DAG. An* ancestral ordering *of* $\boldsymbol{G}$ *is a total ordering of* $\boldsymbol{X}$ *where* X_i *appears before* X_j *if* X_i *is a parent of* X_j.

Note that every node appears before its children in an ancestral ordering of a given DAG. It is well known that there exists at least one ancestral ordering for every DAG.

Definition 2.11 *Let* $\boldsymbol{G} = (\boldsymbol{X}, \boldsymbol{E})$ *be a graph. The undirected graph that is obtained from* $\boldsymbol{G}$ *by replacing directed edges by undirected edges is called* underlying undirected graph *of* $\boldsymbol{G}$ *or* embedded graph *of* $\boldsymbol{G}$.

For example, the underlying undirected graph of the DAG presented in Figure 2.1(b), $\boldsymbol{G}_D$, is the undirected graph $\boldsymbol{G}_U$ shown in Figure 2.1(a).

Definition 2.12 *A graph* $\boldsymbol{G} = (\boldsymbol{X}, \boldsymbol{E})$ *is called* connected graph *if there is a path between every pair of distinct vertices in the underlying directed graph of* $\boldsymbol{G}$. *Otherwise,* $\boldsymbol{G}$ *is disconnected.*

Definition 2.13 *The graph* $\boldsymbol{G^Y} = (\boldsymbol{Y}, \boldsymbol{E^Y})$ *is called a* subgraph *of the graph* $\boldsymbol{G} = (\boldsymbol{X}, \boldsymbol{E})$ *if* $\boldsymbol{Y} \subseteq \boldsymbol{X}$ *and* $\boldsymbol{E^Y} \subseteq \boldsymbol{E}$ *such that for all* $(X_i, X_j) \in \boldsymbol{E^Y}$*, we have that* $X_i, X_j \in \boldsymbol{Y}$*. Moreover, if for all* $(X_i, X_j) \in \boldsymbol{E}$ *such that* $X_i, X_j \in \boldsymbol{Y}$ *we have that* $(X_i, X_j) \in \boldsymbol{E^Y}$*, the* $\boldsymbol{G^Y}$ *is called the subgraph of* $\boldsymbol{G}$ *induced by the set of nodes* $\boldsymbol{Y}$*.*

Definition 2.14 *Let* $\boldsymbol{G} = (\boldsymbol{X}, \boldsymbol{E})$ *be a directed graph. The* moral graph *of* $\boldsymbol{G}$ *is defined as the undirected graph obtained from* $\boldsymbol{G}$ *by first adding undirected edges between all the pairs of nodes that have children in common and that are not already joined, and then forming the embedded graph of the resulting graph.*

Figure 2.1(c) shows G_M which is the moral graph of the directed graph G_D, shown in Figure 2.1(b). Note that the moral graph of G_D, G_M, is different from its undirected graph, G_U, shown in Figure 2.1(a).

2.2.1 Conditional independence models as directed acyclic graphs

Given a graph $\boldsymbol{G} = (\boldsymbol{X}, \boldsymbol{E})$, every edge of $\boldsymbol{E}$ can be seen as a relationship between two nodes of $\boldsymbol{X}$. Thus, a graph could be defined as a set of nodes and a set of relationships between them. This is a very appealing point of view that is supported by the broad use of graphs as a powerful formalism for representing conditional independence models, i.e. lists of conditional independence assertions (see Definition 2.15). We focus on the use of DAGs for this purpose, as this kind of graphs plays an important role in the classes of probabilistic graphical models we are interested in. See Castillo et al. (1997) for a parallel study of undirected graphs and DAGs as tools for encoding conditional independence models. In particular, the set of conditional independence statements that is encoded by a DAG corresponds somehow to the relationships, i.e. edges, between the nodes of the graph, i.e. random variables. This section formally presents this idea. It is focused on obtaining the conditional independence model encoded by a given DAG, in other words, on extracting the semantics of the DAG. For this purpose we need a set of formal definitions that enables us to read the conditional independence model that a DAG encodes.

Definition 2.15 *A* conditional independence model *for* $\boldsymbol{X} = (X_1, ..., X_n)$ *consists of a list of conditional independence assertions of the form* $CI(\boldsymbol{U}; \boldsymbol{V}|\boldsymbol{W})$ *where* $\boldsymbol{U}$*,* $\boldsymbol{V}$ *and* $\boldsymbol{W}$ *are disjoint subsets of* $\boldsymbol{X}$*, and* $\boldsymbol{U}$ *and* $\boldsymbol{V}$ *are non-empty sets.*

Definition 2.16 *Let* $\boldsymbol{X} = (X_1, ..., X_n)$ *be a random variable and* $\boldsymbol{U}$*,* $\boldsymbol{V}$ *and* $\boldsymbol{W}$ *are disjoint subsets of* $\boldsymbol{X}$*, and* $\boldsymbol{U}$ *and* $\boldsymbol{V}$ *are non-empty sets. The* conditional independence model *induced from a generalized joint probability distribution for* $\boldsymbol{X}$*,* $\rho(\boldsymbol{X})$*, consists of the list of conditional independence assertions* $CI(\boldsymbol{U}; \boldsymbol{V}|\boldsymbol{W})$ *for any* $\boldsymbol{U}$*,* $\boldsymbol{V}$ *and* $\boldsymbol{W}$ *such that it is verified by* $\rho(\boldsymbol{X})$*.*

In the following subsection we present the probabilistic graphical models which relate the conditional independence assertions encoded by a DAG with the conditional independence assertions inducted from a factorization of the generalized joint probability distribution for $\boldsymbol{X}$. Subsequently, we focus our attention on defining the conditional independence model encoded by a DAG.

Definition 2.17 *Let $\boldsymbol{G} = (\boldsymbol{X}, \boldsymbol{E})$ be an undirected graph. Let $\boldsymbol{U}$, $\boldsymbol{V}$ and $\boldsymbol{W}$ be three disjoint subsets of $\boldsymbol{X}$. $\boldsymbol{W}$* u-separates *$\boldsymbol{U}$ and $\boldsymbol{V}$ in $\boldsymbol{G}$ if every path in $\boldsymbol{G}$ between a node belonging to $\boldsymbol{U}$ and a node belonging to $\boldsymbol{V}$ contains at least one node that belongs to $\boldsymbol{W}$.*

The *u-separation* assertion that states that $\boldsymbol{W}$ u-separates $\boldsymbol{U}$ and $\boldsymbol{V}$ in $\boldsymbol{G}$ is denoted by $US(\boldsymbol{U}; \boldsymbol{V}|\boldsymbol{W})_{\boldsymbol{G}}$ or, simply, by $US(\boldsymbol{U}; \boldsymbol{V}|\boldsymbol{W})$ when the context makes clear the DAG $\boldsymbol{G}$ that is considered. For example, some of the u-separation assertions that can be found in G_U, shown in Figure 2.1(a), are $US(X_1; X_2|X_3)$, $US(X_1; X_4|X_3)$ and $US(X_2; X_4|X_3)$.

Definition 2.18 *Let $\boldsymbol{G} = (\boldsymbol{X}, \boldsymbol{E})$ be a DAG. Let $\boldsymbol{U}$, $\boldsymbol{V}$ and $\boldsymbol{W}$ be three disjoint subsets of $\boldsymbol{X}$. $\boldsymbol{W}$* d-separates *$\boldsymbol{U}$ and $\boldsymbol{V}$ in $\boldsymbol{G}$ if $\boldsymbol{W}$ u-separates $\boldsymbol{U}$ and $\boldsymbol{V}$ in the moral graph of the smallest ancestral set that contains the nodes in $\boldsymbol{U}$, $\boldsymbol{V}$ and $\boldsymbol{W}$.*

The *d-separation* assertion that states that $\boldsymbol{W}$ d-separates $\boldsymbol{U}$ and $\boldsymbol{V}$ in $\boldsymbol{G}$ is represented by $DS(\boldsymbol{U}; \boldsymbol{V}|\boldsymbol{W})_{\boldsymbol{G}}$ or, simply, by $DS(\boldsymbol{U}; \boldsymbol{V}|\boldsymbol{W})$ when the context makes clear the DAG $\boldsymbol{G}$ that is considered. For example, some of the d-separation assertions that can be found in $\boldsymbol{G}_D$, shown in Figure 2.1(b), are $DS(X_1; X_4|X_3)_{\boldsymbol{G}_D}$ and $DS(X_2; X_4|X_3)_{\boldsymbol{G}_D}$.

We are aware that an alternative definition exists (see Pearl (1988); Castillo et al. (1997)). For the sake of brevity and readability, we have chosen the definition that, in our opinion, provides the most intuitive interpretation of the underlying concept. The interested reader may consult [Lauritzen et al. (1990)] where Definition 2.18 appears for the first time. Moreover, Lauritzen et al. (1990) show the equivalence between Definition 2.18 and the alternative definition proposed by Pearl (1988).

Theorem 6. *Let $\boldsymbol{G} = (\boldsymbol{X}, \boldsymbol{E})$ be a DAG. Then, every X_i in $\boldsymbol{X}$ is d-separated by $\boldsymbol{Pa}(\boldsymbol{G})_i$ from the rest of non-descendants of X_i.*

The reader will later realize that this theorem is of great help in the field of probabilistic graphical models (for example see Equation 2.2). The proof of Theorem 6 can be found in Pearl (1988).

Definition 2.19 *Let $\boldsymbol{G} = (\boldsymbol{X}, \boldsymbol{E})$ be a DAG. The conditional independence model encoded by $\boldsymbol{G}$ consists of the list of conditional independence assertions $CI(\boldsymbol{U}; \boldsymbol{V}|\boldsymbol{W})$ such that $CD(\boldsymbol{U}; \boldsymbol{V}|\boldsymbol{W})$ is true.*

In other words, any DAG for X implies a conditional independence model given by the list of d-separation assertions defined by the graph.

Definition 2.20 *The conditional independence model encoded by a DAG* $\boldsymbol{G} = (\boldsymbol{X}, \boldsymbol{E})$ *is a* conditional independence map *(*CI-map*) of a given conditional independence model if every statement* $DS(\boldsymbol{U}; \boldsymbol{V}|\boldsymbol{W})$ *that is derived from* $\boldsymbol{G}$ *implies that the statement* $CI(\boldsymbol{U}, \boldsymbol{V}|\boldsymbol{W})$ *belongs to the conditional independence model for any three disjoint subsets* $\boldsymbol{U}$*,* $\boldsymbol{V}$ *and* $\boldsymbol{W}$ *of* $\boldsymbol{X}$*.*

Note that a CI-map $\boldsymbol{G}$ for a conditional independence model encodes a subset, if not all, of the conditional independence statements of the conditional independence model. Thus, a CI-map never includes conditional independence statements that the conditional independence model does not verify. In other words, a d-separation assertion $DS(\boldsymbol{U}; \boldsymbol{V}|\boldsymbol{W})_{\boldsymbol{G}}$ implies a conditional independence assertion in the associated conditional independence model. This is a key point that probabilistic graphical models exploit.

With the help of Definition 2.20 it is easy to imagine several DAGs that are CI-maps of a given conditional independence model. In fact, any complete DAG is a trivial CI-map of any conditional independence model because a complete DAG does not encode any conditional independence statement. Then, it is obvious that not all CI-maps for a given conditional independence model are equally informative. The following definition characterizes the CI-maps that we prefer.

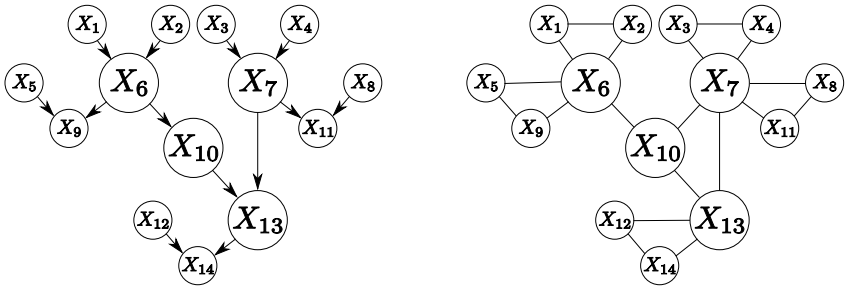
Definition 2.21 *A DAG* $\boldsymbol{G} = (\boldsymbol{X}, \boldsymbol{E})$ *is a* minimal CI-map *of a given conditional independence model if* $\boldsymbol{G}$ *is a CI-map of the conditional independence model and no proper subgraph of* $\boldsymbol{G}$ *is a* CI*-map of the conditional independence model.*

In other words, the removal of any of the arcs of a DAG that is a minimal CI-map of a given conditional independence model implies that the resulting DAG is no longer a CI-map of a given conditional independence model. Note that different CI-maps of a given conditional independence model could exist.

Definition 2.22 *Let* $\boldsymbol{G} = (\boldsymbol{X}, \boldsymbol{E})$ *be a DAG. Let* $\boldsymbol{Y} = (Y_1, ..., Y_m) = (X_{1:\boldsymbol{Y}_1}, ..., X_{m:\boldsymbol{Y}})$ *be a subset of* $\boldsymbol{X}$*, the* Markov blanket *of* $\boldsymbol{Y}$ *in* $\boldsymbol{G}$ *is a subset* $\boldsymbol{Z} \subset \boldsymbol{X}$ *that contains the parents* $\{\boldsymbol{Pa}(\boldsymbol{G})_{Y_i}\}_{i=1}^m$*, the children* $\{\boldsymbol{Ch}(\boldsymbol{G})_{\boldsymbol{Y}_i}\}_{i=1}^m$ *and the parents of the children* $\{\boldsymbol{Pa}(\boldsymbol{G})_{Y_j}|X_{j:\boldsymbol{Y}} \in \boldsymbol{Ch}(\boldsymbol{G})_{Y_i}\}_{i=1}^m$*, of* $X_{i:\boldsymbol{Y}}$ *for all* i*.*

In this dissertation, the Markov blanket of a node X_i in $\boldsymbol{G}$ is represented by $\boldsymbol{Mb}(\boldsymbol{G})_i$ for all i or simply by $\boldsymbol{Mb}_i$, when the context makes clear the graph $\boldsymbol{G}$ that is being considered. On the other hand, the Markov blanket of a random variable $\boldsymbol{Y} \in \boldsymbol{X}$ is represented by $\boldsymbol{Mb}(\boldsymbol{G})_{\boldsymbol{Y}}$ or simply by $\boldsymbol{Mb}_{\boldsymbol{Y}}$, when the context makes clear the graph $\boldsymbol{G}$ that is being considered.

Theorem 7. *Let* $\boldsymbol{G} = (\boldsymbol{X}, \boldsymbol{E})$ *be a DAG. Let* $\boldsymbol{Y}$ *and* $\boldsymbol{Z}$ *be two disjoint subsets of* $\boldsymbol{X}$*,* $\boldsymbol{Y} \cap \boldsymbol{Z} = \emptyset$*, where* $\boldsymbol{Z}$ *is the Markov blanket of* $\boldsymbol{Y}$ *in* $\boldsymbol{G}$*,* $\boldsymbol{Z} = \boldsymbol{Mb}(\boldsymbol{G})_{\boldsymbol{Y}}$*, then* $\boldsymbol{Y}$ *and* $\boldsymbol{X} \setminus (\boldsymbol{Y}, \boldsymbol{Z})$ *are d-separated by* $\boldsymbol{Z}$*.*



(a) The DAG $\boldsymbol{G}$. (b) The moral graph of the DAG $\boldsymbol{G}$.

Fig. 2.2. These figures illustrate the Markov blanket concept based on a directed graph and its associated moral graph. We are interested in the Markov Blanket of X_{10}, $\boldsymbol{Mb}_{10} = (X_6, X_7, X_{10}, X_{13})$, which consists of the parents of X_{10}, $\boldsymbol{Pa}_{10} = \{X_6\}$, the children of X_{10}, $\boldsymbol{Ch}_{10} = \{X_{13}\}$, and the parents of the children X_{13}, $\boldsymbol{Pa}_{13} = \{X_7, X_{10}\}$. The Markov blanket of X_{10} can be obtained from the moral graph of $\boldsymbol{G}$ shown in Figure 2.2(b): the set $\boldsymbol{Mb}_{10} = \{X_6, X_7, X_{13}\}$ clearly u-separates X_{10} from the rest of the nodes. The Markov blanket of X_{10} is represented with bigger nodes in both figures.

The proof of this theorem can be found in Pearl (1988) and it can be derived from Definitions 2.18 and 2.20.

The classifiers based on probabilistic graphical models are interested in modeling $p(c|\boldsymbol{x})$ rather than $p(\boldsymbol{x}, c)$. Theorem 7 will be useful in order to approach $p(c|\boldsymbol{x})$ with $p(c|\boldsymbol{mb}_C)$, which includes all the random variables in $\boldsymbol{X}$ that have a direct influence on C reducing, at the same time, the number of parameters with respect to $p(c|\boldsymbol{x})$.

2.3 Probabilistic graphical models

This section introduces probabilistic graphical models as powerful tools for supporting modeling and reasoning under uncertainty in complex domains. We present probabilistic graphical models as a formal paradigm that enables us to encode a generalized joint probability distribution for the domain random variables. Most of the results presented in this Section has been adapted from [Peña (2001)].

Let $\boldsymbol{X} = (X_1, ..., X_n)$ be an n-dimensional random variable. A *probabilistic graphical model* (*PGM*) for $\boldsymbol{X}$ is a graphical factorization of a generalized joint probability distribution for $\boldsymbol{X}$, $\rho(\boldsymbol{x})$ [Pearl (1988); Whittaker (1990); Bouckaert (1995); Jensen (1996); Lauritzen (1996); Castillo et al. (1997); Cowell et al. (1999); Jensen and Lauritzen (2000)]. A PGM for $\boldsymbol{X}$ consists of two components: A graph $\boldsymbol{s}$ (*model structure*) which determines the conditional (in)dependencies between the random variables of $\boldsymbol{X}$, and a set of *local probability distributions* for the model structure. Thus, every node of the model

structure $\boldsymbol{s}$ is interchangeably called node and random variable in the forthcoming discussion. The model structure represents the *qualitative* part of the PGM, whereas the set of local probability distributions can be thought as the *quantitative* part of the model.

As previously stated, this dissertation is limited to PGMs where the structural part is a DAG. Therefore, neither undirected graphs [Wermuth (1976); Whittaker (1990)] nor chain graphs [Lauritzen and Wermuth (1989)] are considered in this dissertation.

From the chain rule of probability (see Equation 1), we have that the generalized joint probability distribution encoded by a PGM for $\boldsymbol{X}$ graphically factorizes as follows:

$$\rho(\boldsymbol{x}) = \rho(x_1, ..., x_n) = \prod_{i=1}^{n} \rho(x_i | x_1, ..., x_{i-1}) \tag{2.1}$$

We can assume without loss of generality that the structure of the PGM, $\boldsymbol{s}$, obeys an ancestral ordering such that every X_i holds the i-th place in the ordering. As it is well known that there exists, at least, an ancestral ordering for every DAG, if $\boldsymbol{s}$ does not obey the ancestral ordering that we assume, then its nodes can be renamed to do so. Consequently, the set $(X_1, ..., X_{i-1})$ does not include descendants of X_i for all i. Moreover, the model structure $\boldsymbol{s}$ induces a conditional independence model where, given $\boldsymbol{Pa}(\boldsymbol{s})_i$, X_i is conditionally independent of the rest of its non-descendants for all i. Note that this result can be directly obtained from Theorem 6 and Definition 2.20. Then, the model structure $\boldsymbol{s}$ encodes a set of conditional independence statements of the form $CI(X_i; (X_1, ..., X_{i-1}) \setminus \boldsymbol{Pa}(\boldsymbol{s})_i | \boldsymbol{Pa}(\boldsymbol{s})_i)$ for all i. Thus, by Definition 1.17 we have that $\rho(x_i | x_1, ..., x_{i-1}) = \rho(x_i | \boldsymbol{pa}(\boldsymbol{s})_i)$. Therefore, Equation 2.1 can be rewritten as follows:

$$\rho(\boldsymbol{x}|\boldsymbol{s}) = \prod_{i=1}^{n} \rho(x_i | \boldsymbol{pa}(\boldsymbol{s})_i) \tag{2.2}$$

The local probability distributions of the PGM are those induced by the terms of the product that appears in Equation 2.2, and we assume that they depend on a finite set of parameters $\boldsymbol{\Theta_s} = (\boldsymbol{\Theta}_1, ..., \boldsymbol{\Theta}_n)$ (*model parameters*). Moreover, assuming that the true generalized joint probability distribution for $\boldsymbol{X}$ can be graphically factorized according to the conditional independence assertions reflected in $\boldsymbol{s}$ (*model structure hypothesis*), Equation 2.2 can be rewritten as follows:

$$\rho(\boldsymbol{x}|\boldsymbol{\Theta_s}, \boldsymbol{s}) = \prod_{i=1}^{n} \rho(x_i | \boldsymbol{pa}(\boldsymbol{s})_i, \boldsymbol{\Theta}_i) \tag{2.3}$$

The result of Equation 2.3 is reported by several researchers Kiivery et al. (1984); Pearl (1988); Bouckaert (1995); Castillo et al. (1997) in the form of the following theorem.

Theorem 8. *Let* $(\boldsymbol{s}, \boldsymbol{\Theta_s})$ *be a PGM for* $\boldsymbol{X}$ *as defined above. Then,*

$$\rho(\boldsymbol{x}|\boldsymbol{\Theta_s}, \boldsymbol{s}) = \prod_{i=1}^{n} \rho(x_i|\boldsymbol{pa}(\boldsymbol{s})_i, \boldsymbol{\Theta}_i) \tag{2.4}$$

Moreover, the DAG $\boldsymbol{s}$ *is a minimal CI-map of the conditional independence model induced by* $\rho(\boldsymbol{x}|\boldsymbol{\Theta_s}, \boldsymbol{s})$.

The proof of the theorem can be found in the referred works. At this point of the discussion, it should be clear why we are more interested in the conditional independence statements than in the conditional dependence statements that a PGM encodes: Conditional independence statements enable us to graphically factorize the generalized joint probability distribution for $\boldsymbol{X}$, resulting in a reduction in the number of parameters that are needed to completely specify this distribution.

Thus, as we will see in the next subsection, in order to define a PGM it is necessary to specify:

- A DAG $\boldsymbol{s}$ encoding the set of conditional (in)dependence statements between the random variables of the problem domain.
- A set of local probability distributions for the model structure.

2.3.1 Learning a PGM

Once a PGM is built, it constitutes and effective tool for reasoning with uncertainty. Nevertheless, the problem of learning the model remains. The structure $\boldsymbol{s}$ and the set of local probabilities $\rho(x_i|\boldsymbol{pa}(\boldsymbol{s})_i, \boldsymbol{\Theta}_i)$, determined by the parameters $\boldsymbol{\Theta}_i$, that characterize the PGM can be provided either externally by experts, which can be a time consuming process subject to mistakes, or, by learning from a database of instances following an automatic learning procedure. This dissertation is focused on the second approach with classification purposes.

PGM based model learning (also known as *model induction*, *model selection* or *model elicitation*) can be thought of as being separated into two subtasks: *structural leaning* (or *qualitative learning*) and *parametric learning* (or *quantitative learning*). Structural learning is the identification of the graph structure of the model, $\boldsymbol{s}$ and, parametric learning is the numerical assessment of the parameters $\boldsymbol{\Theta}_i$ of the local probability distributions $\rho(x_i|\boldsymbol{pa}(\boldsymbol{s})_i, \boldsymbol{\Theta}_i)$ for a given model structure $\boldsymbol{s}$.

Consequently, most of the learning algorithms work in two stages. Firstly, the appropriate model structure $\boldsymbol{s}$ is determined and, then, the parameters for the selected structure $\boldsymbol{\Theta_s}$ are estimated according to maximum a posteriori (MAP) criterion, i.e. the probability of the parameters given the database at hand and the model structure is a local maximum, or according to maximum likelihood (ML) criterion, i.e. the likelihood of the database at hand given the parameters and the model structure is a local maximum, among others. Under

reasonable assumptions the MAP and the ML parameters for a given model structure can be easily computed from data. Thus, usually, model induction reduces to structural learning. Through this dissertation we only consider the ML approach for learning the parameters and, therefore, the classifier learning algorithms presented in the subsequent sections are mainly focused on the structural learning procedure used.

Structural learning usually involves a search process in the space of graph structures. The search process tries to optimize a score and it generally finishes when a local optimum is found. We consider that structural learning can be carried out in a filter or a wrapper way, depending on the kind of the score which guides the search process. These filter and wrapper concepts are adapted from the feature subset selection literature. In this dissertation, the filter approaches use likelihood as a score and the wrapper approaches use the estimated predictive accuracy.

The space of the graph structures is exponential with the number of variables implied. In order to reduce the search space the structural learning is usually constrained. The constraints are usually defined in terms of the kind and number of (in)dependencies allowed among the variables. This dissertation is focused on augmented naïve Bayes family of structures (see Section 2.5.2).

2.4 Bayesian multinomial networks

This section introduces one of the classes of PGMs we are interested in: Bayesian networks [Pearl (1988); Jensen (1996); Lauritzen (1996); Castillo et al. (1997); Cowell et al. (1999); Peña (2001); Korb and Nicholson (2004); Jensen and Nielsen (2007)].

In the particular case that $\boldsymbol{X} = (X_1, ..., X_n)$ is an n-dimensional discrete random variable, a PGM for $\boldsymbol{X}$ is called *Bayesian network* (*BN*) for $\boldsymbol{X}$ [Pearl (1988); Jensen (1996); Lauritzen (1996); Castillo et al. (1997); Cowell et al. (1999); Korb and Nicholson (2004); Jensen and Nielsen (2007)]. Then, the graphical factorization of the joint probability distribution for $\boldsymbol{X}$ encoded by a BN for $\boldsymbol{X}$ acquires the following form:

$$p(\boldsymbol{x}|\boldsymbol{\Theta_s}, \boldsymbol{s}) = p(x_1, ..., x_n|\boldsymbol{\Theta_s}, \boldsymbol{s}) = \prod_{i=1}^{n} p(x_i|\boldsymbol{pa}(\boldsymbol{s})_i, \boldsymbol{\Theta}_i) \tag{2.5}$$

where $\boldsymbol{\Theta_s} = (\boldsymbol{\Theta}_1, ..., \boldsymbol{\Theta}_n)$ and $\boldsymbol{s}$ are the model parameters and the model structure hypothesis respectively. Note that Equation 2.5 is a particular case of Equation 2.2 that takes into account the discrete nature of $\boldsymbol{X}$.

Typically, the *local probability distributions* for every X_i, $p(x_i|\boldsymbol{pa}(\boldsymbol{s})_i, \boldsymbol{\Theta}_i)$, are restricted to be a set of univariate multinomial distributions, one for each value of $\boldsymbol{Pa}(\boldsymbol{s})_i$. This is also the case in this dissertation. In order to highlight

this particular case from here on we will refer BN to as *Bayesian multinomial network* (*BMN*).

Let us assume that X_i can take r_i distinct values (or states) denoted by $x_i^1, ..., x_i^{r_i}$, and $\boldsymbol{Pa}(\boldsymbol{s})_i$ can have q_i distinct states denoted by $\boldsymbol{pa}(\boldsymbol{s})_i^1, ..., \boldsymbol{pa}(\boldsymbol{s})_i^{q_i}$ with $q_i = \prod_{X_l \in \boldsymbol{Pa}(\boldsymbol{s})_i} r_l$ for all i. Then the univariate multinomial distribution $p(x_i|\boldsymbol{pa}(\boldsymbol{s})_i^j, \boldsymbol{\Theta}_i)$ for all i and j consist of a set of probabilities of the form

$$p(x_i^k|\boldsymbol{pa}(\boldsymbol{s})_i^j, \boldsymbol{\Theta}_i) = \Theta_{x_i^k|\boldsymbol{pa}(\boldsymbol{s})_i^j} = \Theta_i^{jk} \tag{2.6}$$

such that $\Theta_i^{jk} > 0$ representing the conditional probability that X_i takes its k-th state given that $\boldsymbol{Pa}(\boldsymbol{s})_i$ takes its j-th value for all k. Moreover, $\sum_{k=1}^{r_i} \Theta_i^{jk} = 1$ for all i and j.

Consequently, the parameters of the local probability distribution for every X_i are given by $\boldsymbol{\Theta}_i = (\boldsymbol{\Theta}_i^1, ..., \boldsymbol{\Theta}_i^{q_i})$ with $\boldsymbol{\Theta}_i^j = (\Theta_i^{j1} ..., \Theta_i^{jr_i})$ for all j. The number of free parameters of the local probability distributions for X_i given each value of its parents $\boldsymbol{Pa}(\boldsymbol{s})_i$ is $|\boldsymbol{\Theta}_i| = (r_i - 1) \cdot q_i$. Note that the number of free parameters for a multinomial distribution with parameters $\boldsymbol{\Theta} = (\Theta^1, ..., \Theta^r)$ is $r-1$ because $\sum_{i=1}^{r} \Theta^i = 1$. The total number of parameters of the BMN for $\boldsymbol{X}$, $(\boldsymbol{s}, \boldsymbol{\Theta}_{\boldsymbol{s}})$ is given by $|\boldsymbol{\Theta}| = \sum_{i=1}^{n} (r_i - 1) \cdot q_i$ where $q_i = \prod_{X_l \in \boldsymbol{Pa}(\boldsymbol{s})_i} r_l$ tends to increase exponentially with the cardinality $|\boldsymbol{Pa}(\boldsymbol{s})_i|$ for all i. Therefore, the order of the number of parameters of the BMN is given by $\arg_i \max(r_i - 1)q_i$. If we assume that $r_i = r$ for all i, the order of the number of parameters is given by $\arg_i \max r^{|Pa(\boldsymbol{s})_i|+1}$. This growth is problematic because the risk to overfitting of a classifier tends to increase with the number of parameters required to be estimated [Bishop (2006)]. Moreover, the number of relevant cases used to compute each parameter can be very low and, therefore, the statistics obtained might not be robust [Hand and Yu (2001)]. However, the number of parameters required for a BMN can be intuitively controlled imposing constraints to the parents of X_i, $\boldsymbol{Pa}(\boldsymbol{s})_i$, for all i. This intuition leads to the augmented naïve Bayes family of structures (see Section 2.5.2). The cardinality of the parents $\boldsymbol{Pa}(\boldsymbol{s})_i$ for all i is determined by the number of arcs $E_i = \{(X_j, X_i) \in \boldsymbol{s} \text{ for all } j\}$. Thus, in the worst case, the parameters could increase exponentially with the number of arcs in BMNs.

Analogously to the case of a PGM (see Section 2.3), in order to define a Bayesian network, it is necessary to specify:

- A DAG $\boldsymbol{s}$ encoding the set of conditional (in)dependence statements between the random variables of the problem domain.
- A set of the local probability distributions for the model structure, i.e. conditional probabilities Θ_i^{jk} for all i, j and k.

Section 2.5 presents several examples of structural learning algorithms for the induction of augmented naïve Bayes classifiers based on BMN.

As we noted in Section 2.3 the parametric learning is performed using the maximum likelihood estimator and the parameters required are given by the structure $\boldsymbol{s}$ (see Equations 2.5 and 2.6). The maximum likelihood estimator for

the parameters of a multinomial distribution $p(x_i^j; \boldsymbol{pa}(\boldsymbol{s})_i^k, \boldsymbol{\Theta}_i) = \Theta_i^{jk}$ given the data $\boldsymbol{D} = \{\boldsymbol{x}^{(1)}, ..., \boldsymbol{x}^{(N)}\}$ is

$$\hat{\Theta}_i^{jk} = \frac{N_i^{jk}}{N_i^{\cdot k}} \tag{2.7}$$

where N_i^{jk} is the number of cases in $\boldsymbol{D}$ that the variable X_i takes its j-th value when the variables $\boldsymbol{Pa}(\boldsymbol{s})_i$ have their k-th value, and $N_i^{\cdot k}$ is the number of cases in $\boldsymbol{D}$ that the variables $\boldsymbol{Pa}(\boldsymbol{s})_i$ take their k-th value.

2.5 Bayesian multinomial networks for classification

This section presents a family of classifier induction algorithms for the BMN paradigm. First, Subsection 2.5.1 introduces the notion of a classifier based on Bayesian multinomial networks. Then, Subsection 2.5.2 presents a set of related families of Bayesian network structures biased towards classification: *naïve Bayes*, *tree-augmented naïve Bayes*, *k-dependent augmented naïve Bayes* and *joint augmented naïve Bayes* structures. Subsection 2.5.3 introduces a set of classifier induction algorithms based on Bayesian multinomial networks for each of the previously presented families of structures. And finally, Section 2.6 presents the most popular approaches in order to deal with continuous features in BMN paradigm, indicating its main alternatives.

2.5.1 BMN based classifiers

BMN based classifiers (and in general PGM based classifiers) are usually considered generative classifiers (see Section 1.6) because they solve the classification problem by modeling the generalized joint probability distribution for $(\boldsymbol{X}, C)$, $\rho(\boldsymbol{x}, c)$. Then, based on the model $\rho_\phi(\boldsymbol{x}, c)$, they obtain the conditional probability distribution $p_\phi(c|\boldsymbol{x})$ using Equation 1.20. Finally, the classification rule is obtained plugging-in the modeled conditional probability $p_\phi(c|\boldsymbol{x})$ into weighted-winner-takes-all-rule (see Equation 1.21).

2.5.2 Structures biased towards classification

This section presents a family of structures designed for classification. They are focused on obtaining good estimators of the conditional distribution $p(c|\boldsymbol{x})$ for all $\boldsymbol{x}$. Thus, classifiers based on PGM are more interested in obtaining a good estimator of $p(c|\boldsymbol{x})$, with classification purposes, rather than obtaining an accurate model of the generalized joint probability distribution $p(\boldsymbol{x}, c)$. In order to characterize appropriate structures with classification purposes we propose the following intuitive heuristics:

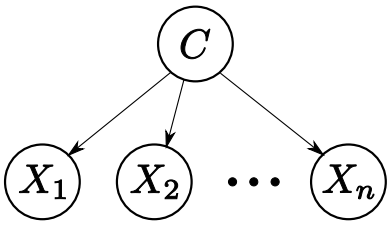
- Assuming that the variables $\boldsymbol{X} = (X_1, ..., X_n)$ are relevant for classification, the structure of the PGM should include at least a set of directed arcs corresponding to the dependence assertions $CD(X_i; C|\boldsymbol{U})$, for any $\boldsymbol{U} \subseteq \boldsymbol{X} \setminus X_i$, for all i. This intuition is closely related to the concept of strongly relevant feature (see Definition 1.24). This heuristic leads to accurate classification models, i.e. estimators of $p(c|\boldsymbol{x})$ with low bias (see Section 1.8.3).
- Minimize the number of parameters of the model $\boldsymbol{\Theta_s}$ (see Section 2.4). This leads to robust classification models, i.e. estimators of $p(c|\boldsymbol{x})$ with low variance (see Section 1.8.3).

One possible approach consists of, firstly, reducing the set of variables for obtaining a set of relevant variables for classification (see Section 1.11). Then, assuming that all the selected variables $\boldsymbol{X} = (X_1, ..., X_n)$ are relevant, the first heuristic guarantees that the variable X_i and the class C will be conditional dependent given the value of any $\boldsymbol{U} \subseteq \boldsymbol{X} \setminus X_i$. In other words, there exist two different values x_i and $x_{i'}$ for which the distribution of C changes given $\boldsymbol{x}_{\neg i}$, $p(c|x_i, \boldsymbol{u}) \neq p(c|x_{i'}, \boldsymbol{u})$ for some c. In order to classify an unlabeled instance $\boldsymbol{x}$ the conditional dependence $CD(X_i; C|\boldsymbol{U})$ for any $\boldsymbol{U} \subseteq \boldsymbol{X} \setminus X_i$ guarantees the influence of X_i in the model, $\hat{p}_\phi(c|\boldsymbol{x})$, for all i. The first heuristic implies that the relevant variables for classification $\boldsymbol{X}$ should be included in the Markov Blanket of the class C, $\boldsymbol{Mb}(\boldsymbol{s})_C$ (see Definition 2.22 and Theorem 7).

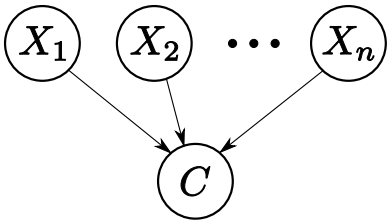
The second heuristic states that we should try to minimize the number of parameters, $|\boldsymbol{\Theta_s}|$. In order to minimize the parameters, the number of parents of each variable should be minimized as we suggest in Section 2.4. Thus, it is preferable to include the arcs $\{(C, X_i)\}_{i=1}^n$ (or $\{C \rightarrow X_i\}_{i=1}^n$) instead of $\{(X_i, C)\}_{i=1}^n$ (or $\{X_i \rightarrow C\}_{i=1}^n$). This fact is illustrated in Figure 2.3. In order to include X_i in the Markov blanket of C, there is a third option: an arc from X_i to X_j can be added to the structure, for any $X_i \notin \boldsymbol{Ch}(\boldsymbol{s})_C$ and $X_j \in \boldsymbol{Ch}(\boldsymbol{s})_C$. But this option does not include the dependence $CD(X_i; C|\emptyset)$ which is related to some definitions of relevance (see Definition 1.22). Therefore, the proposed heuristic suggests that the class variable C should be the root of the graph $\boldsymbol{s}$ and it should be the parent of all relevant variables X_i, $C \in \boldsymbol{Pa}(\boldsymbol{s})_i$, for all i (see Figure 2.3(a)).

Both intuitions lead to the *augmented naïve Bayes* family of structures. As we will note, in order to control the number of parameters and for reducing the search space, the augmented naïve Bayes family imposes restrictions to the maximum number of parents for each feature. From here on, we will refer to the set of parents of each feature X_i minus the class variable C as $\boldsymbol{Px}_i = \boldsymbol{Pa}_i \setminus C$.

Henceforth, we present a set of subfamilies of structures included in the augmented naïve Bayes family, ranging from the simplest family of structures (i.e. with the most restrictive conditional independencies and the minimum number of parameters) to the most complex family. Each subfamily of structures imposes different constraints to the maximum number of feature parents



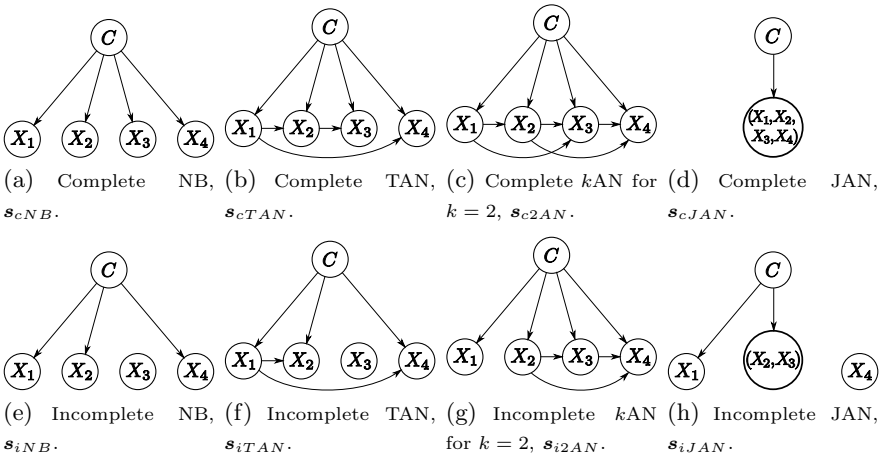
(a) The structure $\boldsymbol{s}_{few}$ induces the independence assertions $\{CI(X_i;(X_1,...,X_{i-1})|C)\}_{i=1}^n$ among others. Due to these useful conditional independence assertions the factorization is greatly simplified: $p(\boldsymbol{x},c|\boldsymbol{s}_{few}) = p(c)\prod_{i=1}^n p(x_i|x_1,...,x_{i-1},c|\boldsymbol{s}_{few}) = p(c)\prod_{i=1}^n p(x_i|c)$. In order to specify the model, the number of parameters required is $(r_C-1)+\sum_{i=1}^n(r_i-1)r_C$. It must be noted that the number of parameters is linear with respect to n. As we will note this structure is named naïve Bayes and it leads to the most popular structures of classifiers based on Bayesian networks: The family of augmented naïve Bayes structures.



(b) The structure $\boldsymbol{s}_{many}$ has the independence assertions $\{CI(X_i;(X_1,...,X_{i-1})|\emptyset)\}_{i=1}^n$, but it does not induce the conditional independence assertions $\{CI(X_i;(X_1,...,X_{i-1})|C)\}_{i=1}^n$. The factorization can be simplified due to the included independence assertions as follows: $p(\boldsymbol{x},c|\boldsymbol{s}_{many}) = p(c|x_1,...,x_n,\boldsymbol{s}_{many})\prod_{i=1}^n p(x_i|x_i,...,x_n,\boldsymbol{s}_{many}) = p(c|x_1,...,x_n)\prod_{i=1}^n p(x_i)$. In order to specify the model, the number of parameters required is $\sum_{i=1}^n(r_i-1)+(r_C-1)\prod_{i=1}^n r_i$. It must be noted that the number of parameters is exponential with respect to n, which is comparable to the number of parameters required to model a complete graph.

Fig. 2.3. These figures represent two intuitive approaches of structures biased towards classification, $\boldsymbol{s}_{few}$ and $\boldsymbol{s}_{many}$. Both structures include the conditional dependence assertion $CD(X_i;C|\boldsymbol{X}_s)$ and $CD(X_i;C|\emptyset)$ for $i=1,...,n$ and for any $\boldsymbol{X}_s \in \boldsymbol{X}_{\neg i} = (X_1,...,X_{i-1},X_{i+1},...,X_n)$. But we prefer $\boldsymbol{s}_{few}$ because it requires fewer parameters.

allowed, $|\boldsymbol{Px}_i| < max$, for all i. Examples of each subfamily are shown in Figure 2.4. Note that all the structures shown in the figure are constrained to graphs with the class variable as the root and $(C,X_1,...,X_4)$ is an ancestral ordering. At each subfamily of structures we consider two types of structures: complete and incomplete. If some arcs can be added to a structure without breaking the structural constraints imposed by its subfamily of structures,



(a) Complete NB, $\boldsymbol{s}_{cNB}$. (b) Complete TAN, $\boldsymbol{s}_{cTAN}$. (c) Complete kAN for $k = 2$, $\boldsymbol{s}_{c2AN}$. (d) Complete JAN, $\boldsymbol{s}_{cJAN}$.

(e) Incomplete NB, $\boldsymbol{s}_{iNB}$. (f) Incomplete TAN, $\boldsymbol{s}_{iTAN}$. (g) Incomplete kAN for $k = 2$, $\boldsymbol{s}_{i2AN}$. (h) Incomplete JAN, $\boldsymbol{s}_{iJAN}$.

Fig. 2.4. Different structure complexities taken from the augmented naïve Bayes family of structures.

the structure is said to be incomplete. Otherwise, the structure is said to be complete.

The simplest classifier structure of augmented naïve Bayes family of structures is the *naïve Bayes structure* (*NB structure*) [Minsky (1961); Duda and Hart (1973); Langley et al. (1992)]. In NB structures the number of parents minus the class for each feature is zero, $|\boldsymbol{Px}_i| = 0$, for all i. In other words, NB structure assumes that the predictors are conditionally independent given the class, $CI(X_i; \boldsymbol{X}_{\neg i}|C)$ for all i. Figures 2.4(a) and 2.4(e) show a complete and incomplete NB structures, $\boldsymbol{s}_{cNB}$ and $\boldsymbol{s}_{iNB}$, respectively. The factorization of the structures $\boldsymbol{s}_{cNB}$ and $\boldsymbol{s}_{iNB}$ is $p(\boldsymbol{x}, c|\boldsymbol{s}_{cNB}) = p(c)p(x_1|c)p(x_2|c)p(x_3|c)p(x_4|c)$ and $p(\boldsymbol{x}, c|\boldsymbol{s}_{iNB}) = p(c)p(x_1|c)p(x_2|c)p(x_3)$ $p(x_4|c)$. Note that since $CI(X_3; (C, X_1, X_2, X_4)|\emptyset)$ is inferred from $\boldsymbol{s}_{iNB}$, the value of X_3 does not affect the classification rule associated to $p(c|x_1, ..., x_4,$ $\boldsymbol{s}_{iNB})$. In the literature, incomplete NB structures are referred to as selective naïve Bayes structures. In spite of the strong conditional independence assumption made by a NB structure, its performance is surprisingly good, even in databases which do not hold with the independence assumption [Domingos and Pazzani (1997)]. The good performance of the NB structures has motivated the research of structures which relax this strong independence assumption. The number of parameters required of a complete NB structure, such us $\boldsymbol{s}_{cNB}$ in Figure 2.4(e), is $(r_C - 1) + \prod_{i=1}^{n}(r_i - 1)r_C$. Assuming that both the features and the class variable take r different values, the number of parameters for modeling a complete NB structure in the BMN paradigm is $r - 1 + n(r - 1)r$, $\mathcal{O}(nr^2)$. The number of parameters is linear with respect to the number of (included) features, n, and it tends to be quadratic with respect to the cardinality of the random variables r.

The *tree-augmented naïve Bayes structures* (*TAN structures*) break with the strong independence assumption made by the NB structures, allowing probabilistic dependencies among predictors. The TAN structures consist of graphs with arcs from the class variable to the subset of relevant predictors, and with arcs between the relevant predictors taking into account that the maximum number of parents of a variable is one plus the class variable, $|\boldsymbol{Pa}_i| \leq 1$ for all i. The TAN structures take into account relationships between predictive variables by extending the naïve Bayes structure with a tree structure among the predictive variables. In other words, it breaks with the strong conditional independence assertions $\{CI(X_i; \boldsymbol{X}_{\neg i}|C)\}_{i=1}^{n}$ by including the novel conditional dependence assertions $\{CD(X_i; \boldsymbol{U}|\boldsymbol{V})\}_{i=1}^{n}$ for any non-empty $\boldsymbol{U} \in \boldsymbol{Pa}_i$ and any possibly empty $\boldsymbol{V} \in (\boldsymbol{X}, C)$ where $\boldsymbol{U} \cap \boldsymbol{V} = \emptyset$ and $|\boldsymbol{Pa}_i| \leq 1$. It should be noted that we consider the NB structures as a particular case of the TAN structures when $|\boldsymbol{Pa}_i| = 0$. Figures 2.4(b) and 2.4(f) show a complete and an incomplete TAN structure, $\boldsymbol{s}_{cTAN}$ and $\boldsymbol{s}_{iTAN}$, respectively. The factorization of the structures $\boldsymbol{s}_{cTAN}$ and $\boldsymbol{s}_{iTAN}$ is $p(\boldsymbol{x}, c|\boldsymbol{s}_{cTAN}) = p(c)p(x_1|c)p(x_2|x_1, c)p(x_3|x_1, c)p(x_4|x_1, c)$ and $p(\boldsymbol{x}, c|\boldsymbol{s}_{iTAN}) = p(c)p(x_1|c)p(x_2|x_1, c)p(x_3)p(x_4|x_1, c)$. Assuming that $\boldsymbol{Pa}_i = X_{1:i}$ where $X_{1:i}$ is the first parent of X_i with cardinality $r_{1:i}$ and $\boldsymbol{Pa}_1 = \emptyset$, without loss of generalization, the number of required parameters to model a complete TAN structure is $(r_C - 1) + (r_1 - 1)r_C \sum_{i=2}(r_i - 1)r_{1:i}r_C$. On the other hand, assuming that the n features and the class variable have r states, the number of parameters for modeling a complete TAN structure in the BMN paradigm is $r-1+(r-1)r+(n-1)(r-1)r^2$, $\mathcal{O}(nr^3)$. The number of parameters is linear with respect to the number of (included) features, n, and it tends to be cubic with respect to the cardinality of the random variables r.

The *k-augmented naïve Bayes structures* (*kAN structures*) extend TAN structures allowing a maximum of k predictor parents plus the class for each predictor variable, $|\boldsymbol{Pa}_i| \leq k$ for all i. In the literature these structures are usually referred to as k-dependence Bayesian classifier structure. It must be highlighted that kAN structures can be regarded as a spectrum of allowable dependence in the family of augmented naïve Bayes structures with the NB structure ($k = 0$) at the most restrictive extreme and the full graph ($k \geq n - 1$) at the most general one. Figures 2.4(c) and 2.4(g) show a complete and an incomplete kAN structures with $k = 2$, $\boldsymbol{s}_{c2AN}$ and $\boldsymbol{s}_{i2AN}$, respectively. The factorization of the structures $\boldsymbol{s}_{c2AN}$ and $\boldsymbol{s}_{i2AN}$ is $p(\boldsymbol{x}, c|\boldsymbol{s}_{c2AN}) = p(c)p(x_1|c)p(x_2|x_1, c)p(x_3|x_1, x_2, c)p(x_4|x_2, x_3, c)$ and $p(\boldsymbol{x}, c|\boldsymbol{s}_{iTAN}) = p(c)$ $p(x_1|c)p(x_2|c)p(x_3|x_2, c)p(x_4|x_2, x_3, c)$. The number of parameters required for specifying a model in the BMN paradigm based on a complete kAN structure, such us $\boldsymbol{s}_{c2AN}$ for $k = 2$, based on the ancestral order $(C, X_1, ..., X_n)$ is $(r_C - 1) + (r_1 - 1)r_C \sum_i = 2(r_i - 1)q_i$ where $q_i = r_C \prod_{X_j \in \boldsymbol{Pa}_i} r_j$ and $|\boldsymbol{Pa}_i| = \min(i - 1, k)$. Assuming that the n features and the class variable have r states, the number of parameters for modeling a complete kAN structure ($k < n$) in the BMN paradigm is $\sum_{i=1}^{k-1}(r - 1)r^i + (n - k + 1)(r - 1)r^k$,

$\mathcal{O}(nr^{k+1})$. Note that this expression summarizes the expressions of NB and TAN structures for $k = 0$ and $k = 1$, respectively.

Finally, the *joint augmented naïve Bayes structure* (*JAN structure*) includes nodes $\boldsymbol{Y}_i$ that represents multidimensional random variables $\boldsymbol{X} = (X_1, ..., X_n) = (\boldsymbol{Y_1}, ..., \boldsymbol{Y_m})$ with $m \leq n$ and $\boldsymbol{Y}_i \cap \boldsymbol{Y}_j = \emptyset$. We call these nodes $\boldsymbol{Y}_i$ *joint nodes* in order to distinguish them from the univariate nodes. A joint node $\boldsymbol{Y} = (X_{1:\boldsymbol{Y}}, ..., X_{l:\boldsymbol{Y}})$ represents a full subgraph in $\boldsymbol{Y}$. In the literature these structures are usually referred to as semi naïve Bayes structures [Pazzani (1997)]. JAN structure is a NB structure defined over joint nodes, i.e. the joint nodes are conditional independent given the class, $CI(\boldsymbol{Y}_i; \boldsymbol{Y}_j|C)$ for all $i \neq j$. The JAN structures extend the NB structures allowing the following conditional dependencies between predictors X_i belonging to the same joint node $\boldsymbol{Y}_j$, $X_i, X_j \in \boldsymbol{Y}_j$: $CD(X_i; X_j|\boldsymbol{U})$ for any $\boldsymbol{U} \subseteq \{\boldsymbol{X}, C\} \setminus \{X_i, X_j\}$. Figures 2.4(d) and 2.4(h) show a complete and an incomplete JAN structure, $\boldsymbol{s}_{cJAN}$ and $\boldsymbol{s}_{iJAN}$, respectively. The factorization of the structures $\boldsymbol{s}_{cJAN}$ and $\boldsymbol{s}_{iJAN}$ is $p(\boldsymbol{x}, c|\boldsymbol{s}_{cJAN}) = p(c)p(x_1, x_2, x_3, x_4|c)$ and $p(\boldsymbol{x}, c|\boldsymbol{s}_{iJAN}) = p(c)p(x_1|c)p(x_2, x_3|c)p(x_4)$, respectively. The number of parameters of a complete JAN structure, such us $\boldsymbol{s}_{cJAN}$, in the BMN paradigm constraining the maximum number of variables in a joint node $\boldsymbol{Y}_i = (X_{1:i}, ..., X_{|\boldsymbol{Y}_i|:i})$ to k, $|\boldsymbol{Y}_i| \leq k$, for all i is given by $(r_C - 1) + \sum_{i=1}^{m} (r_{\boldsymbol{Y}_i} - 1) r_C$ where $m = \lfloor n/k \rfloor$ and $r_{\boldsymbol{Y}_i} = \prod_{j=1}^{|\boldsymbol{Y}_i|} r_{j:i}$. Assuming that the n features and the class variable have r states, the number of parameters required for modeling a complete JAN structure with a maximum of k variables at each joint node is $(r-1) + \sum_{i=1}^{m} r^{|\boldsymbol{Y}_i|+1}$, $\mathcal{O}(r^{k+1})$.

2.5.3 Classifier induction algorithms based on BMN

In the following subsections, for each kind of structure previously introduced, a set of wrapper and filter classifier induction algorithms based on the BMN paradigm are presented. This section is mainly focused on the structural learning algorithms, specially those which will be adapted to the conditional Gaussian network and kernel based Bayesian network paradigms in Sections 3 and 4, respectively. In most of the classifier induction algorithms presented in this section the parametric learning of the Bayesian multinomial network is performed using the maximum likelihood estimator. An extensive review of classifier induction algorithms based on BMN with the maximum likelihood estimator is presented in [Blanco (2005)].

2.5.3.1 Multinomial naïve Bayes

The *naïve Bayes* classifier (*NB*) [Duda and Hart (1973); Langley et al. (1992); Minsky (1961)] is characterized by the conditional independence assumption between variables given the class. The NB classifier is a BMN with a complete naïve Bayes structure and the parameters obtained by maximum likelihood

estimation. We call multinomial NB (MNB) to the NB classifier based on BMN. Although MNB was previously used in statistics and pattern recognition [Duda and Hart (1973)], as far as we know, the first time it appears in supervised classification is in Cestnik et al. (1987). In the literature, the NB paradigm is designated with several names: idiot Bayes [Ohmann et al. (1988)], naïve Bayes [Kononenko (1990)], simple Bayes [Gammerman and Thatcher (1991); Domingos and Pazzani (1997)] or independent Bayes [Todd and Stamper (1994)]. Moreover, all the variables are included in the model, so the classifier structure is given *a priori*: *complete NB structure* (see for example Figure 2.4(a)). The accuracy obtained with this classifier based on BMN paradigm is surprisingly high in some domains, even in data sets that do not obey the strong conditional independence assumption [Domingos and Pazzani (1997)]. For instance, there are successful applications of naïve Bayes classifiers in medical domains [Kononenko (1990); Ohmann et al. (1996); Mani et al. (1997); Movellan et al. (2002)], in web site classification according to user interest [Pazzani et al. (1996)], in collaborative filter approaches [Miyahara and Pazzani (2000)], text classification [McCallum and Nigam (1998)] or failure detection [Hamerly and Elkan (2001)]. For more details about naïve Bayes, the reader may be interested in the historical review and improvements of the naïve Bayes classifier presented in [Larrañaga (2003)] and [Blanco (2005)].

Despite the good performance of MNB even when the assumption of conditional independence of the variables is not fulfilled, highly correlated features could hurt its accuracy in practice (see Section 1.11) [Langley and Sage (1994)]. The selective naïve Bayes [Langley and Sage (1994)] is a variant of the naïve Bayes classifier that uses only a subset of the variables to make predictions $\boldsymbol{X}_r \subseteq \boldsymbol{X}$, i.e. incomplete naïve Bayes structure. In fact, the selective naïve Bayes classifier can be seen as a naïve Bayes classifier for which a feature subset selection process has been performed. We call selective multinomial NB (sMNB) to the selective NB classifier based on BMN paradigm.

As the search space has 2^n structures, i.e. the number of different subsets of predictors $\boldsymbol{X}_r \subseteq \boldsymbol{X}$, an exhaustive search of the space is not practical. Hence, an alternative is to perform a heuristic search. In general, any feature subset selection method can be used to obtain the subset of relevant and non-redundant variables for classification purposes. In the literature two main approaches are proposed. On the one hand, filter scores proposed are the entropy, the mutual information, the Euclidean distance or the Kullback-Leibler divergence [Ben-Bassat (1982); Doak (1992); Inza et al. (2004); Blanco (2005)]. On the other hand, wrapper methods use model-based scores such as accuracy to find the set of relevant variables for classification. These latter methods usually use optimization heuristics such as hill-climbing [Langley and Sage (1994); Inza et al. (2002)], simulated annealing [Vinciotti and Liu (2006)], genetic algorithms [Inza et al. (2001a,b)] or estimation of distribution algorithms [Inza et al. (2000, 2001b); Blanco et al. (2003)], among others. By contrast, Dash and Cooper (2002); Cerquides and de Mántaras (2003); Dash and Cooper (2004) propose a Bayesian approach by averaging over all the

selective naïve Bayes models in order to obtain a single naïve Bayes model which, despite including all the predictive variables in the model, is able to capture the relevance of the variables for classification purposes. Besides, contrary to the intuition, the average of NB models in general breaks with the conditional independence assumption made by a single naïve Bayes classifier.

As we noted before, Langley and Sage (1994) proposes a wrapper feature selection process where a greedy forward algorithm is used. The process starts with an empty set of variables and, at each step, the variable which most increases the accuracy measured in a leaving-one-out cross-validation is added to the model. We call this algorithm *wrapper selective multinomial naïve Bayes* (*wsMNB*) and we will adapt it to conditional Gaussian network paradigm in Chapter 3. Algorithm 2 shows the pseudo-code of the wsMNB classifier induction algorithm [Langley and Sage (1994)].

Algorithm 2: **Wrapper multinomial selective naïve Bayes.**

1 Let the set of selected variables be empty, $\boldsymbol{X}_s = \emptyset$.
2 **while** true **do** {
3 Construct the MNB models $MNB(X_i \cup \boldsymbol{X}_s)$ for each X_i, where $X_i \in \boldsymbol{X} \setminus \boldsymbol{X}_s$.
4 Estimate using a cross validation the error of the constructed models $MNB(X_i \cup \boldsymbol{X}_s)$, $\epsilon(MNB(X_i \cup \boldsymbol{X}_s))$, for all i.
5 Select the variable X_{min} with the associated lowest error, $\epsilon(MNB(X_{min} \cup \boldsymbol{X}_s))$.
6 **if** $\epsilon(MNB(X_{min} \cup \boldsymbol{X}_s)) < \epsilon(MNB(\boldsymbol{X}_s))$.
7 **then** Include the variable X_{min} in the selected set $\boldsymbol{X}_s$.
8 **else** Return the classifier $MNB(\boldsymbol{X}_s)$.

On the other hand, Blanco (2005) proposes a filter algorithm for the induction of selective naïve Bayes classifiers. We name this approach *filter selective multinomial naïve Bayes* (*fsMNB*). This algorithm obtains (in)complete NB structures based on the *mutual information* between the predictor variables and the class (see Section 1.10) and the hypothesis test explained in Section 1.11.1.3, Paragraph A. [Kullback (1959); Pardo (1997)]. The fsMNB algorithm finds the set of features which are considered correlated with the class variable using the statistical test [Kullback (1959); Pardo (1997)] at a certain significance level, α. The pseudo-code of the fsMNB can be seen in Algorithm 3. We propose an adaptation of this algorithm to the conditional Gaussian paradigm in Chapter 3 where the threshold to the mutual information is heuristically fixed.

Algorithm 3: **Filter multinomial selective naïve Bayes.**

1 Initialize the set of selected variables to empty, $\boldsymbol{X}_r = \emptyset$.
2 Compute the mutual information $I(X_i; C)$ for all $i = 1, \ldots, n$.
3 **for** $X_i \in \boldsymbol{X} = (X_1, ..., X_n)$ **to**
4 Compute $I(X_i, C)$.
5 Given a test size α compute the threshold $\tau_i = \chi^2_{(r_C-1)(r_i-1),\alpha}$.
6 **if** $\tau_i \leq I(X_i; C)$.
7 **then** Add X_i in the selected features $\boldsymbol{X}_r$.
8 Return the classifier associated with the selected features, $MNB(\boldsymbol{X}_r)$.

2.5.3.2 Multinomial tree-augmented naïve Bayes

As we noted in Section 2.5.2 TAN structures break with the strong independence assumption of NB structures, i.e. conditional independence of the features given the class, $CI(X_i; X_j|C)$ for all $i \neq j$. This section is focused on learning algorithms which obtain Bayesian multinomial classifiers with TAN structures. We will introduce in detail the two classifier induction algorithms that will be adapted to the conditional Gaussian network and the kernel based Bayesian network paradigms in Chapters 3 and 4, respectively.

The first classifier induction algorithm proposed for TAN structures [Friedman et al. (1997)] is a filter approach based on conditioned mutual information, and we call it *filter multinomial tree-augmented naïve Bayes* (*fMTAN*). The structural learning procedure proposed by Friedman et al. (1997) constructs a tree between the features taking into account the special role of the class variable, which is the parent of all features, $X_i \in \boldsymbol{Ch}_C$ for all i. The algorithm adapts the procedure proposed by Chow and Liu (1968) using the class conditional mutual information instead of the mutual information. Each conditioned mutual information $I(X_i; X_j|C)$, which measures the strength of the conditional dependence assertion $CD(X_i; X_j|C)$, is used to weight the undirected edge between each pair X_i and X_j. Then, the Kruskal algorithm is used to obtain the maximum weighted spanning tree. The algorithm needs the calculation of $n(n-1)/2$ ($\mathcal{O}(n^2)$) conditional mutual information quantities. Algorithm 4 shows the pseudo-code for the fMTAN algorithm and Algorithm 5 shows the pseudo-code for the Kruskal algorithm.

It should be highlighted that fMTAN preserves the computational cost of the algorithm of Chow-Liu, requiring a polynomial time in the number of variables [Chow and Liu (1968)]. The Chow-Liu algorithm obtains a tree structure that maximizes the likelihood. The fMTAN algorithm inherits this important property and it obtains a complete TAN structure which maximizes the likelihood of the model given the data [Friedman et al. (1997)]. Moreover, fMTAN is asymptotically correct if the data have been generated from a TAN structure. Two aspects must be taken into account. First, the

Algorithm 4: **Filter multinomial tree-augmented naïve Bayes.**

1 Compute $I(X_i; X_j|C)$ for all $i < j$.
2 Let $\boldsymbol{s}$ be a complete undirected graph where $I(X_i, X_j|C)$ is the weight of edge (X_i, X_j).
3 Use the Kruskal algorithm to obtain the maximum spanning tree from $\boldsymbol{s}$.
4 Select randomly a node as the root to set the direction of the edges.
5 Add the class variable as parent of each predictive variable.
6 Learn the parameters of the associated Bayesian multinomial network using the maximum likelihood estimator.

Algorithm 5: **Kruskal algorithm.**

1 Let $\boldsymbol{s} = (\boldsymbol{X}, \boldsymbol{E})$ be an empty graph, $|\boldsymbol{E}| = 0$, with n nodes, $\boldsymbol{X} = (X_1, ..., X_n)$.
2 **While** $|\boldsymbol{E}| < n - 1$ **do**
3 Add to $\boldsymbol{s}$ the edge with the highest weight which does not create a cycle in $\boldsymbol{s}$.

structural likelihood maximization does not necessarily imply a predictive error minimization. Second, the fMTAN constructs a complete TAN structure. Thus, some redundant variables and irrelevant arcs could be added. An alternative for avoiding the compulsory induction of complete TAN structures, and following the intuitions behind fsMNB, consist of imposing a threshold to the mutual information (see Blanco (2005) for further details).

The wrapper alternative to the fMTAN algorithm was proposed by Keogh and Pazzani (1999). More than a direct attempt to approximate the underlying probability distribution, they solely concentrate on using the same representation to improve the estimated classification accuracy. As the space of possible structures is exponential with the number of variables, generally, the wrapper optimization is approximate. For example Keogh and Pazzani (1999) performs a forward greedy search algorithm in the space of allowed structures guided by the estimated accuracy. They propose a greedy approach to obtain a TAN structure. The algorithm starts with a complete NB structure and the arcs that maximize the proposed score, e.g. accuracy, are successively added if they agree with TAN structural restrictions. For each arc added to the network, $\mathcal{O}(n^2)$ classifier structures are considered and evaluated, where n is the number of predicted variables. In order to obtain a TAN structure, $\mathcal{O}(n)$ arcs can be added. Hence, the time complexity of the algorithm is $\mathcal{O}(n^3)$. We propose a modification to this algorithm called *wrapper selective multinomial tree-augmented naïve Bayes* (*wsMTAN*), which starts from an empty structure without arcs. The pseudo-code of wsMTAN is shown in Algorithm 6. The wsMTAN algorithm should avoid the disadvantages of fMTAN mentioned at the end of the previous paragraph, because it is not restricted to complete TAN structures and because it performs an implicit variable selection.

Algorithm 6: **Wrapper selective multinomial tree-augmented naïve Bayes.**

1 Let the structure $\boldsymbol{s} = (\boldsymbol{V}, \boldsymbol{E})$ with $\boldsymbol{E} = \emptyset$ and $\boldsymbol{V} = (\boldsymbol{X}, \boldsymbol{C})$.
2 **Repeat** until non-improvement is reached.
3 Select the best structure $\boldsymbol{s}_{TAN}$, i.e. with the best estimated score, among the following alternatives:
4 (a) Consider the classifiers with the structure $\boldsymbol{s}' = (\boldsymbol{V}, \boldsymbol{E}')$ with $\boldsymbol{E}' = \boldsymbol{E} \cup (C, X_i)$ where $X_i \notin \boldsymbol{Ch}(\boldsymbol{s})_C$.
5 (b) Consider the classifiers with the structure $\boldsymbol{s}' = (\boldsymbol{V}, \boldsymbol{E}')$ with $\boldsymbol{E}' = \boldsymbol{E} \cup (X_i, X_j)$ where $X_i, X_j \subseteq \boldsymbol{Ch}(\boldsymbol{s})_C$ and $\boldsymbol{Pa}(\boldsymbol{s})_j \leq 2$.
6 **If** the classifier based on BMN associated to $\boldsymbol{s}_TAN$ improves the classifier associated to $\boldsymbol{s}$.
7 **Then** Set $\boldsymbol{s}$ to $\boldsymbol{s}_{TAN}$.
8 **Else** Return the structure $\boldsymbol{s}$.

We adapt this algorithm to conditional Gaussian network paradigm in Chapter 3 [Pérez et al. (2006b)]. wsMTAN performs an implicit feature selection following the intuitions behind wsMNB. It must be noted that the algorithms fMTAN and wsMTAN perform a forward greedy structure inclusion of arcs, and this can be problematic for detecting and modeling XOR-like relations between more than two features.

Another alternative is proposed in Pernkopf and Bilmes (2005) where the algorithm proposed by Friedman et al. (1997) is modified in order to perform a discriminative learning of TAN structures by leading the structural learning with the 3-way interaction information $I(X_i; X_j; C)$ instead of the conditioned mutual information $I(X_i; X_j|C)$. As we will discuss in Chapter 5 this algorithm is an approximation to the TAN structure that maximizes the conditional likelihood instead of the joint likelihood.

Moreover, there are some Bayesian approaches for TAN induction. Dash and Cooper (2003); Cerquides and de Mántaras (2003); Dash and Cooper (2004); Cerquides and de Mántaras (2005) introduce different tractable approaches to average over all the possible TAN structures given an ancestral order over the predictive variables. However, although the Bayesian model averaging process is performed over TAN models, the resultant model is a complete Bayesian network. Nevertheless, Dash and Cooper (2004) propose some restrictions to make the calculations more efficient.

2.5.3.3 Multinomial k-dependent augmented naïve Bayes

As we noted in Section 2.5.2, kDB structures can be regarded as a spectrum of allowable dependence in augmented naïve Bayes family of structures with the NB structure at the most restrictive extreme ($k = 0$) and the full BMN at the most general one ($k \geq n - 1$). This section is focused on classifier induction algorithms based on kDB structures for the BMN paradigm. We will introduce in detail a classifier induction algorithm based on kDB structures that we will

adapt to the conditional Gaussian network and the kernel based Bayesian network paradigms in Chapters 3 and 4, respectively.

The kDB structure allows each predictor X_i to have not more than k predictor variables as parents. This constraint allows to control the number of parameters required by the structures as noted in Section 2.5.2. Besides, this constraint reduces the search space with respect to the entire set of structures.

We start by presenting the classifier induction algorithm proposed by Sahami (1996) which we call *filter multinomial k-dependent augmented naïve Bayes* (*fMkAN*). fMkAN is a generalization of the algorithm fMTAN [Friedman et al. (1997)] allowing a maximum of k parents plus the class for each feature. The fm$k$AN algorithm is a filter forward greedy approach which uses the class conditional mutual information between variables $I(X_i; X_j \mid C)$ and the mutual information $I(X_i; C)$ between class and variables to lead the structure search process. First, $I(X_i; C)(i = 1, ..., n)$ and $I(X_i; X_j \mid C)(i = 1, ..., n)(j = i, ..., n)$ are computed. The fMkAN algorithm starts from a structure with only the class variable. At each step, from the subset of non-included predictor variables, the variable X_{max} with the highest $I(X_i; C)$ is added. Next, arcs from the variables included in the structure to variable X_{max} are added while it is possible, as long as the maximum number of parents k is not surpassed. The arcs are added following the order of $I(X_{max}; X_j \mid C)$ from the greatest value to the smallest one. The algorithm continues until a *complete kAN structure* is obtained. Thus, the redundant variables and several irrelevant relations between variables are also inevitably added. Therefore, the fMkDB could perform worse in data sets with redundant variables. It is advisable to perform a feature subset selection in order to remove the irrelevant and redundant variables. The pseudo-code of fMkAN algorithm is shown in Figure 7.

Algorithm 7: **Filter multinomial k-dependent augmented naïve Bayes (fMkAN)**

1 For each feature $X_i \in \boldsymbol{X} = (X_1, ..., X_n)$ compute the mutual information $I(X_i; C)$.

2 Compute the conditional mutual information $I(X_i; X_j|C)$, for each pair of features, where $i \leq j$.

3 Let $\boldsymbol{s} = (\boldsymbol{V}, \boldsymbol{E})$ be an empty graph $|\boldsymbol{E}| = 0$ with $\boldsymbol{V} = (\boldsymbol{X}, C)$.

4 **for** $i = 1$ **to** n

5 Select the predictor X_{max} which is not included in $\boldsymbol{Ch}(\boldsymbol{s})_C$ and has the largest mutual information with the class $I(X_{max}; C)$.

6 Add the directed arc (C, X_{max}) to $\boldsymbol{s}$.

7 Add $m = min(i - 1, k)$ arcs to $\boldsymbol{s}$ from m distinct features X_j in $\boldsymbol{Ch}(\boldsymbol{s})_C$ with the highest class conditional mutual information $I(X_j; X_{max}|C)$.

8 Learn the parameters of the BMN with the structure $\boldsymbol{s}$ using the maximum likelihood estimator and return the classifier.

This algorithm induces complete kAN structures, similarly to fMTAN which induces complete TAN structures. Following the intuitions behind fsMNB, some arcs can be forbidden based on a threshold or a set of thresholds.

The wrapper alternative follows the intuitions behind wsMTAN allowing the induction of incomplete kAN structures. We call this algorithm *wrapper selective multinomial k-dependence augmented naïve Bayes* (*wsMkAN*). The pseudo-code of wsMkAN is shown in Algorithm 8.

Algorithm 8: **Wrapper multinomial selective k-dependence augmented naïve Bayes.**

1 Let the structure $\boldsymbol{s} = (\boldsymbol{V}, \boldsymbol{E})$ with $\boldsymbol{E} = \emptyset$ and $\boldsymbol{V} = (\boldsymbol{X}, \boldsymbol{C})$.
2 } **while** true
3 Select the best structure $\boldsymbol{s}^*$, i.e. with the best estimated wrapper score, among the following alternatives:
4 (a) Consider the classifiers based in BMN with the structure $\boldsymbol{s}' = (\boldsymbol{V}, \boldsymbol{E}')$ with $\boldsymbol{E}' = \boldsymbol{E} \cup (C, X_i)$ where $X_i \notin \boldsymbol{Ch}(\boldsymbol{s})_C$.
5 (b) Consider the classifiers based in BMN with the structure $\boldsymbol{s}' = (\boldsymbol{V}, \boldsymbol{E}')$ with $\boldsymbol{E}' = \boldsymbol{E} \cup (X_i, X_j)$ where $\{X_i, X_j\} \subseteq \boldsymbol{Ch}(\boldsymbol{s})_C$ and $\boldsymbol{Pa}(\boldsymbol{s})_j \leq k + 1$.
6 **If** structure $\boldsymbol{s}^*$ improves $\boldsymbol{s}$
7 **Then** Set $\boldsymbol{s}$ to $\boldsymbol{s}^*$.
8 **Else** Return the structure $\boldsymbol{s}$.

Note that fMkAN, and wsMkAN perform a forward greedy inclusion of arcs and, thus, they could have problems for detecting and modeling XOR-like relations between more than two predictors.

2.5.3.4 Multinomial joint augmented naïve Bayes

An alternative to the NB structures which also break their strong class conditional assumption are the JAN structures [Kononenko (1991); Pazzani (1997); Zheng et al. (1999)]. For this purpose, they introduce the joint nodes $\boldsymbol{Y} = (X_{1:\boldsymbol{Y}}, ..., X_{l:\boldsymbol{Y}})$. A joint variable $\boldsymbol{Y} = (X_{1:\boldsymbol{Y}}, ..., Y_{l:\boldsymbol{Y}})$ represents a multidimensional random variable and it can be interpreted as a complete subgraph in $\boldsymbol{Y}$. As a JAN structure considers independent joint variables, the factorization of a JAN structure is very similar to NB structure factorization as it is illustrated in Section 2.5.2. The number of parameters required to model the distribution of a joint variable conditioned to the class, $p(\boldsymbol{y} = (y_{1:\boldsymbol{Y}}, ..., y_{l:\boldsymbol{Y}})|c)$, based on BMN paradigm is $r_C((\prod_{i=1}^{l} r_{i:\boldsymbol{Y}}) - 1)$ which is exponential with respect to l. Thus, we should limit the maximum number of component nodes in the joint nodes in order to control the number of parameters and the variance of the error of the classifiers based on JAN structures.

Depending on the direction of the greedy search process (forward and backward), Pazzani (1997) presents two wrapper ways to detect dependencies among variables. We name both algorithms *forward wrapper multinomial joint augmented naïve Bayes* (*fwMJAN*) and *backward wrapper multinomial joint augmented naïve Bayes* (*bwMJAN*). The fwMJAN algorithm starts from a structure containing the class node C. It considers two operators to carry out the search in the space of possible structures:

1. Add a variable not used by the current classifier as a new variable. The added variable is class conditioned and conditionally independent given the class with respect to the other variables used in the current classifier.
2. Join a variable not used by the current classifier to a variable currently used by it.

At each step in the structural learning process, a set of candidate structures is considered. The set consists of all structures that can be inferred from the actual one, applying one of the operators previously introduced once. Each structure contemplated is evaluated by means of estimated accuracy. Afterwards, the best candidate is chosen. If the best option does not improve the accuracy, the current classifier structure is returned.

The bwMJAN is similar to fwMJAN except in that bwMJAN starts from a *complete NB structure*, and, at each step, it considers two different operators:

1. Remove a variable used by the current classifier.
2. Join a variable used by the current classifier to another variable currently used by it.

This algorithm also considers the best option. According to Pazzani (1997), the backward search performs better than the forward search with multinomial variables.

In both algorithms, for each change in the network using the mentioned operators, $\mathcal{O}(n^2)$ classifier structures are considered and evaluated. Besides, in the worst case, $\mathcal{O}(n)$ changes could be made. Thus, in the worst case, the time complexity for both algorithms is $\mathcal{O}(n^3)$.

Some filter approaches for the induction of JAN structures based on BMN are presented in Blanco (2005).

2.6 Bayesian multinomial networks and continuous features

A classifier based on PGMs is determined by the structure of the graph and the probability distributions and probability density functions which model the (in)dependence relations between the variables. In order to model a density function of a continuous variable X, for a structure $\boldsymbol{s}$, four approaches are generally considered:

1. To discretize the continuous variable X and to estimate the local distribution $p(x'|\boldsymbol{pa}(\boldsymbol{s})_{X'})$ of the discretized variable X' by means of a multinomial probability distribution.
2. To directly estimate the local density function $f(x|\boldsymbol{pa}(\boldsymbol{s})_x)$ in a parametric way, e.g. by means of the mixed Gaussian distribution [Lauritzen (1996)] (see Chapter 3 for further details).
3. To directly estimate the local probability density function in a non-parametric way, e.g. using the kernel based density functions [Silverman (1986)] (see Chapter 4 for further details).
4. To directly estimate the local probability density functions in a semi-parametric way, e.g. by means of finite mixture models [Figueiredo et al. (1999)].

The BMN paradigm only handles discrete variables and if a continuous variable is present, it must be discretized with the consequent loss of information [Yang and Webb (2003)]. The loss of information due to the discretization is illustrated in Chapter 4 using four continuous artificial domains. In spite of this, the discretization plus multinomial distribution approach have shown competitive performances with respect to the other alternatives [Cheng and Greiner (1999); Pérez et al. (2006b, 2009)]. Clearly, an improvement in the procedures used to estimate the local probability density functions does not imply an improvement in the classification performance. On the other hand, the discretization plus multinomial approach could have some problems when modeling a graph with a complex structure and/or with variables discretized in many intervals because the number of parameters to be estimated can be very high (see Section 2.5.2).

There are alternatives to BMN among the PGMs which can handle directly continuous attributes: *conditional Gaussian networks* [Lauritzen (1996); Peña et al. (2002); Pérez et al. (2006b)] and *kernel based Bayesian networks* [Pérez et al. (2009)]. Conditional Gaussian networks and kernel based Bayesian networks are a parametric and non-parametric alternative to the discretization plus multinomial approach, respectively. This dissertation is mainly focused on the presentation of both paradigms for supervised classification. The intuition behind both paradigms is to allow the use of generalized probability functions that can deal with mixed random variables in the field of PGMs.

2.7 Summary and Future work

This chapter has introduced Bayesian multinomial networks focused on supervised classification. For this purpose we have introduced some fundamentals of graph theory and probabilistic graphical models. Bayesian multinomial networks are introduced as an probabilistic graphical model. Based on this paradigm, the family of augmented naïve Bayes structures is presented as a set of structures biased towards supervised classification: naïve Bayes,

tree-augmented naïve Bayes, k-dependent augmented naïve Bayes and joint-augmented naïve Bayes. Finally, a set of classifier induction algorithms is introduced for each subfamily of structures.

A future work line consists of designing new classifier induction algorithms for augmented naïve Bayes structures which control the bias and the variance of the learned classifiers. These algorithms will be focused on two heuristics:

- (a) Equilibrate the number of parameters of the model and the average number of cases for computing each parameter.
- (b) Ensure a minimum number of cases l for computing each parameter.

Both heuristics are proposed in order to control the variance of the classifier (see Section 1.8.3). They could be thought of as being inspired by [Yang and Webb (2003)], which proposes a discretization for the naïve Bayes classifier trying to balance the trade-off between the bias and variance of the classification error. We propose a set of classifier induction algorithms called *dynamic k-dependent augmented naïve Bayes* (*dynkAN*). The general pseudocode for dyn$k$AN algorithms in the forward greedy search version is shown in Algorithm 9, where the function $heuristic(X_i, X_j, \boldsymbol{X}, \mathcal{T})$ depends on the heuristic proposed being $\mathcal{T} = \{(\boldsymbol{x}^{(1)}, c^{(1)}), ..., (\boldsymbol{x}^{(N)}, c^{(N)})\}$ the training set used.

Algorithm 9: **Wrapper selective dynamic multinomial kAN.**

1 Let the structure $\boldsymbol{s} = (\boldsymbol{V}, \boldsymbol{E})$ with $\boldsymbol{E} = \emptyset$ and $\boldsymbol{V} = (\boldsymbol{X}, \boldsymbol{C})$.
2 **While** true.
3 Select the best structure $\boldsymbol{s}^*$, i.e. with the best estimated wrapper score, among the following alternatives:
4 (a) Consider the classifiers with the structure $\boldsymbol{s}' = (\boldsymbol{V}, \boldsymbol{E}')$ with $\boldsymbol{E}' = \boldsymbol{E} \cup (C, X_i)$ where $X_i \notin \boldsymbol{Ch}(\boldsymbol{s})_C$.
5 (b) Consider the classifiers with the structure $\boldsymbol{s}' = (\boldsymbol{V}, \boldsymbol{E}')$ with $\boldsymbol{E}' = \boldsymbol{E} \cup (X_i, X_j)$ where $X_i, X_j \subseteq \boldsymbol{Ch}(\boldsymbol{s})_C$ and $\boldsymbol{Pa}(\boldsymbol{s})_j \leq k + 1$ where $k = heuristic(X_i, X_j, \boldsymbol{X}, \mathcal{T})$.
6 **if** structure $\boldsymbol{s}^*$ improves $\boldsymbol{s}$
7 **then** Set $\boldsymbol{s}$ to $\boldsymbol{s}^*$.
8 **else** Return the structure $\boldsymbol{s}$.

For example, possible implementations of the heuristics could be

- (a) $heuristic(X_i, X_j, \boldsymbol{Pa}(\boldsymbol{s})bdX, \mathcal{T}) = \lfloor \frac{\log(\sqrt{N}/(r_i * r))}{\log \bar{r}_i} \rfloor$
- (b) $heuristic(X_i, X_j, \boldsymbol{X}, \mathcal{T}) = \lfloor \frac{\log N/(lr_i r)}{log \bar{r}_i} \rfloor$

where $\bar{r}_i = 1/|\boldsymbol{Pa}(\boldsymbol{s})_i| \sum_{X_m \in \boldsymbol{Pa}(\boldsymbol{s})_i} r_m$. Note that the proposed implementations for both heuristics come from the equalities $\lceil r_i \bar{r}^k r \rceil = \sqrt{N}$ and $\lfloor N/(r_i \bar{r}^k r) \rfloor = l$, respectively, where l is the average minimum number of cases for computing each parameter.

Besides, we will propose new search strategies based on other metaheuristics such as genetic algorithms [Goldberg (1989)] or estimation of distribution algorithms [Larrañaga and Lozano (2002)].

Part II

Methodological contributions

3

Supervised classification with conditional Gaussian networks

3.1 Introduction

Bayesian multinomial network-based classifiers (see Chapter 2) are able to handle only discrete variables. However, most real-world domains involve continuous variables. A common practice to deal with continuous variables in Bayesian multinomial networks is to discretize them, with the subsequent loss of information (see Section 2.6). In order to deal directly with continuous and discrete random variables, this chapter formally introduces conditional Gaussian networks for supervised classification.

This chapter is organized as follows. Section 3.2 introduces the Gaussian and mixed Gaussian distribution and a set of results related to conditioning and marginalizing. In addition, Section 3.3 provides the estimators for a set of quantities of the information theory under mixed Gaussian assumption. Section 3.4 formally introduces conditional Gaussian networks for general purposes. Then, in Section 3.5 we introduce a set of classifier induction algorithms based on conditional Gaussian networks, most of them originally proposed for Bayesian multinomial networks. The classifier induction algorithms introduced are ordered and grouped according to their structural complexity: naïve Bayes, tree augmented naïve Bayes, k-dependent augmented naïve Bayes and joint augmented naïve Bayes. All the classifier induction algorithms are empirically evaluated in Section 3.6. The study suggests that joint augmented naïve Bayes structure based classifiers seems to outperform the behavior of the rest of the presented classifiers. Joint augmented naïve Bayes structures also obtain quite competitive results compared to the state-of-the-art algorithms included in the study. Finally, Section 4.9 summarizes the main contributions provided throughout this chapter and it indicates the main future work lines.

3.2 Gaussian and conditional Gaussian distributions

This section presents *Gaussian density function* and *conditional Gaussian generalized probability distribution* [Lauritzen and Wermuth (1984); Lauritzen (1992, 1996)], which we call *mixed Gaussian distribution* (*MG distribution*). It also presents a set of propositions taken from [Lauritzen and Wermuth (1989); Lauritzen (1996); Cowell et al. (1999)], which show the properties of the marginal and conditional distributions associated to Gaussian density functions and mixed Gaussian generalized probability distributions. In addition, we provide an alternative proof to most of the theorems and propositions related to Gaussian and mixed Gaussian distributions based on moment characteristics instead of canonical characteristics. It must be highlighted that MG distribution and its marginalizing and conditioning properties are the basis of the definition of conditional Gaussian networks (see Section 3.4). The provided results are not restricted to a particular estimation of the parameters. However, this dissertation is focused on parameters estimated using maximum likelihood.

This section is divided in two parts. Subsection 3.2.1 introduces the notation for sub-matrixes and sub-vectors and, then, Subsection 3.2.2 presents the joint, marginal and conditional forms for Gaussian density functions and mixed Gaussian generalized probability distributions.

The results provided in this chapter are based on three interrelated random variables. Let $\boldsymbol{X} = (X_1, ..., X_{n+m})$ be an $(n+m)$-dimensional mixed random variable. By definition, there exist two subsets of $\boldsymbol{X}$, $\boldsymbol{Y} = (Y_1, ..., Y_n)$ and $\boldsymbol{Z} = (Z_1, ..., Z_m)$, such that (i) $\boldsymbol{X} = \boldsymbol{Y} \cup \boldsymbol{Z}$ and $\boldsymbol{Y} \cap \boldsymbol{Z} = \emptyset$, and (ii) $\boldsymbol{Y}$ is an n-dimensional continuous random variable and $\boldsymbol{Z}$ is an m-dimensional discrete random variable. Thus, there exist two indexed sets $\{1{:}\boldsymbol{Y}, ..., n{:}\boldsymbol{Y}\}$ and $\{1{:}\boldsymbol{Z}, ..., m{:}\boldsymbol{Z}\}$ such that $Y_i = X_{i:\boldsymbol{Y}}$ and $Z_j = X_{j:\boldsymbol{Z}}$.

3.2.1 Notation for matrix and vectors

The notation introduced in this section will be used throughout this dissertation.

Let $\boldsymbol{Y}$ be a multidimensional continuous random variable. We denote a matrix of $\boldsymbol{Y}$ as $A_{\boldsymbol{Y},\boldsymbol{Y}}$ or $A_{\boldsymbol{Y}}$. When it is clear from the context the subindex is omitted, and it is simply referred to as A. Given $\boldsymbol{U} \subseteq \boldsymbol{Y}$ and $\bar{\boldsymbol{U}} = \boldsymbol{Y} \setminus \boldsymbol{U}$, the sub-matrix of A with the elements $A_{U,V}$ for $U \in \boldsymbol{U}$ and $\bar{V} \in \bar{\boldsymbol{U}}$ is denoted as $A_{\boldsymbol{U},\bar{\boldsymbol{U}}}$.

$$A = A_{\boldsymbol{Y}} = A_{\boldsymbol{Y},\boldsymbol{Y}} = \begin{pmatrix} A_{\boldsymbol{U}} & , A_{\boldsymbol{U},\bar{\boldsymbol{U}}} \\ A_{\bar{\boldsymbol{U}},\boldsymbol{U}} & , A_{\bar{\boldsymbol{U}}} \end{pmatrix} = \begin{pmatrix} A_{\boldsymbol{U},\boldsymbol{U}} & , A_{\boldsymbol{U},\bar{\boldsymbol{U}}} \\ A_{\bar{\boldsymbol{U}},\boldsymbol{U}} & , A_{\bar{\boldsymbol{U}},\bar{\boldsymbol{U}}} \end{pmatrix}$$

We denote a vector related to $\boldsymbol{Y}$ as $\boldsymbol{a}_{\boldsymbol{Y}}$. When it is clear from the context the subindex is omitted, and it is simply referred to as $\boldsymbol{a}$. Given $\boldsymbol{U} \subseteq \boldsymbol{Y}$ the sub-vector of $\boldsymbol{a}$ with the elements $\boldsymbol{a}_U$ for $U \in \boldsymbol{U}$ is denoted as $\boldsymbol{a}_{\boldsymbol{U}}$.

$$\boldsymbol{a} = \boldsymbol{a}_{\boldsymbol{Y}} = (\boldsymbol{a}_{\boldsymbol{U}}, \boldsymbol{a}_{\bar{\boldsymbol{U}}}) = ((\boldsymbol{a}_{\boldsymbol{Y}})_{\boldsymbol{U}}, (\boldsymbol{a}_{\boldsymbol{Y}})_{\bar{\boldsymbol{U}}})$$

Let $\boldsymbol{Z}$ be a multidimensional discrete random variable which takes the values $\Omega_{\boldsymbol{z}} = \{\boldsymbol{z}_1, ..., \boldsymbol{z}_r\}$. A set of matrixes and vectors for $\boldsymbol{Y}$ indexed by the values in $\Omega_{\boldsymbol{z}}$ are denoted as $\boldsymbol{A}_{\boldsymbol{Y}}(\boldsymbol{Z})$ and $\boldsymbol{a}_{\boldsymbol{Y}}(\boldsymbol{Z})$, respectively. Alternatively, the matrix and vector indexed by the value $\boldsymbol{z}$ are denoted as $\boldsymbol{A}_{\boldsymbol{Y}}(\boldsymbol{z})$ and $\boldsymbol{a}_{\boldsymbol{Y}}(\boldsymbol{z})$, respectively.

3.2.2 Joint, conditional and marginal distributions

Following, we formally introduce the joint, marginal and conditional functions for Gaussian density functions and mixed Gaussian generalized probability distributions. Most of the presented results have been taken from [Lauritzen (1996)].

Definition 3.1 (Joint Gaussian) $\boldsymbol{Y} = (Y_1, ..., Y_n)$ *is said to follow an* n-dimensional Gaussian density function *if its joint probability density function is an* n*-dimensional normal distribution,* $f(\boldsymbol{y}) \rightsquigarrow \mathcal{N}(\boldsymbol{y}; \boldsymbol{\mu}, \Sigma)$*:*

$$f(\boldsymbol{y}) = \frac{1}{(2\pi)^{n/2} \mid \Sigma \mid^{1/2}} \exp[-\frac{1}{2}(\boldsymbol{y} - \boldsymbol{\mu})^T \Sigma^{-1} (\boldsymbol{y} - \boldsymbol{\mu})]$$

where Σ *and* $\boldsymbol{\mu}$ *are the covariance matrix and mean vector of* $\boldsymbol{Y}$*, respectively.*

Note that $\Sigma_{Y_i,Y_j} = \Sigma_{Y_j,Y_i}$ and, thus, the number of parameters required for modeling $f(\boldsymbol{y})$ is $n(n+1)/2 + n$. Therefore, the order of the number of parameters is $\mathcal{O}(n^2)$. From here on, we will refer to the joint Gaussian probability density function as *Gaussian distribution*, for the sake of simplicity.

Proposition 3.1 (Marginal Gaussian) *Let* $\boldsymbol{Y}$ *follow an* n*-dimensional Gaussian density function,* $f(\boldsymbol{y}) \rightsquigarrow \mathcal{N}(\boldsymbol{y}; \boldsymbol{\mu}_{\boldsymbol{Y}}, \Sigma_{\boldsymbol{Y}})$*, and let* $\boldsymbol{U} = (U_1, ..., U_k) \subseteq \boldsymbol{Y}$ *and* $\bar{\boldsymbol{U}} = \boldsymbol{Y} \setminus \boldsymbol{U}$*. The marginal joint density function of* $\boldsymbol{U}$*, obtained by marginalizing* $f(\boldsymbol{y})$ *on* $\bar{\boldsymbol{U}}$*, follows an* k*-dimensional Gaussian distribution with parameters* $\boldsymbol{\mu}_{\boldsymbol{U}} = (\boldsymbol{\mu}_{\boldsymbol{Y}})_{\boldsymbol{U}}$ *and* $\Sigma_{\boldsymbol{U}} = (\Sigma_{\boldsymbol{Y}})_{\boldsymbol{U}}$*,* $f(\boldsymbol{u}) = \int f(\boldsymbol{y}) d\bar{\boldsymbol{u}} \rightsquigarrow \mathcal{N}(\boldsymbol{u}; \boldsymbol{\mu}_{\boldsymbol{U}}, \Sigma_{\boldsymbol{U}})$*.*

The proof to this proposition can be found in Lauritzen (1996), p.255-256, using canonical characteristics instead of moment characteristics. The number of parameters required for modeling the marginal distribution $f(\boldsymbol{u})$ is $l(l+1)/2 + l$. Thus the order of the number of parameters is $\mathcal{O}(l^2)$.

Proposition 3.2 (Conditional Gaussian) *Let* $\boldsymbol{Y} = (Y_1, ..., Y_n)$ *follow an* n*-dimensional Gaussian density function,* $f(\boldsymbol{y}) \rightsquigarrow \mathcal{N}(\boldsymbol{y}; \boldsymbol{\mu}_{\boldsymbol{Y}}, \Sigma_{\boldsymbol{Y}})$*, and let* $\boldsymbol{U} = (U_1, ..., U_k) \subseteq \boldsymbol{Y}$ *and* $\bar{\boldsymbol{U}} = \boldsymbol{Y} \setminus \boldsymbol{U}$*. The conditional joint density function of* $\boldsymbol{U}$ *conditioned to* $\bar{\boldsymbol{U}} = \bar{\boldsymbol{u}}$*,* $f(\boldsymbol{U} = \boldsymbol{u} \mid \bar{\boldsymbol{U}} = \bar{\boldsymbol{u}})$*, follows an* k*-dimensional joint Gaussian density function with parameters* $\boldsymbol{\mu}_{\boldsymbol{U}|\bar{\boldsymbol{u}}}$ *and* $\Sigma_{\boldsymbol{U}|\bar{\boldsymbol{U}}}$*,* $f(\boldsymbol{u}|\bar{\boldsymbol{u}}) = \frac{f(\boldsymbol{u},\bar{\boldsymbol{u}})}{\int f(\boldsymbol{u},\bar{\boldsymbol{u}}) d\boldsymbol{u}} \rightsquigarrow \mathcal{N}(\boldsymbol{u}; \boldsymbol{\mu}_{\boldsymbol{U}|\bar{\boldsymbol{u}}}, \Sigma_{\boldsymbol{U}|\bar{\boldsymbol{U}}})$*, where*

$$\boldsymbol{\mu}_{\boldsymbol{U}|\bar{\boldsymbol{u}}} = \boldsymbol{\mu}_{\boldsymbol{U}} + \Sigma_{\boldsymbol{U},\bar{\boldsymbol{U}}}\Sigma_{\bar{\boldsymbol{U}}}^{-1}(\bar{\boldsymbol{u}} - \boldsymbol{\mu}_{\bar{\boldsymbol{U}}})$$
$$\Sigma_{\boldsymbol{U}|\bar{\boldsymbol{U}}} = \Sigma_{\boldsymbol{U}} - \Sigma_{\boldsymbol{U},\bar{\boldsymbol{U}}}\Sigma_{\bar{\boldsymbol{U}}}^{-1}\Sigma_{\bar{\boldsymbol{U}},\boldsymbol{U}}$$

The term $\Sigma_{\boldsymbol{U},\bar{\boldsymbol{U}}}\Sigma_{\bar{\boldsymbol{U}}}^{-1}$ is a $k \times (n-k)$ matrix which is usually referred as regression coefficients [Shachter and Kenley (1989); Lauritzen (1992)].

The proof to Proposition 3.2 can be found in Lauritzen (1996), p.255, using the canonical characteristics. Note that the covariance $\Sigma_{\boldsymbol{U}|\bar{\boldsymbol{U}}}$ is independent from the value $\bar{\boldsymbol{u}}$, while $\boldsymbol{\mu}_{\boldsymbol{U}|\bar{\boldsymbol{u}}}$ depends on the entire covariance matrix Σ, the mean vector $\boldsymbol{\mu}$ and the value $\bar{\boldsymbol{u}}$. The number of required parameters is $n(n+1)+n$. Therefore, the order of the number of parameters is $\mathcal{O}(m^2)$. It should be highlighted that in contrast to marginal distribution $f(\boldsymbol{u})$, the conditional Gaussian distribution $f(\boldsymbol{u}|\bar{\boldsymbol{u}})$ does not reduce the number of parameters with respect to the joint distribution $f(\boldsymbol{y})$.

Let us consider the joint Gaussian density for $\boldsymbol{Y}$, $f(\boldsymbol{y}) \rightsquigarrow \mathcal{N}(\boldsymbol{y}; \boldsymbol{\mu}_{\boldsymbol{Y}}, \Sigma_{\boldsymbol{Y}})$ and a factorization based on a structure $\boldsymbol{s}$ for $\boldsymbol{Y}$, $\prod_{i=1}^{n} f(y_i|\boldsymbol{pa}(\boldsymbol{s})_i)$, where $f(y_i|\boldsymbol{pa}(\boldsymbol{s})_i) \rightsquigarrow \mathcal{N}(y_i; \mu_{Y_i|\boldsymbol{pa}(\boldsymbol{s})_i}, \sigma^2_{Y_i|\boldsymbol{Pa}(\boldsymbol{s})_i})$. The reader should note that, Propositions 3.1 and 3.2 can be used to obtain the parameters of $f(y_i|\boldsymbol{pa}(\boldsymbol{s})_i)$, $\mu_{Y_i|\boldsymbol{pa}(\boldsymbol{s})_i}$ and $\sigma^2_{Y_i|\boldsymbol{Pa}(\boldsymbol{s})_i}$, from the parameters of the joint Gaussian density $f(\boldsymbol{y})$, $\boldsymbol{\mu}_{\boldsymbol{Y}}$ and $\Sigma_{\boldsymbol{Y}}$, for $i = 1, ..., n$. Conversely, in order to obtain the parameters of the joint density $f(\boldsymbol{y})$ from the parameters of the factors $f(y_i|\boldsymbol{pa}(\boldsymbol{s})_i)$ for $i = 1, ..., n$, the reader should consult the procedure proposed in [Shachter and Kenley (1989)], p. 538 and Appendix B.

Definition 3.2 (Joint mixed Gaussian) *Let $\boldsymbol{X} = (\boldsymbol{Y}, \boldsymbol{Z}) = (X_1, ..., X_{n+m})$ be an $(n+m)$-dimensional mixed random variable, where $\boldsymbol{Y} = (Y_1, ..., Y_n)$ is an n-dimensional continuous random variable and $\boldsymbol{Z} = (Z_1, ..., Z_m)$ is an m-dimensional discrete random variable. $\boldsymbol{X}$ is said to follow an* $(n+m)$-dimensional mixed Gaussian distribution *if its generalized joint probability distribution is given by:*

$$\rho(\boldsymbol{x}) = f(\boldsymbol{y} \mid \boldsymbol{z})p(\boldsymbol{z}) \rightsquigarrow \mathcal{MG}(\boldsymbol{x}; p(\boldsymbol{z}), \boldsymbol{\mu}_{\boldsymbol{Y}}(\boldsymbol{z}), \Sigma_{\boldsymbol{Y}}(\boldsymbol{z})) \tag{3.1}$$

for $p(\boldsymbol{z}) > 0$, where $p(\boldsymbol{z})$ follows an m-dimensional multinomial probability mass function, $p(\boldsymbol{z}) = Pr(\boldsymbol{Z} = \boldsymbol{z})$, and $f(\boldsymbol{y} \mid \boldsymbol{z})$ follows an n-dimensional joint Gaussian density function for all $\boldsymbol{z}$, $f(\boldsymbol{y}|\boldsymbol{z}) \rightsquigarrow \mathcal{N}(\boldsymbol{y}; \Sigma_{\boldsymbol{Y}}(\boldsymbol{z}), \boldsymbol{\mu}_{\boldsymbol{Y}}(\boldsymbol{z}))$, where $\Sigma_{\boldsymbol{Y}}(\boldsymbol{z})$ and $\boldsymbol{\mu}_{\boldsymbol{Y}}(\boldsymbol{z})$ are the covariance matrix and mean vector of $\boldsymbol{Y}$ when $\boldsymbol{Z} = \boldsymbol{z}$, respectively.

In order to avoid misunderstandings, and for the sake of brevity, it should be noted that we refer to *conditional Gaussian generalized probability distribution* as *mixed Gaussian distribution* (*MG distribution*). The term conditional Gaussian distribution is used in this dissertation for naming a conditional Gaussian density function (see Proposition 3.2). We prefer the term *mixed* instead of conditional, because it is used for naming a joint generalized probability function and not a conditional one, and because it provides information about the nature of the implied random variables.

We denote that $\boldsymbol{X}$ follows a MG distribution with parameters $p(\boldsymbol{z})$, $\Sigma_{\boldsymbol{Y}}(\boldsymbol{z})$ and $\boldsymbol{\mu}_{\boldsymbol{Y}}(\boldsymbol{z})$ as $\rho(\boldsymbol{x}) \rightsquigarrow \mathcal{MG}(\boldsymbol{x}; p(\boldsymbol{z}), \Sigma_{\boldsymbol{Y}}(\boldsymbol{z}), \boldsymbol{\mu}_{\boldsymbol{Y}}(\boldsymbol{z}))$. It must be noted that a mixed Gaussian distribution is a generalization of both Gaussian density function and multinomial probability distribution for multidimensional mixed random variables. Intuitively, it can be said that the statement "$\boldsymbol{X}$ follows a MG distribution", is equivalent to the statement "Y follows a multidimensional Gaussian probability density given the discrete variables" [Cowell et al. (1999)].

The number of parameters for modeling the mixed Gaussian distribution for $\boldsymbol{X}$, $\rho(\boldsymbol{x})$, is $(\prod_{i=1}^{m} r_i)-1+(\prod_{i=1}^{m} r_i)n(n+1)/2+(\prod_{i=1}^{m} r_i)n$. Thus, the order of magnitude of the number of parameters is $\mathcal{O}(n^2(\prod_{i=1}^{m} r_i))$. If we assume that $r_i = r$ for any $1 \leq i \leq m$ the number of required parameters is $\mathcal{O}(n^2 r^m)$.

Conversely to Gaussian distribution, the marginal of a MG distribution is not always a MG distribution [Lauritzen (1996)]. The marginalization of MG distributions over continuous random variables always follows a MG distribution but, on the other hand, the marginalization of MG distributions over discrete random variables leads in general to complicated mixture distributions [Lauritzen (1996)]. The following two propositions formally introduce these facts giving the closed forms for the marginal and conditional MG distributions. These propositions have been taken from [Lauritzen (1996)].

Proposition 3.3 (Marginal mixed Gaussian I) *Let $\boldsymbol{X} = (\boldsymbol{Y}, \boldsymbol{Z}) = (X_1, ..., X_{n+m})$ be an $(n + m)$-dimensional mixed random variable, being $\boldsymbol{Y} = (Y_1, ..., Y_n)$ an n-dimensional continuous random variable and $\boldsymbol{Z} = (Z_1, ..., Z_m)$ an m-dimensional discrete random variable. Let $\boldsymbol{U} = (U_1, ..., U_k) \subseteq \boldsymbol{Y}$ be a multidimensional continuous random variable being $\bar{\boldsymbol{U}} = \boldsymbol{Y} \setminus \boldsymbol{U}$ its complement in $\boldsymbol{Y}$. If $\boldsymbol{X}$ follows an $(n + m)$-dimensional mixed Gaussian distribution, $\rho(\boldsymbol{x}) \rightsquigarrow \mathcal{MG}(\boldsymbol{x}; p(\boldsymbol{z}), \Sigma(\boldsymbol{z}), \boldsymbol{\mu}(\boldsymbol{z}))$, then $(\boldsymbol{U}, \boldsymbol{Z})$ follows the mixed Gaussian distribution:*

$$\rho(\boldsymbol{u}, \boldsymbol{z}) = \int \rho(\boldsymbol{x}) d\bar{\boldsymbol{u}} \rightsquigarrow \mathcal{MG}(\boldsymbol{u}, \boldsymbol{z}; p(\boldsymbol{z}), \boldsymbol{\mu}_{\boldsymbol{U}}(\boldsymbol{z}), \Sigma_{\boldsymbol{U}}(\boldsymbol{z}))$$

where

$$\boldsymbol{\mu}_{\boldsymbol{U}}(\boldsymbol{z}) = \boldsymbol{\mu}(\boldsymbol{z})_{\boldsymbol{U}}$$
$$\Sigma_{\boldsymbol{U}}(\boldsymbol{z}) = \Sigma(\boldsymbol{z})_{\boldsymbol{U}}$$

Proof.

$$\rho(\boldsymbol{u}, \boldsymbol{z}) = \int \rho(\boldsymbol{x}) d\bar{\boldsymbol{u}} = p(\boldsymbol{z}) \int f(\boldsymbol{y}|\boldsymbol{z}) d\bar{\boldsymbol{u}} = p(\boldsymbol{z}) f(\boldsymbol{u}|\boldsymbol{z})$$

Since $f(\boldsymbol{y}|\boldsymbol{z}) \rightsquigarrow \mathcal{N}(\boldsymbol{y}; \boldsymbol{\mu}(\boldsymbol{z}), \Sigma(\boldsymbol{z}))$, due to Proposition 3.2 the term $f(\boldsymbol{u}|\boldsymbol{z})$ follows a Gaussian density function for each value $\boldsymbol{z}$, $f(\boldsymbol{u}|\boldsymbol{z}) \rightsquigarrow \mathcal{N}(\boldsymbol{u}; \boldsymbol{\mu}_{\boldsymbol{U}}(\boldsymbol{z}), \Sigma_{\boldsymbol{U}}(\boldsymbol{z}))$ where $\boldsymbol{\mu}_{\boldsymbol{U}}(\boldsymbol{z}) = \boldsymbol{\mu}(\boldsymbol{z})_{\boldsymbol{U}}$ and $\Sigma_{\boldsymbol{U}}(\boldsymbol{z}) = \Sigma(\boldsymbol{z})_{\boldsymbol{U}}$. ■

The proof of this proposition can be found in Lauritzen (1996), p.160, using the canonical characteristics.

The number of required parameters for modeling the marginal mixed distribution $\rho(\boldsymbol{u}, \boldsymbol{z})$ is $(\prod_{i=1}^{m} r_i) - 1 + (\prod_{i=1}^{m} r_i)l(l+1)/2 + (\prod_{i=1}^{m} r_i)l$. Thus, the order of magnitude of the number of parameters is $\mathcal{O}(l^2(\prod_{i=1}^{m} r_i))$. If we assume that $r_i = r$ for any $1 \leq i \leq m$ the number of required parameters is $\mathcal{O}(l^2 r^m)$. The reduction in the number of parameters with respect to the joint mixed distribution $\rho(\boldsymbol{x})$ is inherited from the reduction in the parameters of a Gaussian distribution when it is marginalized (see Proposition 3.1).

Proposition 3.4 (Marginal mixed Gaussian II) *Let $\boldsymbol{X} = (\boldsymbol{Y}, \boldsymbol{Z}) = (X_1, ..., X_{n+m})$ be an $(n+m)$-dimensional mixed random variable which follows a mixed Gaussian distribution, $\rho(\boldsymbol{x}) \rightsquigarrow \mathcal{MG}(\boldsymbol{x}; p(\boldsymbol{z}), \boldsymbol{\mu}_{\boldsymbol{Y}}(\boldsymbol{z}), \Sigma_{\boldsymbol{Y}}(\boldsymbol{z}))$ being $\boldsymbol{Y} = (Y_1, ..., Y_n)$ an n-dimensional continuous random variable and $\boldsymbol{Z} = (Z_1, ..., Z_m)$ an m-dimensional discrete random variable. Let $\boldsymbol{V} = (V_1, ..., V_l) \subseteq \boldsymbol{Z}$ be a multidimensional discrete random variable being $\bar{\boldsymbol{V}} = \boldsymbol{Z} \setminus \boldsymbol{V}$ its complement in $\boldsymbol{Y}$. If the conditional independence assertion $CI(\bar{\boldsymbol{V}}; \boldsymbol{Y}|\boldsymbol{V})$ is satisfied, then $(\boldsymbol{Y}, \boldsymbol{V})$ follows the mixed Gaussian distribution*

$$\rho(\boldsymbol{y}, \boldsymbol{v}) \rightsquigarrow \mathcal{MG}(\boldsymbol{y}, \boldsymbol{v}; p(\boldsymbol{v}), \boldsymbol{\mu}_{\boldsymbol{Y}}(\boldsymbol{v}), \Sigma_{\boldsymbol{Y}}(\boldsymbol{v}))$$

where

$$p(\boldsymbol{v}) = \sum_{\bar{\boldsymbol{v}}} p(\boldsymbol{v}, \bar{\boldsymbol{v}})$$
$$\boldsymbol{\mu}_{\boldsymbol{Y}}(\boldsymbol{v}) = \boldsymbol{\mu}_{\boldsymbol{Y}}(\boldsymbol{v}, \bar{\boldsymbol{v}}) \text{ for any } \bar{\boldsymbol{v}}$$
$$\Sigma_{\boldsymbol{Y}}(\boldsymbol{v}) = \Sigma_{\boldsymbol{Y}}(\boldsymbol{v}, \bar{\boldsymbol{v}}) \text{ for any } \bar{\boldsymbol{v}}$$

Proof.

$$\rho(\boldsymbol{y}, \boldsymbol{v}) = \sum_{\bar{\boldsymbol{v}}} \rho(\boldsymbol{x}) = \sum_{\bar{\boldsymbol{v}}} f(\boldsymbol{y}|\boldsymbol{z}) p(\boldsymbol{z}) = f(\boldsymbol{y}|\boldsymbol{v}) \sum_{\bar{\boldsymbol{v}}} p(\boldsymbol{v}, \bar{\boldsymbol{v}}) = f(\boldsymbol{y}|\boldsymbol{v}) p(\boldsymbol{v})$$

The third equality holds due to the conditional independence $CI(\bar{\boldsymbol{V}}; \boldsymbol{Y}|\boldsymbol{V})$. Note that, for all $\boldsymbol{v}$, both mean vector $\boldsymbol{\mu}_{\boldsymbol{Y}}(\boldsymbol{v}, \bar{\boldsymbol{v}})$ and covariance matrix $\Sigma_{\boldsymbol{Y}}(\boldsymbol{v}, \bar{\boldsymbol{v}})$ are independent from the value $\bar{\boldsymbol{v}}$. Thus, for all $\bar{\boldsymbol{v}}$, the parameters $\boldsymbol{\mu}_{\boldsymbol{Y}}(\boldsymbol{v}, \bar{\boldsymbol{v}})$ and $\Sigma_{\boldsymbol{Y}}(\boldsymbol{v}, \bar{\boldsymbol{v}})$ can be summarized in $\boldsymbol{\mu}_{\boldsymbol{Y}}(\boldsymbol{v})$ and $\Sigma_{\boldsymbol{Y}}(\boldsymbol{v})$, respectively. ■

The proof of this proposition can be found in Lauritzen (1996), p.161, using the canonical characteristics. It must be highlighted that, in fact, the converse to Proposition 3.4 holds, i.e. if the marginal of a MG distribution, $\rho(\boldsymbol{x})$, over the variable $\bar{\boldsymbol{V}}$, $\rho(\boldsymbol{y}, \boldsymbol{v})$, is again a MG distribution, then the conditional independence assertion $CI(\bar{\boldsymbol{V}}; \boldsymbol{Y}|\boldsymbol{V})$ is true [Frydemberg (1990)].

The number of parameters required for modeling the marginal $\rho(\boldsymbol{y}, \boldsymbol{v})$ is $(\prod_{i=1}^{l} r_{i:\boldsymbol{V}}) - 1 + (\prod_{i=1}^{l} r_{i:\boldsymbol{V}})n(n+1)/2 + (\prod_{i=1}^{l} r_{i:\boldsymbol{V}})n$. Thus, the order of magnitude of the number of parameters is $\mathcal{O}(n^2(\prod_{i=1}^{l} r_{i:\boldsymbol{V}}))$. If we assume that

$r_{i:\boldsymbol{V}} = r$ for any $1 \leq i \leq l$ the number of required parameters is $\mathcal{O}(n^2 r^l)$. The reduction of parameters is due to the conditional independence $CI(\bar{\boldsymbol{V}}; \boldsymbol{Y}|\boldsymbol{V})$ which implies that some of the parameters involved in $\rho(\boldsymbol{x})$ are repeated and can be summarized.

In contrast to the situation concerning marginals, conditioning with any subset of variables preserves the MG distribution [Lauritzen (1996)]. The following theorem [Lauritzen (1996)] gives the closed forms for the parameters of a conditional MG distribution.

Proposition 3.5 (Conditional mixed Gaussian) *Let $\boldsymbol{X} = (\boldsymbol{Y}, \boldsymbol{Z}) = (X_1, ..., X_{n+m})$ be an $(n+m)$-dimensional mixed random variable which follows a mixed Gaussian distribution, $\rho(\boldsymbol{x}) \rightsquigarrow \mathcal{MG}(\boldsymbol{x}; p(\boldsymbol{z}), \boldsymbol{\mu}(\boldsymbol{z}), \Sigma(\boldsymbol{z}))$, being $\boldsymbol{Y} = (Y_1, ..., Y_n)$ an n-dimensional continuous random variable and $\boldsymbol{Z} = (Z_1, ..., Z_m)$ an m-dimensional discrete random variable. Let $\boldsymbol{U} = (U_1, ..., U_k) \subseteq \boldsymbol{Y}$ be a multidimensional continuous random variable and $\boldsymbol{V} = (V_1, ..., V_k) \subseteq \boldsymbol{Z}$ be a multidimensional discrete random variable, being $\bar{\boldsymbol{U}} = \boldsymbol{Y} \setminus \boldsymbol{U}$ and $\bar{\boldsymbol{V}} = \boldsymbol{Z} \setminus \boldsymbol{V}$ their complements in $\boldsymbol{Y}$ and $\boldsymbol{Z}$, respectively. For each value $(\bar{\boldsymbol{u}}, \bar{\boldsymbol{v}})$, $(\boldsymbol{U}, \boldsymbol{V})$ follows a multivariate mixed Gaussian distribution, $\rho(\boldsymbol{u}, \boldsymbol{v}|\bar{\boldsymbol{u}}, \bar{\boldsymbol{v}}) \rightsquigarrow \mathcal{MG}(\boldsymbol{u}, \boldsymbol{v}; p(\boldsymbol{v}|\bar{\boldsymbol{u}}, \bar{\boldsymbol{v}}), \boldsymbol{\mu}_{\boldsymbol{U}|\bar{\boldsymbol{u}}}((\boldsymbol{v}, \bar{\boldsymbol{v}}) = \boldsymbol{z}), \Sigma_{\boldsymbol{U}|\bar{\boldsymbol{U}}}(\boldsymbol{z}))$, where*

$$p(\boldsymbol{v}|\bar{\boldsymbol{u}}, \bar{\boldsymbol{v}}) = p(\boldsymbol{v}|\bar{\boldsymbol{v}}) \frac{\mathcal{N}(\bar{\boldsymbol{u}}; \boldsymbol{\mu}(\boldsymbol{v}, \bar{\boldsymbol{v}})_{\bar{\boldsymbol{U}}}, \Sigma(\boldsymbol{v}, \bar{\boldsymbol{v}})_{\bar{\boldsymbol{U}}})}{\sum_{\boldsymbol{v}'} p(\boldsymbol{v}'|\bar{\boldsymbol{v}}) \mathcal{N}(\bar{\boldsymbol{u}}; \boldsymbol{\mu}(\boldsymbol{v}', \bar{\boldsymbol{v}})_{\bar{\boldsymbol{U}}}, \Sigma(\boldsymbol{v}', \bar{\boldsymbol{v}})_{\bar{\boldsymbol{U}}})}$$

$$\boldsymbol{\mu}_{\boldsymbol{U}|\bar{\boldsymbol{u}}}(\boldsymbol{v}, \bar{\boldsymbol{v}}) = \boldsymbol{\mu}(\boldsymbol{v}, \bar{\boldsymbol{v}})_{\boldsymbol{U}} + \Sigma(\boldsymbol{v}, \bar{\boldsymbol{v}})_{\boldsymbol{U}, \bar{\boldsymbol{U}}} \Sigma(\boldsymbol{v}, \bar{\boldsymbol{v}})_{\bar{\boldsymbol{U}}}^{-1} (\bar{\boldsymbol{u}} - \boldsymbol{\mu}(\boldsymbol{z})_{\bar{\boldsymbol{U}}})$$

$$\Sigma_{\boldsymbol{U}|\bar{\boldsymbol{U}}}(\boldsymbol{v}, \bar{\boldsymbol{v}}) = \Sigma(\boldsymbol{v}, \bar{\boldsymbol{v}})_{\boldsymbol{U}} - \Sigma(\boldsymbol{v}, \bar{\boldsymbol{v}})_{\boldsymbol{U}, \bar{\boldsymbol{U}}} \Sigma(\boldsymbol{v}, \bar{\boldsymbol{v}})_{\bar{\boldsymbol{U}}}^{-1} \Sigma(\boldsymbol{v}, \bar{\boldsymbol{v}})_{\bar{\boldsymbol{U}}, \boldsymbol{U}}$$

Proof.

$$\rho(\boldsymbol{u}, \boldsymbol{v}|\bar{\boldsymbol{u}}, \bar{\boldsymbol{v}}) = p(\boldsymbol{v}|\bar{\boldsymbol{u}}, \bar{\boldsymbol{v}}) f(\boldsymbol{u}|\bar{\boldsymbol{u}}, \boldsymbol{z})$$

where $\boldsymbol{z} = (\boldsymbol{v}, \bar{\boldsymbol{v}})$.

For any $\boldsymbol{z}$, $f(\boldsymbol{u}|\bar{\boldsymbol{u}}, \boldsymbol{z})$ can be seen as the conditional form of $f(\boldsymbol{u}, \bar{\boldsymbol{u}}|\boldsymbol{z})$, which follows a Gaussian density function by Definition 3.2. Thus, for any $\boldsymbol{z}$, by Proposition 3.2, we have that $f(\boldsymbol{u}|\bar{\boldsymbol{u}}, \boldsymbol{z}) \rightsquigarrow \mathcal{N}(\boldsymbol{u}; \boldsymbol{\mu}_{\boldsymbol{U}|\bar{\boldsymbol{u}}}(\boldsymbol{z}), \Sigma_{\boldsymbol{U}|\bar{\boldsymbol{U}}}(\boldsymbol{z}))$, where

$$\boldsymbol{\mu}_{\boldsymbol{U}|\bar{\boldsymbol{u}}}(\boldsymbol{v}, \bar{\boldsymbol{v}}) = \boldsymbol{\mu}(\boldsymbol{v}, \bar{\boldsymbol{v}})_{\boldsymbol{U}} + \Sigma(\boldsymbol{v}, \bar{\boldsymbol{v}})_{\boldsymbol{U}, \bar{\boldsymbol{U}}} \Sigma(\boldsymbol{v}, \bar{\boldsymbol{v}})_{\bar{\boldsymbol{U}}}^{-1} (\bar{\boldsymbol{u}} - \boldsymbol{\mu}(\boldsymbol{z})_{\bar{\boldsymbol{U}}})$$

$$\Sigma_{\boldsymbol{U}|\bar{\boldsymbol{U}}}(\boldsymbol{v}, \bar{\boldsymbol{v}}) = \Sigma(\boldsymbol{v}, \bar{\boldsymbol{v}})_{\boldsymbol{U}} - \Sigma(\boldsymbol{v}, \bar{\boldsymbol{v}})_{\boldsymbol{U}, \bar{\boldsymbol{U}}} \Sigma(\boldsymbol{v}, \bar{\boldsymbol{v}})_{\bar{\boldsymbol{U}}}^{-1} \Sigma(\boldsymbol{v}, \bar{\boldsymbol{v}})_{\bar{\boldsymbol{U}}, \boldsymbol{U}}$$

Given $\bar{\boldsymbol{u}}$ and $\bar{\boldsymbol{v}}$, the term $p(\boldsymbol{v}|\bar{\boldsymbol{u}}, \bar{\boldsymbol{v}})$ is given by the following normalized quantity:

$$p(\boldsymbol{v}|\bar{\boldsymbol{u}}, \bar{\boldsymbol{v}}) = \frac{\rho(\bar{\boldsymbol{u}}, \boldsymbol{v}, \bar{\boldsymbol{v}})}{\sum_{\boldsymbol{v}'} \rho(\bar{\boldsymbol{u}}, \boldsymbol{v}, \bar{\boldsymbol{v}})} = p(\boldsymbol{v}|\bar{\boldsymbol{v}}) \frac{f(\bar{\boldsymbol{u}}|\boldsymbol{v}, \bar{\boldsymbol{v}})}{\sum_{\boldsymbol{v}'} p(\boldsymbol{v}'|\bar{\boldsymbol{v}}) f(\bar{\boldsymbol{u}}|\boldsymbol{v}', \bar{\boldsymbol{v}})}$$

where, by Definition 3.2 and Proposition 3.3, for any $(\boldsymbol{v}, \bar{\boldsymbol{v}})$, $f(\bar{\boldsymbol{u}}|\boldsymbol{v}, \bar{\boldsymbol{v}})$ is distributed according to $\mathcal{N}(\bar{\boldsymbol{u}}; \boldsymbol{\mu}(\boldsymbol{v}, \bar{\boldsymbol{v}})_{\bar{\boldsymbol{U}}}, \Sigma(\boldsymbol{v}, \bar{\boldsymbol{v}})_{\bar{\boldsymbol{U}}})$. ■

An alternative proof of this proposition using the canonical characteristics can be found in Lauritzen (1996), p.164.

The number of parameters required for modeling $\rho(\boldsymbol{u}, \boldsymbol{v}|\bar{\boldsymbol{u}}, \bar{\boldsymbol{v}})$ for all $(\bar{\boldsymbol{u}}, \bar{\boldsymbol{v}})$ is $\prod_{i=1}^{m-k} r_{i:\bar{\boldsymbol{V}}}((\prod_{i=1}^{k} r_{i:\boldsymbol{V}}) - 1) + n(\prod_{i=1}^{m} r_{i:\boldsymbol{Z}}) + n(n+1)/2(\prod_{i=1}^{m} r_{i:\boldsymbol{Z}})$. Thus, the order of magnitude of the number of parameters is $\mathcal{O}(n^2 \prod_{i=1}^{m} r_{i:\boldsymbol{Z}})$. If we assume that $r_i = r$ for $i = 1, ..., m$ the number of parameters required is $\mathcal{O}(r^m n^2)$. Note that the number of parameters required for modeling the conditional distribution $\rho(\boldsymbol{u}, \boldsymbol{v}|\bar{\boldsymbol{u}}, \bar{\boldsymbol{v}})$ for all $(\bar{\boldsymbol{u}}, \bar{\boldsymbol{v}})$ is slightly smaller than the number of parameters required for modeling the joint distribution $\rho(\boldsymbol{x})$. However, the number of parameters required is of the same order of magnitude.

3.3 Information theory and Gaussian assumption

This section introduces a set of estimators for the mutual information and conditional mutual information for continuous and mixed random variables under Gaussian assumptions. In order to present the estimators for general purpose the formulation is presented for multidimensional random variables. These estimators will be used by the filter classifier induction algorithms introduced in Section 3.5.

Following the notation of Section 3.2, the results provided in this section are based on the following interrelated random variables: $\boldsymbol{X} = (X_1, ..., X_{n+m}) = (\boldsymbol{Y}, \boldsymbol{Z})$ is a $(n+m)$-dimensional random variable. $\boldsymbol{Y} = (Y_1, ..., Y_n) = (\boldsymbol{U}, \bar{\boldsymbol{U}})$ is an n-dimensional continuous random variable, where $\bar{\boldsymbol{U}} = \boldsymbol{Y} \setminus \boldsymbol{U}$ and $\boldsymbol{U} = (U_1, ..., U_k)$. $\boldsymbol{Z} = (Z_1, ..., Z_m) = (\boldsymbol{V}, \bar{\boldsymbol{V}})$ is an m-dimensional discrete random variable, where $\bar{\boldsymbol{V}} = \boldsymbol{Z} \setminus \boldsymbol{V}$ and $\boldsymbol{V} = (V_1, ..., V_l)$. The following proposition has been taken from [Cover and Thomas (1991)].

Proposition 3.6 *Let $\boldsymbol{Y} = (Y_1, ..., Y_n)$ be an n-dimensional continuous random variable normally distributed with parameters $\boldsymbol{\mu}$ and Σ, $f(\boldsymbol{y}) \rightsquigarrow \mathcal{N}(\boldsymbol{y}; \boldsymbol{\mu}, \Sigma)$. The differential entropy of $\boldsymbol{Y}$ is given by:*

$$h(\boldsymbol{Y}) = \frac{n}{2}\log(2\pi e) + \frac{1}{2}\log(|\Sigma|)$$

The proof can be found in Cover and Thomas (1991).

Note that the differential entropy of a Gaussian distributed n-dimensional continuous random variable is proportional to $\log(|\Sigma|)$. In the particular case of $n = 1$, the entropy is computed as

$$h(Y) = \frac{1}{2}\log(2\pi e)\sigma^2$$

where σ^2 is the variance of Y.

The following proposition has been taken from [Larrañaga (2003)].

Proposition 3.7 *Let $\boldsymbol{X} = (\boldsymbol{Y}, \boldsymbol{Z}) = (X_1, ..., X_{n+m})$ be an $(n+m)$-dimensional mixed random variable, where $\boldsymbol{Y} = (Y_1, ..., Y_n)$ is an n-dimensional continuous random variable and $\boldsymbol{Z} = (Z_1, ..., Z_m)$ is an m-dimensional discrete random variable. If $\boldsymbol{X}$ follows an* $(n + m)$-dimensional mixed Gaussian distribution *with parameters $p(\boldsymbol{z})$, $\boldsymbol{\mu}_{\boldsymbol{Y}}(\boldsymbol{z})$ and $\Sigma_{\boldsymbol{Y}}(\boldsymbol{z})$, $\rho(\boldsymbol{x}) \rightsquigarrow \mathcal{MG}(\boldsymbol{x}; p(\boldsymbol{z}),$ $\boldsymbol{\mu}_{\boldsymbol{Y}}(\boldsymbol{z}), \Sigma_{\boldsymbol{Y}}(\boldsymbol{z}))$, the conditional entropy of $\boldsymbol{Y}$ conditioned to $\boldsymbol{Z}$ is given by:*

$$h(\boldsymbol{Y}|\boldsymbol{Z}) = \frac{n}{2}\log(2\pi e) + \frac{1}{2}\sum_{\boldsymbol{z}} p(\boldsymbol{z}) \log(|\Sigma_{\boldsymbol{Y}}(\boldsymbol{z})|)$$

Proof. The proposition can be proved as follows:

$$\begin{aligned} h(\boldsymbol{Y}|\boldsymbol{Z}) &= E_{\boldsymbol{Z}}[h(\boldsymbol{Y}|\boldsymbol{z})] \\ &= \sum_{\boldsymbol{z}} p(\boldsymbol{z}) \frac{1}{2}\log(|\Sigma_{\boldsymbol{Y}}(\boldsymbol{z})|(2\pi e)^n) \\ &= \frac{n}{2}\log(2\pi e) + \frac{1}{2}\sum_{\boldsymbol{z}} p(\boldsymbol{z}) \log(|\Sigma_{\boldsymbol{Y}}(\boldsymbol{z})|) \end{aligned}$$

where the first equality holds by Definition 1.50 and, the second equality holds due to Definition 3.2 and Proposition 3.5. ■

In the particular case of $n = m = 1$, the conditional entropy is computed as

$$h(Y|Z) = \frac{1}{2}\log(2\pi e) + \frac{1}{2}\sum_{z} p(z) \log(\sigma^2(z))$$

where $\sigma^2(z)$ is the variance of Y when $Z = z$.

Proposition 3.8 *Let $\boldsymbol{Y} = (Y_1, ..., Y_n) = (\boldsymbol{U}, \bar{\boldsymbol{U}})$ be an n-dimensional continuous random variable, where $\boldsymbol{U} = (U_1, ..., U_k)$ is a k-dimensional continuous random variable. If $\boldsymbol{Y}$ follows a Gaussian density function with parameters $\boldsymbol{\mu}$ and Σ, $f(\boldsymbol{y}) \rightsquigarrow \mathcal{N}(\boldsymbol{y}; \boldsymbol{\mu}, \Sigma)$, the conditional entropy of $\boldsymbol{U}$ given $\bar{\boldsymbol{U}}$ is given by:*

$$h(\boldsymbol{U}|\bar{\boldsymbol{U}}) = \frac{k}{2}\log(2\pi e) + \frac{1}{2}\log(|\Sigma_{\boldsymbol{U}|\bar{\boldsymbol{U}}}|)$$

where

$$\Sigma_{\boldsymbol{U}|\bar{\boldsymbol{U}}} = \Sigma_{\boldsymbol{U}} - \Sigma_{\boldsymbol{U},\bar{\boldsymbol{U}}}\Sigma_{\bar{\boldsymbol{U}}}^{-1}\Sigma_{\bar{\boldsymbol{U}},\boldsymbol{U}}$$

Proof. The proposition can be proved as follows:

$$\begin{aligned} h(\boldsymbol{U}|\bar{\boldsymbol{U}}) &= E_{\bar{\boldsymbol{U}}}[H(\boldsymbol{U}|\bar{\boldsymbol{U}})] \\ &= \int_{-\infty}^{\infty} \frac{k}{2}\log(2\pi e) + \frac{1}{2}\log(|\Sigma_{\boldsymbol{U}|\bar{\boldsymbol{U}}}|) d\bar{\boldsymbol{u}} \\ &= \frac{k}{2}\log(2\pi e) + \frac{1}{2}\log(|\Sigma_{\boldsymbol{U}|\bar{\boldsymbol{U}}}|) \end{aligned}$$

where the first equality holds due to Equation 1.50 and the second is given by Proposition 3.7 and Proposition 3.6. ■

It should be noted that the determinant can be also computed as follows:

$$|\Sigma_{\boldsymbol{U}|\bar{\boldsymbol{U}}}| = \frac{|\Sigma|}{|\Sigma_{\bar{\boldsymbol{U}}}|}$$

In the particular case of $n = 2$ and $k = 1$, $\boldsymbol{Y} = (U, \bar{U})$, the conditional entropy is computed as

$$h(U|\bar{U}) = \frac{1}{2}\log[(2\pi e) + \frac{\sigma_U^2\sigma_{\bar{U}}^2 - \sigma_{U,\bar{U}}^2}{\sigma_{\bar{U}}^2}]$$

where $\sigma_{U,\bar{U}}$ is the covariance of U and $\bar{U}$, and $\sigma_{U,\bar{U}}^2 = \sigma_{\bar{U},U}^2$.

Proposition 3.9 *Let $\boldsymbol{X} = (\boldsymbol{Y}, \boldsymbol{Z}) = (X_1, ..., X_{n+m})$ be an $(n+m)$-dimensional mixed random variable, being $\boldsymbol{Y} = (Y_1, ..., Y_n)$ an n-dimensional continuous random variable and $\boldsymbol{Z} = (Z_1, ..., Z_m)$ an m-dimensional discrete random variable. If $\boldsymbol{X}$ follows an $(n + m)$-dimensional mixed Gaussian distribution, $\rho(\boldsymbol{x}) \rightsquigarrow \mathcal{MG}(\boldsymbol{x}; p(\boldsymbol{z}), \boldsymbol{\mu}_{\boldsymbol{Y}}(\boldsymbol{z}), \Sigma_{\boldsymbol{Y}}(\boldsymbol{z}))$, then the entropy of $\boldsymbol{X}$ is given by:*

$$H(\boldsymbol{X}) = \frac{n}{2}\log(2\pi e) + \sum_{\boldsymbol{z}} p(\boldsymbol{z})\log(\frac{\sqrt{|\Sigma_{\boldsymbol{Y}}(\boldsymbol{z})|}}{p(\boldsymbol{z})})$$

Proof. The proposition is proved as follows:

$$\begin{aligned} H(\boldsymbol{X}) &= h(\boldsymbol{Y}|\boldsymbol{Z}) + H(\boldsymbol{Z}) \\ &= \frac{n}{2}\log(2\pi e) + \frac{1}{2}\sum_{\boldsymbol{z}} p(\boldsymbol{z})\log(|\Sigma_{\boldsymbol{Y}}(\boldsymbol{z})|) \\ &\quad - \sum_{\boldsymbol{z}} p(\boldsymbol{z})\log p(\boldsymbol{z}) \\ &= \frac{n}{2}\log(2\pi e) + \sum_{\boldsymbol{z}} p(\boldsymbol{z})\log(\frac{\sqrt{|\Sigma_{\boldsymbol{Y}}(\boldsymbol{z})|}}{p(\boldsymbol{z})}) \end{aligned}$$

where the first equality holds by Equation 1.51, and the second one holds by Proposition 3.7 and Equation 1.46. ■

It should be noted that Theorem 1.46 and Proposition 3.6 are particular cases of this theorem. In the particular case of $n = m = 1$, the entropy is computed as

$$H(\boldsymbol{X}) = H(Y, Z) = \frac{1}{2}\log(2\pi e) + \sum_{z} p(z)\log(\frac{\sigma(z)}{p(z)}) \tag{3.2}$$

where $\sigma^2(z)$ is the variance of Y when $Z = z$.

The following proposition has been taken from [Cover and Thomas (1991)].

Proposition 3.10 *Let $\boldsymbol{Y} = (Y_1, ..., Y_m) = (\boldsymbol{U}, \bar{\boldsymbol{U}})$ be a multidimensional continuous random variable, where $\boldsymbol{U} = (U_1, ..., U_k)$. If it is verified that $f(\boldsymbol{y}) \rightsquigarrow \mathcal{N}(\boldsymbol{y}; \boldsymbol{\mu_Y}, \Sigma_{\boldsymbol{Y}})$, the mutual information of $\boldsymbol{U}$ and $\bar{\boldsymbol{U}}$ is given by:*

$$I(\boldsymbol{U}; \bar{\boldsymbol{U}}) = \frac{1}{2} \log \frac{|\Sigma_{\boldsymbol{U}}||\Sigma_{\bar{\boldsymbol{U}}}|}{|\Sigma_{\boldsymbol{Y}}|} \tag{3.3}$$

where $\Sigma_{\boldsymbol{U}}$ and $\Sigma_{\bar{\boldsymbol{U}}}$ are the sub-matrixes of $\Sigma_{\boldsymbol{Y}}$ corresponding to $\boldsymbol{U}$ and $\bar{\boldsymbol{U}}$ respectively.

Proof.

$$\begin{aligned} I(\boldsymbol{U}, \bar{\boldsymbol{U}}) &= H(\boldsymbol{U}) + H(\bar{\boldsymbol{U}}) - H(\boldsymbol{U}, \bar{\boldsymbol{U}}) \\ &= \frac{k}{2} \log(2\pi e) + \frac{1}{2} \log(|\Sigma_{\boldsymbol{U}}|) \\ &\quad + \frac{n-k}{2} \log(2\pi e) + \frac{1}{2} \log(|\Sigma_{\bar{\boldsymbol{U}}}|) \\ &\quad - \frac{n}{2} \log(2\pi e) + \frac{1}{2} \log(|\Sigma_{\boldsymbol{Y}}|) \\ &= \frac{1}{2} \log \frac{|\Sigma_{\boldsymbol{U}}||\Sigma_{\bar{\boldsymbol{U}}}|}{|\Sigma_{\boldsymbol{Y}}|} \end{aligned}$$

where the first equality holds due to Equation 1.55 and the second equality is given by Proposition 3.6. The proposition is then demonstrated by simple algebraic manipulations. ■

Note that, contrary to the case of the entropy, in the formulation of mutual information under Gaussian assumption, the term $\log(2\pi e)$ does not appear. In the particular case of $n = 2$ and $k = 1$, the mutual information is computed as

$$\begin{aligned} I(U; \bar{U}) &= \frac{1}{2} \log \frac{\sigma_U^2 \sigma_{\bar{U}}^2}{\sigma_{U,\bar{U}}^2 - \sigma_U \sigma_{\bar{U}}} \\ &= -\frac{1}{2} \log(1 - \rho^2(U, \bar{U})) \end{aligned}$$

where $\rho(U, \bar{U}) = \frac{\sigma_{U,\bar{U}}}{\sqrt{\sigma_U^2 \sigma_{\bar{U}}^2}}$ is the linear correlation coefficient between U and $\bar{U}$.

The following proposition has been adapted from [Larrañaga (2003)] to multidimensional random variables.

Proposition 3.11 *Let $\boldsymbol{Y} = (Y_1, ..., Y_n)$ be an n-dimensional continuous random variable $\boldsymbol{Z} = (Z_1, ..., Z_m)$ be an m-dimensional discrete random variable. If it is verified that $f(\boldsymbol{y}) \rightsquigarrow \mathcal{N}(\boldsymbol{y}; \boldsymbol{\mu_Y}, \Sigma_{\boldsymbol{Y}})$ and $f(\boldsymbol{y}|\boldsymbol{z}) \rightsquigarrow \mathcal{N}(\boldsymbol{y}; \boldsymbol{\mu_Y}(\boldsymbol{z}), \Sigma_{\boldsymbol{Y}}(\boldsymbol{z}))$, then the mutual information of $\boldsymbol{Y}$ and $\boldsymbol{Z}$ is given by:*

$$I(\boldsymbol{Y}; \boldsymbol{Z}) = \frac{1}{2} [\log |\Sigma_{\boldsymbol{Y}}| - \sum_{\boldsymbol{z}} p(\boldsymbol{z}) \log |\Sigma_{\boldsymbol{Y}}(\boldsymbol{z})|] \tag{3.4}$$

Proof. This proposition can be proved as follows

$$\begin{aligned} I(\boldsymbol{Y};\boldsymbol{Z}) &= H(\boldsymbol{Y}) - H(\boldsymbol{Y}|\boldsymbol{Z}) \\ &= \frac{n}{2}\log(2\pi e) + \frac{1}{2}\log|\Sigma_{\boldsymbol{Y}}| - \frac{n}{2}(2\pi e) - \frac{1}{2}\sum_{\boldsymbol{z}} p(\boldsymbol{z})\log|\Sigma_{\boldsymbol{Y}}(\boldsymbol{z})| \\ &= \frac{1}{2}[\log|\Sigma_{\boldsymbol{Y}}| - \sum_{\boldsymbol{z}} p(\boldsymbol{z})\log|\Sigma_{\boldsymbol{Y}}(\boldsymbol{z})|] \end{aligned}$$

where the first equality is given by Equation 1.54 and the second one holds due to Propositions 3.6 and 3.7. ■

It should be noted that the assumptions of this proposition are not fulfilled in general. Moreover, by the converse to Proposition 3.4, they are fulfilled if and only if $\boldsymbol{Y}$ and $\boldsymbol{Z}$ are independent and, therefore, the mutual information is zero. However, if the parameters $\Sigma_{\boldsymbol{Y}}$ and $\Sigma_{\boldsymbol{Y}}(\boldsymbol{z})$ are estimated directly from data, for example using maximum likelihood estimator, this quantity could be used as an initial approach to the general case when (Y, Z) is distributed according to a mixed Gaussian distribution with general $\Sigma_{\boldsymbol{Y}}(\boldsymbol{z})$ parameter. This quantity depends on the scale of the implied continuous variables and, thus, it is advisable to perform a normalization of the variables in order to transform them into variables with the same dispersion, e.g. same typical deviation.

For the particular case of $n = m = 1$ [Larrañaga (2003)], $\boldsymbol{X} = (Y, Z)$, the mixed mutual information is given by:

$$I(Y;Z) = \frac{1}{2}[\log(\sigma^2_Y) - \sum_z p(z)\log(\sigma^2_Y(z))] \tag{3.5}$$

The following proposition has been adapted from [Larrañaga (2003)] to multidimensional random variables.

Proposition 3.12 *Let $\boldsymbol{X} = (X_1, ..., X_{n+m}) = (\boldsymbol{Y}, \boldsymbol{Z})$ be an $(n + m)$-dimensional mixed random variable, being $\boldsymbol{Y} = (Y_1, ..., Y_n) = (\boldsymbol{U}, \bar{\boldsymbol{U}})$ an n-dimensional continuous random variable, where $\boldsymbol{U} = (U_1, ..., U_k)$ with $k \geq 1$, and $\boldsymbol{Z} = (Z_1, ..., Z_m)$ an m-dimensional discrete random variable with $m \geq 1$. If $\boldsymbol{X}$ follows an $(n + m)$-dimensional mixed Gaussian distribution, $\rho(\boldsymbol{x}) \rightsquigarrow \mathcal{MG}(\boldsymbol{x}; p(\boldsymbol{z}), \Sigma(\boldsymbol{z}), \boldsymbol{\mu}(\boldsymbol{z}))$, then the mutual information of $\boldsymbol{U}$ and $\bar{\boldsymbol{U}}$ given $\boldsymbol{Z}$ is computed as:*

$$I(\boldsymbol{U};\bar{\boldsymbol{U}}|\boldsymbol{Z}) = \frac{1}{2}\sum_{\boldsymbol{z}} p(\boldsymbol{z})\log\frac{|\Sigma_{\boldsymbol{U}}(\boldsymbol{z})||\Sigma_{\bar{\boldsymbol{U}}}(\boldsymbol{z})|}{|\Sigma(\boldsymbol{z})|} \tag{3.6}$$

where $\Sigma_{\boldsymbol{U}}(\boldsymbol{z}) = \Sigma(\boldsymbol{z})_{\boldsymbol{U}}$ and $\Sigma_{\bar{\boldsymbol{U}}}(\boldsymbol{z}) = \Sigma(\boldsymbol{z})_{\bar{\boldsymbol{U}}}$.

Proof. The proposition can be demonstrated as follows:

$$
\begin{aligned}
I(\boldsymbol{U};\bar{\boldsymbol{U}}|\boldsymbol{Z}) &= E_{\boldsymbol{Z}}[I(\boldsymbol{U};\bar{\boldsymbol{U}}|\boldsymbol{z})] \\
&= \sum_{\boldsymbol{z}} p(\boldsymbol{z}) \frac{1}{2} \log \frac{|\Sigma_{\boldsymbol{U}}(\boldsymbol{z})||\Sigma_{\bar{\boldsymbol{U}}}(\boldsymbol{z})|}{|\Sigma(\boldsymbol{z})|} \\
&= \frac{1}{2} \sum p(\boldsymbol{z}) \log \frac{|\Sigma_{\boldsymbol{U}}(\boldsymbol{z})||\Sigma_{\bar{\boldsymbol{U}}}(\boldsymbol{z})|}{|\Sigma(\boldsymbol{z})|}
\end{aligned}
$$

where the first equality is given by Equation 1.57 and the second equality holds by Propositions 3.10, 3.3 and 3.5. ■

In the particular case of $n = 2$, $m = k = 1$ [Larrañaga (2003)] the conditional mutual information can be computed as

$$
\begin{aligned}
I(U;\bar{U}|Z) &= \frac{1}{2} \sum_{z} p(z) \log \frac{\sigma_U^2(z)\sigma_{\bar{U}}^2(z)}{|\Sigma(z)|} \\
&= -\frac{1}{2} \sum_{z} p(z)(1 - \rho_z^2(U,\bar{U})) \qquad (3.7)
\end{aligned}
$$

where $\rho_z(U,\bar{U}) = \frac{\sigma_{U,\bar{U}}(z)}{\sqrt{\sigma_U^2 \sigma_{\bar{U}}^2}}$ is the conditional linear correlation coefficient between U and $\bar{U}$ when $Z = z$.

The following proposition has been adapted from [Larrañaga (2003)] to multidimensional random variables.

Proposition 3.13 *Let $\boldsymbol{Y} = (Y_1, ..., Y_n)$ be an n-dimensional continuous random variable and $\boldsymbol{Z} = (Z_1, ..., Z_m) = (\boldsymbol{V}, \bar{\boldsymbol{V}})$ be an m-dimensional discrete random variable, where $\boldsymbol{V} = (V_1, ..., V_l)$ with $l \geq 1$ and $m \geq 2$. If it is verified that $f(\boldsymbol{y}|\bar{\boldsymbol{v}}) \rightsquigarrow \mathcal{N}(\boldsymbol{y}; \boldsymbol{\mu}_{\boldsymbol{Y}}(\bar{\boldsymbol{v}}), \Sigma_{\boldsymbol{Y}}(\bar{\boldsymbol{v}}))$, $f(\boldsymbol{y}|\boldsymbol{z}) \rightsquigarrow \mathcal{N}(\boldsymbol{y}; \boldsymbol{\mu}_{\boldsymbol{Y}}(\boldsymbol{z}), \Sigma_{\boldsymbol{Y}}(\boldsymbol{z}))$, the conditional mutual information of $\boldsymbol{Y}$ and $\boldsymbol{V}$ given $\bar{\boldsymbol{V}}$ is computed as:*

$$
I(\boldsymbol{Y};\boldsymbol{V}|\bar{\boldsymbol{V}}) = \frac{1}{2}[\sum_{\bar{\boldsymbol{v}}} p(\bar{\boldsymbol{v}}) \log(|\Sigma_{\boldsymbol{Y}}(\bar{\boldsymbol{v}})|) - \sum_{\boldsymbol{z}} p(\boldsymbol{z}) \log(|\Sigma_{\boldsymbol{Y}}(\boldsymbol{z})|)]
$$

Proof. Given Equation 1.54 and, Propositions 3.6 and 3.7 is demonstrated that

$$
\begin{aligned}
I(\boldsymbol{Y};\boldsymbol{V}|\bar{\boldsymbol{V}}) &= H(\boldsymbol{Y}|\bar{\boldsymbol{V}}) - H(\boldsymbol{Y}|\boldsymbol{V},\bar{\boldsymbol{V}}) \\
&= \frac{n}{2}\log(2\pi e) + \frac{1}{2}\sum_{\bar{\boldsymbol{v}}} p(\bar{\boldsymbol{v}}) \log(|\Sigma_{\boldsymbol{Y}}(\bar{\boldsymbol{v}})|) \\
&- \frac{n}{2}\log(2\pi e) - \frac{1}{2}\sum_{\boldsymbol{z}} p(\boldsymbol{z}) \log(|\Sigma_{\boldsymbol{Y}}(\boldsymbol{z})|) \\
&= \frac{1}{2}[\sum_{\bar{\boldsymbol{v}}} p(\bar{\boldsymbol{v}}) \log(|\Sigma_{\boldsymbol{Y}}(\bar{\boldsymbol{v}})|) - \sum_{\boldsymbol{z}} p(\boldsymbol{z}) \log(|\Sigma_{\boldsymbol{Y}}(\boldsymbol{z})|)]
\end{aligned}
$$

■

It should be noted that the assumptions of this proposition are not fulfilled in general. Moreover, by the converse to Proposition 3.4, they are fulfilled if and only if $\boldsymbol{Y}$ and $\boldsymbol{V}$ are conditional independent given $\bar{\boldsymbol{V}}$, $CI(\boldsymbol{X};\boldsymbol{V}|\bar{\boldsymbol{V}})$, and, therefore, the conditional mutual information is zero. However, if the parameters $\Sigma_{\boldsymbol{Y}}(\bar{\boldsymbol{v}})$ and $\Sigma_{\boldsymbol{Y}}(\boldsymbol{z})$ are estimated directly from data, for example using maximum likelihood estimator, this quantity could be used as an initial approach to the general case when (Y, Z) is distributed according to a mixed Gaussian distribution with general $\Sigma_{\boldsymbol{Y}}(\boldsymbol{z})$ parameter (see [Lauritzen (1996)], Lemma 6.4, p. 162, for theoretical properties of the weak marginal) . This quantity depends on the scale of the implied continuous variables and, thus, it is advisable to transform the continuous random variables into variables with the same dispersion, e.g. same typical deviation.

For the particular case of $n = 1$, $m = 2$ and $l = 1$ [Larrañaga (2003)] the mixed mutual information is given by:

$$I(Y;V|\bar{V}) = \frac{1}{2}[\sum_{\bar{v}} \log(\sigma^2_Y(\bar{v})) - \sum_{z} p(z)\log(\sigma^2_Y(z))] \tag{3.8}$$

Proposition 3.14 *Let $\boldsymbol{Y} = (Y_1, ..., Y_n)$ be a n-multidimensional continuous random variable. If it is verified that $f(\boldsymbol{y}) \rightsquigarrow \mathcal{N}(\boldsymbol{y}; \boldsymbol{\mu}_{\boldsymbol{Y}}, \Sigma_{\boldsymbol{Y}})$, the n-way interaction information of $\{Y_1, ..., Y_n\}$ is given by:*

$$I(Y_1; ...; Y_n) = -\frac{1}{2}\sum_{\boldsymbol{U}\subseteq\boldsymbol{Y}} (-1)^{n-|\boldsymbol{U}|}\log(|\Sigma_{\boldsymbol{U}}|)$$

Proof. The proposition can be demonstrated as follows:

$$\begin{aligned}
I(Y_1, ..., Y_n) &= -\sum_{\boldsymbol{U}\subseteq\boldsymbol{Y}} (-1)^{n-|\boldsymbol{U}|} H(\boldsymbol{U}) \\
&= -\sum_{\boldsymbol{U}\subseteq\boldsymbol{Y}} (-1)^{n-|\boldsymbol{U}|}\frac{1}{2}\log[(2\pi e)^{|\boldsymbol{U}|}|\Sigma_{\boldsymbol{U}}|] \\
&= -\sum_{\boldsymbol{U}\subseteq\boldsymbol{Y}} (-1)^{n-|\boldsymbol{U}|}\frac{1}{2}\log(2\pi e)^{|\boldsymbol{U}|} \\
&\quad -\sum_{\boldsymbol{U}\subseteq\boldsymbol{Y}} (-1)^{n-|\boldsymbol{U}|}\frac{1}{2}\log(|\Sigma_{\boldsymbol{U}}|) \\
&= -\frac{1}{2}\sum_{\boldsymbol{U}\subseteq\boldsymbol{Y}} (-1)^{n-|\boldsymbol{U}|}\log(|\Sigma_{\boldsymbol{U}}|)
\end{aligned}$$

The first equality is given by the definition of n-way interaction information (see Equation 1.58). The second equality holds due to Propositions 3.1 and 3.6. The third equality holds because $\sum_{\boldsymbol{U}\subseteq\boldsymbol{Y}}(-1)^{n-|\boldsymbol{U}|}\frac{1}{2}\log(2\pi e)^{|\boldsymbol{U}|} = 0$ when $n > 1$, and the proposition is demonstrated. ■

Note that the generalization of this proposition to multidimensional $\boldsymbol{Y}_i$ random variables is straightforward.

In the particular case of $n = 3$, the 3-way interaction information can be computed as

$$I(Y_1; Y_2; Y_3) = \frac{1}{2} \log \frac{|\Sigma_{(Y_1,Y_2)}||\Sigma_{(Y_1,Y_3)}||\Sigma_{(Y_2,Y_3)}|}{|\Sigma_{(Y_1,Y_2,Y_3)}|\sigma^2_{Y_1}\sigma^2_{Y_2}\sigma^2_{Y_3}} \tag{3.9}$$

3.4 Conditional Gaussian networks

This section presents the second class of PGMs we are interested in, *conditional Gaussian networks* (*CGN*). BMNs deal with pure discrete discrete random variables. However, we are often faced with *mixed data*, i.e. domains where every case represents an assignment of a state to a mixed random variable. This section introduces conditional Gaussian networks, a class of PGMs that is able to deal directly with continuous and discrete random variables without discretizing the continuous random variables.

In order to distinguish the discrete and continuous nature of the parents of a node we introduce the following notation for the remainder of the dissertation. The set of all the continuous and discrete parents of a given node X_i in $\boldsymbol{s}$ is represented by $\boldsymbol{Pc}(\boldsymbol{s})_i = \boldsymbol{Pa}(\boldsymbol{s})_i \cap \boldsymbol{Y}$ and $\boldsymbol{Pd}(\boldsymbol{s})_i = \boldsymbol{Pa}(\boldsymbol{s})_i \cap \boldsymbol{Z}$ for all i, respectively. Both representations, $\boldsymbol{Pc}(\boldsymbol{s})_i$ and $\boldsymbol{Pd}(\boldsymbol{s})_i$, can be replaced by $\boldsymbol{Pc}_i$ and $\boldsymbol{Pd}_i$ when the context makes clear the structure $\boldsymbol{s}$ that is being considered.

Let us consider a PGM for an $(n + m)$-dimensional mixed random variable $\boldsymbol{X} = (X_1, ..., X_{n+m})$ with the proper subsets $\boldsymbol{Y} = (Y_1, ..., Y_n) = (X_{1:\boldsymbol{Y}}, ..., X_{n:\boldsymbol{Y}})$ and $\boldsymbol{Z} = (Z_1, ..., Z_m) = (X_{1:\boldsymbol{Z}}, ..., X_{m:\boldsymbol{Z}})$ as described at the beginning of Section 3.2. If (i) no discrete random variables have continuous parents, $\boldsymbol{Pc}(\boldsymbol{s})_{i:\boldsymbol{Z}} = \emptyset$ for $i = 1, ..., m$, and (ii) the variable $\boldsymbol{X}$ follows a mixed Gaussian distribution, $\rho(\boldsymbol{x}) \rightsquigarrow \mathcal{MG}(\boldsymbol{x}; p(\boldsymbol{z}), \boldsymbol{\mu}(\boldsymbol{z}), \Sigma(\boldsymbol{z}))$ (see Definition 3.2), then the PGM for $\boldsymbol{X}$ is called *conditional Gaussian network* for $\boldsymbol{X}$ [Lauritzen and Jensen (2001); Bottcher (2004)]. The structural constraints imposed to CGN paradigm implies that any structure codified by a CGN paradigm are decomposable (see [Lauritzen (1992)], p. 1101, and [Cowell (2001)], p.131, Definition 7.7).

The graphical factorization of the generalized joint probability distribution for $\boldsymbol{X}$ encoded by a CGN for $\boldsymbol{X}$ acquires the following form:

$$\begin{aligned}\rho(\boldsymbol{x}|\boldsymbol{\Theta_s}, \boldsymbol{s}) &= \rho(x_1, ..., x_n|\boldsymbol{\Theta_s}, \boldsymbol{s}) \\ &= \rho(\boldsymbol{y}, \boldsymbol{z}|\boldsymbol{\Theta_s}, \boldsymbol{s}) \\ &= \prod_{i=1}^{n} f(y_i|\boldsymbol{pa}(\boldsymbol{s})_{i:\boldsymbol{Y}}, \boldsymbol{\Theta}_{i:\boldsymbol{Y}}) \prod_{j=1}^{m} p(z_j|\boldsymbol{pa}(\boldsymbol{s})_{j:\boldsymbol{Z}}, \boldsymbol{\Theta}_{j:\boldsymbol{Z}}) \\ &= \prod_{i=1}^{n} f(y_i|\boldsymbol{pc}(\boldsymbol{s})_{i:\boldsymbol{Y}}, \boldsymbol{pd}(\boldsymbol{s})_{i:\boldsymbol{Y}}, \boldsymbol{\Theta}_{i:\boldsymbol{Y}}) \\ &\quad \cdot \prod_{j=1}^{m} p(z_j|\boldsymbol{pd}(\boldsymbol{s})_{j:\boldsymbol{Z}}, \boldsymbol{\Theta}_{j:\boldsymbol{Z}}) \end{aligned} \tag{3.10}$$

where $\boldsymbol{\Theta_s} = (\boldsymbol{\Theta}_1, ..., \boldsymbol{\Theta}_{n+m})$ and $\boldsymbol{s}$ are, as defined before, the model parameters and the model structure respectively. Note that Equation 3.10 is a particularization of Equation 2.2, which takes into account the continuous and discrete nature of $\boldsymbol{Y} = (X_{1:\boldsymbol{Y}}, ..., X_{n:\boldsymbol{Y}})$ and $\boldsymbol{Z} = (X_{1:\boldsymbol{Z}}, ..., X_{m:\boldsymbol{Z}})$, respectively, and the structural constraint (i), $\boldsymbol{Pc}(\boldsymbol{s})_{i:\boldsymbol{Z}} = \emptyset$ for $i = 1, ..., m$.

It should be noted that by the structural constraints imposed to the CGN paradigm, if the local factors of X_i given $\boldsymbol{pa}(\boldsymbol{s})_i$ for $i = 1, ..., m + n$ follow a MG distribution, then the joint probability distribution for $\boldsymbol{X}$ encoded by the CGN follows MG distribution (see [Lauritzen and Jensen (2001)]). In order to obtain the parameters of the joint MG distribution for $\boldsymbol{X}$, $\boldsymbol{\mu_Y}(\boldsymbol{z})$ and $\Sigma_{\boldsymbol{Y}}(\boldsymbol{z})$ for all $\boldsymbol{z}$, encoded by a CGN for $\boldsymbol{X}$ with parameters $\boldsymbol{\Theta_s} = (\boldsymbol{\Theta}_1, ..., \boldsymbol{\Theta}_{n+m})$, the reader may consult the properties of the conditional Gaussian potentials* provided in [Cowell et al. (1999)], Section 7, and [Lauritzen and Jensen (2001)], Section 4.

Typically, the local probability distributions for every discrete random variable $Z_i = X_{i:\boldsymbol{Z}}$, are restricted to be a set of univariate multinomial distributions, one for each value of $\boldsymbol{Pd}(\boldsymbol{s})_{i:\boldsymbol{Z}}$, $\boldsymbol{pd}(\boldsymbol{s})_{i:\boldsymbol{Z}}$. Let us assume that Z_i can take r_i distinct values, denoted by $z_i^1, ..., z_i^{r_i}$, and that $\boldsymbol{Pd}(\boldsymbol{s})_{i:\boldsymbol{Z}}$ can have q_i distinct states denoted by $\boldsymbol{pd}(\boldsymbol{s})_{i:\boldsymbol{Z}}^1, ..., \boldsymbol{pd}(\boldsymbol{s})_{i:\boldsymbol{Z}}^{q_i}$ with $q_i = \prod_{Z_l \in \boldsymbol{Pa}(\boldsymbol{s})_{i:\boldsymbol{Z}}} r_l$ for all $i = 1, ..., m$. Then, the univariate multinomial distribution $p(z_i|\boldsymbol{pd}(\boldsymbol{s})_{i:\boldsymbol{Z}}^j, \boldsymbol{\Theta}_{i:\boldsymbol{Z}})$ for all $i = 1, ..., m$ consists of a set of probabilities of the form

$$\begin{aligned} p(z_i^k|\boldsymbol{pd}(\boldsymbol{s})_{i:\boldsymbol{Z}}^j, \boldsymbol{\Theta}_{i:\boldsymbol{Z}}) &= \Theta_{z_i^k|\boldsymbol{pd}(Z_i)_{\boldsymbol{s}}^j} \\ &= \Theta_{i:\boldsymbol{Z}}^{jk} = Pr(Z_i = z_i^k|\boldsymbol{Pd}(\boldsymbol{s})_{i:\boldsymbol{Z}} = \boldsymbol{pd}(\boldsymbol{s})_{i:\boldsymbol{Z}}^j) \end{aligned} \tag{3.11}$$

such that $\Theta_{i:\boldsymbol{Z}}^{jk} > 0$ representing the conditional probability that $Z_i = X_{i:\boldsymbol{Z}}$ takes its k-th state given that $\boldsymbol{Pd}(\boldsymbol{s})_{i:\boldsymbol{Z}}$ takes its j-th value, for $k = 1, ..., r_i$. Moreover, $\sum_{k=1}^{r_i} \Theta_{i:\boldsymbol{Z}}^{jk} = 1$ for $i = 1, ..., m$ and $1 \leq j \leq q_i$. Consequently, the parameters of the local probability distributions for every discrete random variable $Z_i = X_{i:\boldsymbol{Z}}$ are given by $\boldsymbol{\Theta}_{i:\boldsymbol{Z}} = (\boldsymbol{\Theta}_{i:\boldsymbol{Z}}^j)_{j=1}^{q_i}$ with $\boldsymbol{\Theta}_{i:\boldsymbol{Z}}^j = (\Theta_{i:\boldsymbol{Z}}^{jk})_{k=1}^{r_i}$ for all $j \leq q_i$. Using Propositions 3.4 and 3.5, the parameters $\boldsymbol{\Theta}_{i:\boldsymbol{Z}}$ for $i = 1, ..., m$ can be obtained from $p(\boldsymbol{z})$ as follows:

* i.e. potentials associated to the MG distributions

$$\Theta_{i:\boldsymbol{Z}}^{jk} = \frac{\sum_{l=1}^{o_i} p(z_i^k, \boldsymbol{pd}(\boldsymbol{s})_{i:\boldsymbol{Z}}^k, \neg \boldsymbol{pd}(\boldsymbol{s})_{i:\boldsymbol{Z}}^l)}{\sum_{l=1}^{o_i} \sum_{k=1}^{r_i} p(z_i^k, \boldsymbol{pd}(\boldsymbol{s})_{i:\boldsymbol{Z}}^k, \neg \boldsymbol{pd}(\boldsymbol{s})_{i:\boldsymbol{Z}}^l)} \tag{3.12}$$

where $\neg \boldsymbol{Pd}(\boldsymbol{s})_{i:\boldsymbol{Z}} = \boldsymbol{Z} \setminus \{Z_i, \boldsymbol{Pd}(\boldsymbol{s})_{i:\boldsymbol{Z}}\}$ and $o_i = \prod_{Z_j \in \neg \boldsymbol{Pd}(\boldsymbol{s})_{i:\boldsymbol{Z}}} r_j$ is the number of states of $\neg \boldsymbol{Pd}(\boldsymbol{s})_{i:\boldsymbol{Z}}$. The number of parameters required for determining the conditional distribution $p(z_i | \boldsymbol{pd}(\boldsymbol{s})_{i:\boldsymbol{Z}})$ is $(\prod_{Z_j \in \{Z_i, \boldsymbol{Pd}(\boldsymbol{s})_{i:\boldsymbol{Z}}\}} r_j) - 1$ and assuming that $r_j = r$ for any j the number of required parameters is $\mathcal{O}(r^{|\boldsymbol{Pd}(\boldsymbol{s})_{i:\boldsymbol{Z}}|+1})$.

On the other hand, the probability density function $f(y_i | \boldsymbol{pc}(\boldsymbol{s})_{i:\boldsymbol{Y}}, \boldsymbol{pd}(\boldsymbol{s})_{i:\boldsymbol{Y}}, \boldsymbol{\Theta}_{i:\boldsymbol{Y}})$ given a value $\boldsymbol{pa}(\boldsymbol{s})_{i:\boldsymbol{Y}}$ follows a Gaussian distribution for all $i = 1, ..., n$

$$\begin{aligned} f(y_i \mid & \boldsymbol{pc}(\boldsymbol{s})_{i:\boldsymbol{Y}}, \boldsymbol{pd}(\boldsymbol{s})_{i:\boldsymbol{Y}}, \boldsymbol{\Theta}_{i:\boldsymbol{Y}}) \\ & \rightsquigarrow \mathcal{N}(y_i; \mu_{Y_i | \boldsymbol{pc}(\boldsymbol{s})_{i:\boldsymbol{Y}}}(\boldsymbol{pd}(\boldsymbol{s})_{i:\boldsymbol{Y}}), \sigma^2_{Y_i | \boldsymbol{Pc}(\boldsymbol{s})_{i:\boldsymbol{Y}}}(\boldsymbol{pd}(\boldsymbol{s})_{i:\boldsymbol{Y}})) \end{aligned} \tag{3.13}$$

where

$$\begin{aligned} \mu_{Y_i | \boldsymbol{pc}(\boldsymbol{s})_{i:\boldsymbol{Y}}}(\boldsymbol{pd}(\boldsymbol{s})_{i:\boldsymbol{Y}}) = & \boldsymbol{\mu}(\boldsymbol{pd}(\boldsymbol{s}))_{Y_i} \\ & + \Sigma(\boldsymbol{pd}(\boldsymbol{s})_{i:\boldsymbol{Y}})_{Y_i, \boldsymbol{Pc}(\boldsymbol{s})_{i:\boldsymbol{Y}}} \\ & \cdot \Sigma(\boldsymbol{pd}(\boldsymbol{s})_{i:\boldsymbol{Y}})^{-1}_{\boldsymbol{Pc}(\boldsymbol{s})_{i:\boldsymbol{Y}}} \\ & \cdot (\boldsymbol{pc}(\boldsymbol{s})_{i:\boldsymbol{Y}} - \boldsymbol{\mu}(\boldsymbol{pd}(\boldsymbol{s})_{i:\boldsymbol{Y}})_{\boldsymbol{Pc}(\boldsymbol{s})_{i:\boldsymbol{Y}}}) \end{aligned} \tag{3.14}$$

$$\begin{aligned} \sigma^2_{Y_i | \boldsymbol{Pc}(\boldsymbol{s})_{i:\boldsymbol{Y}}}(\boldsymbol{pd}(\boldsymbol{s})_{i:\boldsymbol{Y}}) = & \Sigma_{Y_i | \boldsymbol{Pc}(\boldsymbol{s})_{i:\boldsymbol{Y}}}(\boldsymbol{pd}(\boldsymbol{s})_{i:\boldsymbol{Y}}) \\ = & \Sigma(\boldsymbol{pd}(\boldsymbol{s})_{i:\boldsymbol{Y}})_{Y_i} \\ & - \Sigma(\boldsymbol{pd}(\boldsymbol{s})_{i:\boldsymbol{Y}})_{Y_i, \boldsymbol{Pc}(\boldsymbol{s})_{i:\boldsymbol{Y}}} \\ & \cdot \Sigma(\boldsymbol{pd}(\boldsymbol{s})_{i:\boldsymbol{Y}})^{-1}_{\boldsymbol{Pc}(\boldsymbol{s})_{i:\boldsymbol{Y}}} \\ & \cdot \Sigma(\boldsymbol{pd}(\boldsymbol{s})_{i:\boldsymbol{Y}})_{\boldsymbol{Pc}(\boldsymbol{s})_{i:\boldsymbol{Y}}, Y_i} \end{aligned} \tag{3.15}$$

We are aware that alternative definitions exist for CGN paradigm. CGNs essentially belong to a class of PGM for mixed random variables that is introduced for the first time by Lauritzen and Wermuth (1989), and further developed by Lauritzen (1992); Geiger and Heckerman (1994); Lauritzen (1996); Cowell et al. (1999); Lauritzen and Jensen (2001). The definition proposed by Geiger and Heckerman (1994) imposes additional structural constraints since they are of great help for deriving a factorable closed score for the marginal likelihood of data given a CGN structure [Geiger and Heckerman (1994); Peña et al. (2002)]. However the definition for CGN provided in this dissertation is proposed in [Lauritzen and Jensen (2001); Bottcher (2004)]. In [Bottcher (2004)] the author generalizes the work of Geiger and Heckerman (1994) by relaxing the structural constraints of previous definitions and deriving a factorable closed form for the marginal likelihood of mixed data given a model structure. This definition allows the use of priors in a Bayesian way. The local parameter conjugation can be specified as a Dirichlet distribution for the discrete part, and for the mixed part of the network as a Gaussian-inverse Gamma distribution for each configuration of the discrete parents [Bottcher (2004)].

3.5 Classifiers based on CGN

Since this dissertation was initially focused on continuous domains, this section and the experimental results presented in Section 3.6 are focused on continuous domains, that is, domains with only continuous predictor variables and the class being the single discrete variable. In other words we are considering domains defined over the random variables $(\boldsymbol{Y}, C)$ where the class variable C is discrete and takes r distinct values and $\boldsymbol{Y} = (Y_1, ..., Y_n)$ is an n-dimensional continuous random variable. Note that in a CGN for $(\boldsymbol{Y}, C)$ with an augmented naive Bayes structure, the parameters of the local factor $f(y_i|pc_i, c)$ can be obtained from $\boldsymbol{mu}_{\boldsymbol{Y}}(c)$ and $\Sigma_{\boldsymbol{Y}}(c)$ using Propositions 3.3 and 3.5, for $i = 1, ..., n$.

This section is based on the results presented in [Pérez et al. (2006a)]. Although we are considering only continuous domains, they are useful to contrast the behavior of classifiers based on the BMN plus discretization approach and CGN approach because the part of the model concerning the discrete part is equivalent in both BMN and CGN paradigms. See Section 2.6 for further details of BMN plus multinomial approach.

This section introduces a set of wrapper and filter classifier induction algorithms based on CGN for $(\boldsymbol{Y}, C)$ which learn classifiers with augmented-naïve Bayes structures, such us, NB, TAN, kAN and JAN structures. Most of the introduced classifier induction algorithms are adaptations to the CGN paradigms of the algorithms presented in Section 2.5 for the BMN paradigm. It must be highlighted that the parameters are learned using the maximum likelihood estimator.

For the sake of readability, next, we present the particularization of the equations provided in Sections 3.3 and 3.4 for continuous domains $(\boldsymbol{Y}, C)$ where the structure $\boldsymbol{s}$ is considered implicitly.

The particularization of the factorization presented in Equation 3.10 to continuous domains is as follows

$$\rho(\boldsymbol{y}, c) = p(c)f(\boldsymbol{y}|c) = p(c)\prod_{i=1}^{n} f(y_i|\boldsymbol{pa}_i) = p(c)\prod_{i=1}^{n} f(y_i|\boldsymbol{pc}_i, c) \tag{3.16}$$

where by definition of CGN $\rho(\boldsymbol{y}, c) \rightsquigarrow \mathcal{MG}(\boldsymbol{y}, c; p(c), \boldsymbol{\mu}(c), \Sigma(c))$. Moreover each factor $f(y_i|\boldsymbol{pc}_i, c)$ follows a Gaussian distribution, $f(y_i|\boldsymbol{pc}_i, c) \sim \mathcal{N}(y_i; \mu_{i|c}, \sigma^2_{i|c})$ where

$$\mu_{i|c} = \boldsymbol{\mu}(c)_{Y_i} + \Sigma(c)_{Y_i,\boldsymbol{Pc}_i}\Sigma(c)^{-1}_{\boldsymbol{Pc}_i}(\boldsymbol{pc}_i - \boldsymbol{\mu}(c)_{\boldsymbol{Pc}_i}) \tag{3.17}$$

$$\sigma^2_{i|c}(\boldsymbol{z}) = \Sigma(c)_{Y_i} - \Sigma(c)_{Y_i,\boldsymbol{Pc}_i}\Sigma(c)^{-1}_{\boldsymbol{Pc}_i}\Sigma(c)_{\boldsymbol{Pc}_i,Y_i} \tag{3.18}$$

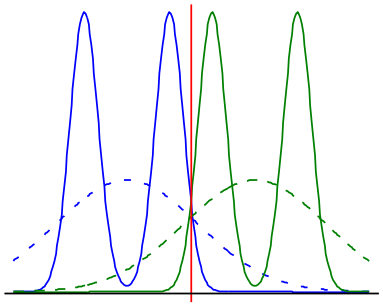
One of the main advantages of CGNs with respect to BMNs is related to the number of parameters needed to model the continuous part of the domain. This advantage is highlighted in continuous domains such as those defined over $(\boldsymbol{Y}, C)$. In contrast to the exponential number of parameters necessary to learn

a complete graph in BMNs ($\mathcal{O}(r\prod_{i=1}^{n} r_i)$, where r_i and r are the cardinality of the variables Y_i and C respectively), the number of parameters necessary to model a complete graph based on CGNs with continuous variables has a low polynomial rate [Geiger and Heckerman (1994)], $\mathcal{O}(n^2 r)$. Due to the fewer number of parameters, the CGN-based classifiers tend to have less sensitivity to the changes in the training set. They also adjust the training data sets less than BMN-based classifiers. Therefore, in general, CGN-based classifiers tend to be more stable. In other words, they tend to have a lower variance term of the error [Friedman (1997); Kohavi and Wolpert (1996)]. Besides, a lower number of parameters allows a more reliable and robust computation of the necessary statistics, which is closely related with the stability of the model.

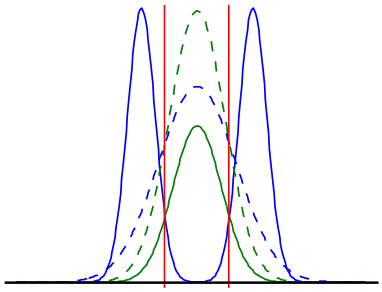
Moreover, the parameters $p(c)$, $\boldsymbol{\mu}_{\boldsymbol{Y}}(c)$ and $\Sigma_{\boldsymbol{Y}}(c)$ can be computed *a priori*, without taking into account the structure to be considered. In other words, complete TAN, kAN and JAN structures in CGN paradigm for continuous domains $(\boldsymbol{Y}, C)$ require the same parameters, $\boldsymbol{\mu}_{\boldsymbol{Y}}(c)$ and $\Sigma_{\boldsymbol{Y}}(c)$. Due to this property, the classifiers induction algorithms based on TAN, kAN and JAN structures tends to have the same variance term associated to the bias plus variance decomposition of the error (see Figure 3.3). In other words, the stability of these algorithms to changes in the training set can be considered similar. Besides, the possibility of computing the parameters *a priori* allows a more efficient backward structure search compared to BMN-based algorithms.

BMNs only handle discrete variables. The continuous variables must be discretized in order to handle them. There is a loss of information in this discretization process [Yang and Webb (2003)]. The same classifier induction algorithm obtains different classifiers and different classification scores depending on the criteria used to discretize the data. It could be concluded that the lost information depends on the discretization criteria used. On the other hand, CGNs are only able to handle continuous variables assuming that they follow a Gaussian distribution. Therefore, information is used erroneously if the real distribution of the variables differs much from the Gaussian distribution and, thus, the estimation of $\rho(\boldsymbol{y}, c)$ tends to have higher bias term in the estimated error decomposition.

However, in supervised classification Gaussian approach is suitable for many non-Gaussian domain. As we noted in Chapter 1, the ultimate goal consists of obtaining good estimators of the classification boundaries. In the case of generative and conditional classifiers, the estimation of the distributions $\rho(\boldsymbol{y}, c)$ and $p(c|\boldsymbol{y})$, respectively, can be considered an intermediate step which determines the classification boundaries. Moreover, when the parameters are learned discriminatively, focusing on obtaining good classification boundaries rather than good estimators of the distributions $\rho(\boldsymbol{y}, c)$ and $p(c|\boldsymbol{x})$, the Gaussian approach is even more suitable for many non-Gaussian densities. However, the classifiers presented in this section are based on maximum likelihood, which is considered a generative learning of parameters. Figure 3.1 shows two examples of a non Gaussian domain for which the Gaussian-approach is suitable with both generative and discriminative learning of parameters. This



(a) The parameters of the estimated Gaussian densities can be learned with a generative approach, such as maximum likelihood.



(b) In this example, the parameters should be learned using a discriminative approach.

Fig. 3.1. These figures illustrates how to model the classification boundaries obtained with the true underlaying densities using Gaussian densities. The true densities are represented by solid lines, the approached Gaussian densities are represented by dotted lines and the classification boundaries by solid vertical red lines.

fact is corroborated by the experimentation summarized in Section 3.6: even most of the variables implied in the selected domains are non-Gaussian the estimated errors for classifiers based on CGN and those based on BMN plus discretization approach obtains comparable results. In addition, the assumption of normally distributed data can avoid the overfitting problem when the structure of the graph is too complex, and also tends to obtain an estimation of the joint distribution with less *variance* due to the low polynomial number of parameters.

3.5.1 naïve Bayes

This subsection presents three algorithms adapted from BMN paradigm for the induction of classifiers based on CGN paradigm with naïve Bayes structure (see Section 2.5.3.1). The adaptation of MNB, fsMNB and wsMNB are called *Gaussian naïve Bayes* (*GNB*), *ranking selective Gaussian naïve Bayes* (*rsGNB*) and *wrapper selective Gaussian naïve Bayes* (*wsGNB*), respectively. The pseudo-code of wsGNB is similar to Figure 2 considering CGN instead of the BMN paradigm.

We consider the rsGNB algorithm an adaptation of fsMNB because it computes the mutual information with the class $I(Y_i; C)$ and using a threshold τ. It decides to include the arc (Y_i, C) iff $I(Y_i; C) > \tau$. However, the threshold is fixed using a wrapper approach and therefore we consider rsGNB a mixed approach. rsGNB ranks the predictor variables $(Y_1, ..., Y_n)$ in descendant order of their mutual information $I(Y_i; C)$. We assume, without loss of generality, that the variables are ordered according to this criteria, that is, $I(Y_i; C) \geq I(Y_j; C)$

for any $i < j$. The quantity $I(Y_i; C)$ is computed using the particularization of Proposition 3.13 to the case $n = |C| = 1$. The estimation of the covariance matrixes $\Sigma_{\boldsymbol{Y}}(c)$ and the probabilities $p(c) = Pr(C = c)$ have been obtained using the maximum likelihood estimator. This classifier induction algorithm construct n classifiers, $\{NB_1, ..., NB_n\}$, with NB structures for $(Y_1, ..., Y_i, C)$ for $i = 1, ..., n$. Then, it estimates their performance and the best constructed classifier is selected as the final model returns the classifier with the lower estimated error. Note that this procedure can be understood as setting a threshold $\tau = I(Y_i; C)$ where NB_i is the selected classifier. The rsGNB algorithm compared to wsGNB has less time complexity: in the worst case, only $\mathcal{O}(n)$ classifiers are constructed compared with $\mathcal{O}(n^2)$ of the wrapper approach. However, in practice, fRankingNB could has problems with redundant variables. It ranks variables in terms of $I(Y_i; C)$ without considering the redundant information that the variable shares with the previously included variables, Y_j with $j < i$. Therefore, any variable Y_i with redundant information (with the variables already included in the model) and a great $I(Y_i; C)$ value, could be added in the first steps of the forward greedy structural search process and in practice it could hurt the accuracy of the obtained NB model Langley and Sage (1994).

For further details on classifiers based on naïve Bayes structures for the BMN paradigm see Subsection 2.5.3.1.

3.5.2 Tree augmented naïve Bayes

This subsection presents the adaptation of two algorithms for the induction of classifiers based on CGN paradigm with tree augmented naïve Bayes structure. The adaptation of fMTAN and wsMTAN are called *Gaussian tree augmented naïve Bayes* (*fGTAN*) and *wrapper selective Gaussian tree augmented naïve Bayes* (*wsMTAN*), respectively. See Section 2.5.3.2 for further details on fMTAN and wsMTAN.

As in the original fMTAN [Friedman et al. (1997)], *fGTAN* finds the tree structure that maximizes the likelihood given the data. Hence, fGTAN is considered a pure filter algorithm. In order to adapt this algorithm to continuous domains defined over $(\boldsymbol{Y}, C)$, we need to calculate the conditional mutual information between every pair of continuous predictor variables given the class variable, $I(Y_i; Y_j|C)$ for all $1 \leq i < j \leq n$. This quantity can be approached using the particularization of Proposition 3.12 replacing U, $\bar{U}$ and Z by Y_i, Y_j and C, respectively. The algorithm has the same computational complexity as $fMTAN$.

The classifier induction algorithm wsGTAN is similar to wsMTAN [Keogh and Pazzani (1999)] and, it has the same computational complexity, $\mathcal{O}(n^2)$. The pseudocode is similar to Algorithm 6 replacing BMN by CGN paradigm.

The algorithm wsGTAN can induce classifiers with incomplete TAN structures and it performs an implicit wrapper feature selection. Assuming that the discretized variables have r states the number of parameters required

by complete TAN structures in BMN is $\mathcal{O}(nr^3)$ and, on the other hand, in CGN paradigm is $\mathcal{O}(n^2r)$. Thus, the number of parameters required to model a complete TAN structure is lower in CGN paradigm than in BMN when $n < r^2$.

3.5.3 *k*-dependent augmented naïve Bayes

This subsection presents two classifier induction algorithms based on CGN paradigm with k-dependent augmented naïve Bayes structure, which have been adapted from the BMN paradigm. The kDB structures can be regarded as a spectrum of allowable dependence in a given probabilistic graphical model with the NB structure at the most restrictive extreme and the full BMN at the most general one. The reader could consult Subsection 2.5.2 for further details on kAN structures. The adaptation of fMkAN [Sahami (1996)] and wsMkAN are called *filter Gaussian k-dependent augmented naïve Bayes* (*fGkAN*) and *wrapper selective Gaussian k-dependent augmented naïve Bayes* (*wsGkTAN*), respectively. See Section 2.5.3.3 for further details on fMkAN and wsMkAN.

It should be highlighted that the number of parameters required for complete kAN structures increases exponentially with k for BMN paradigm and it is constant for CGN. Assuming that the discretized variables have r states, the number of required parameters in BMN is $\mathcal{O}(nr^{k+1})$ and in CGN paradigm is $\mathcal{O}(rn^2)$. The number of required parameters in CGN is lower when $n < r^k$.

The algorithms fGkAN and wsGkAN are similar to fMkAN and wsMkAN, respectively, but based on CGN paradigm. The pseudocode of fGkAN and wsGkAN could be given by Figures 7 and 8 replacing BMN by CGN paradigm. fGkAN is based on the mutual information $I(Y_i; C)$ and conditional mutual information $I(Y_i; Y_j|C)$ given by Equations 3.8 and 3.7 where the parameters $\sigma^2_{Y_i}(c)$, $\sigma^2_{Y_j}(c)$, $\Sigma_{(Y_i,Y_j)}(c)$ and $\sigma^2_{Y_i}$ are estimated by maximum likelihood. The algorithm wsMkAN can obtain classifiers with an incomplete kAN structure and it performs an implicit wrapper feature selection.

3.5.4 Joint augmented naïve Bayes

Joint augmented structures [Kononenko (1991); Pazzani (1997)] break the strong independence assumption of NB structures introducing the joint nodes. A *joint node* $\boldsymbol{U} = (Y_{1:\boldsymbol{U}}, ..., Y_{l:\boldsymbol{U}})$ represents a multidimensional random variable and it can be interpreted as a complete graph defined over its *components*, $(Y_{1:\boldsymbol{U}}, ..., Y_{l:\boldsymbol{U}})$. The joint nodes consist of subsets of the original variables, where the joint nodes are disjoint among them, $\boldsymbol{Y} = (Y_1, ..., Y_n) = (\boldsymbol{U}_1, ..., \boldsymbol{U}_k)$ where $\boldsymbol{U}_i \cap \boldsymbol{U}_j = \emptyset$

If a joint variable consists of discrete random variables, the states of the joint variable consist of the Cartesian product of the states of the multinomial random variables [Pazzani (1997)]. The main problem of joint variables consisting of multinomial variables Z_i is the estimation of their class conditional probability tables. A joint node $\boldsymbol{V} = (Z_{1:\boldsymbol{V}}, ..., Z_{l:\boldsymbol{V}})$ has a number of

exponential states in l, the same number of parameters as a complete graph in $(Z_{1:\boldsymbol{V}}, ..., Z_{l:\boldsymbol{V}})$. The estimation of a high number of parameters often can be unreliable and it could tend to obtain unstable parameters, which can increase the error of the associated classifier due to an increase in the variance term of the decomposition of the error. For further details on JAN structures the reader could consult Section 2.5.2.

On the other hand, considering the CGN paradigm, if a joint variable $\boldsymbol{U} = (X_{1:\boldsymbol{U}}, ..., X_{l:\boldsymbol{U}})$ consists of a set of continuous random variables, $\boldsymbol{U} = (X_{1:\boldsymbol{U}}, ..., X_{l:\boldsymbol{U}})$, we propose that it follows a *multidimensional normal distribution* Anderson (1958) conditioned to its parent, C. The conditional joint probability density function is given by:

$$f(\boldsymbol{u}|c) = (2\pi)^{-\frac{l}{2}} \mid \Sigma_{\boldsymbol{U}}(c) \mid^{-\frac{1}{2}} e^{-\frac{1}{2}(\boldsymbol{u}-\boldsymbol{\mu}_{\boldsymbol{U}}(c))^t \Sigma_{\boldsymbol{U}}(c)^{-1}(\boldsymbol{u}-\boldsymbol{\mu}_{\boldsymbol{U}}(c))} \tag{3.19}$$

where $\Sigma_{\boldsymbol{U}}(c)$ and $\boldsymbol{\mu}_{\boldsymbol{U}}(c)$ are the covariance matrix and the mean vector of $\boldsymbol{U}$ conditioned to the class value $C = c$, respectively. In order to model this density function, $\mathcal{O}(l^2)$ parameters are required. This fact avoids the problem of the probability table size needed to model the joint variable relation with the class variable when the component variables are discrete. Therefore, it is not mandatory to establish any limitation to the maximum number of predictor variables at each joint node.

This subsection presents the adaptation of the induction of classifiers based on CGN paradigm with joint augmented naïve Bayes structures. In addition, we present a novel algorithm called wrapper condensed Gaussian joint augmented naïve Bayes (*wcGJAN*). The adaptation of wfMJAN [Pazzani (1997)] and wbMJAN [Pazzani (1997)] are called *wrapper forward Gaussian joint augmented naïve Bayes* (*wfGJAN*) and *wrapper backward Gaussian joint augmented naïve Bayes* (*wbGJAN*), respectively. Both wfGJAN and wbGJAN are equivalent to wfMJAN and wbMJAN but considering the CGN paradigm instead of BMN. See Section 2.5.3.4 for further details on wfMJAN and wbMJAN.

As we say above, usually, it is not mandatory to limit the number of components of a joint node in CGNs because the number of parameters is quadratic with the number of components. The structure of a BN in $(\boldsymbol{X}, C)$ can be interpreted in terms of conditional independencies among the implied random variables. The conditional independencies provide a factorization of the joint generalized probability function $\rho(\boldsymbol{x}, c)$ with less parameters than the general case. Thus a complete graph can be seen as an exact factorization of $\rho(\boldsymbol{x}, c)$ whether an incomplete graph can be understood as an approximation to $\rho(\boldsymbol{x}, c)$ under some conditional independence assumptions.

This section presents the *wrapper condensed Gaussian joint augmented naïve Bayes* (*wcGJAN*) which tries to model $\rho(\boldsymbol{x}, c)$ considering the minimum number of conditional independencies, removing at the same time the variables which increase the classification error. wcGJAN is a wrapper greedy backward algorithm which, at each step, uses a selection of the predictor variables as a

multidimensional joint variable. It starts with all variables but, at each step of the algorithm, one of the selected variables is excluded. The pseudocode of wcGJAN is shown in Algorithm 10. In the worst case, the time complexity of the algorithm is the same as in wsGNB ($\mathcal{O}(n^2)$). When designing the wcGJAN algorithm, we have taken into account that the use of a *backward* structure search process costs the same as a *forward* process because the parameters needed can be computed *a priori*. We consider that the backward search is preferable to forward search because it allows to detect dependencies of higher order.

Algorithm 10: ***wcGJAN* algorithm**

1 Initialize structure $\mathcal{S}$ to JAN structure with a unique joint node which contains all the original predictor variables.
2 **do** {
3 Estimate the performance of each possible classifier based on CGN paradigm, considering all the structures with a unique joint node equal to the joint node of $\mathcal{S}$ but removing one of the included variables.
4 Select as $\mathcal{S}$ the best option between $\mathcal{S}$ and the evaluated classifiers.
5 } **until** No option improves the inducted classifier.

3.6 Experimental results

This section presents an empirical study performed with CGN-based classifier induction algorithms. This study includes a set of benchmark classifiers taken from different families of classifiers. The empirical results are divided in two parts. The first part includes the estimation of the classification error of the benchmarks and the CGN-based classifiers. Using on the estimated errors the proposed classifier induction algorithms are compared using the Friedman plus Shaffer's static procedure as is suggested by García and Herrera (2008). The second part consists of analyzing the sources of the error associated to CGN-based classifier induction algorithms using the bias plus variance decomposition of the error proposed by Kohavi and Wolpert (1996).

The results have been obtained in nine *UCI repository* data sets Asuncion and Newman (2007), which only contain continuous predictor variables. In order to interpret the results, it should be take into account that most parts of the UCI repository data sets are already preprocessed Kohavi (1995b): in the data sets included, there are few irrelevant or redundant variables, and little noise as suggested by van der Putten and van Someren (2004). Thus, it is more difficult to obtain statistically significant differences between the results of the algorithms in this type of data sets [van der Putten and van Someren (2004)]. The main characteristics of the data sets included are summarized

in Table 3.1. It must be noted that none of the included data sets, except *WAVEFORM* and a subset of variables of *WINE*, clearly obey the assumption that class-conditioned variables follow a conditional Gaussian distribution.

♯	Data Set	num. class values	num. variables	num. instances
1	BALANCE	3	4	625
2	BLOCK	5	10	5474
3	HABERMAN	2	3	307
4	IRIS	3	4	150
5	LIVER	2	6	345
6	PIMA	2	8	768
7	VEHICLE	4	19	846
8	WAVEFORM	3	21	5000
9	WINE	3	13	179

Table 3.1. Basic characteristics of the data sets: the number of different values of the class variable, the number of predictor variables, and the number of instances.

Table 4.3 shows the errors obtained with the following set of well known state-of-the-art classifier induction algorithms: k-NN [Cover and Hart (1967)] with different k, MNB [Duda and Hart (1973)] and fMTAN [Friedman et al. (1997)], ID3 [Quinlan (1986)] and C4.5 [Quinlan (1993)], linear discriminant analysis (LDA) [Fisher (1936)] and Multilayer Perceptron (MP) [Rosenblatt (1959)] (all of them implemented in Weka 3.4.3 package [Witten and Frank (2005)]). The estimated predictive errors summarized in Table 4.3 have been obtained, for each classifier at each data set, by a stratified 10-fold cross-validation process. In order to learn the discrete classifiers presented in Table 4.3 (MNB, fMTAN and ID3), data sets have been discretized following the supervised discretization presented by Fayyad and Irani (1993).

	k-NN		Bayesian		Trees			
Data Set	1-NN	3-NN	NB	TAN	ID3	C4.5	MP	LDA
1	15.2±3.5	15.2±3.5	29.3±4.1	28.6±3.7	30.4±3.8	23.4±3.8	19.3±3.8	12.3±6.1
2	4.0±0.6	4.1±0.6	6.4±0.6	3.9±0.9	4.5±0.7	3.1±0.4	3.9±1.5	10.0±0.6
3	32.4±7.0	29.7±4.9	27.1±3.2	27.1±3.2	27.1±3.2	28.1±4.1	27.1±6.1	26.5±6.3
4	4.7±5.5	4.7±5.5	6.0±5.8	5.3±5.3	6.0±6.6	4.0±5.6	2.3±3.4	1.3±2.7
5	27.1±6.3	28.3±5.9	26.8±10.5	26.7±10.5	26.8±10.5	21.3±8.7	28.4±7.4	30.7±6.1
6	29.8±4.7	27.3±5.1	22.1±3.5	21.1±3.8	23.1±3.8	26.2±5.7	24.9±5.5	23.1±4.2
7	20.1±4.5	28.5±5.3	37.4±4.2	25.8±4.8	29.6±4.4	27.4±6.0	17.5±3.1	20.2±4.2
8	23.1±2.0	18.7±1.9	18.2±1.5	16.8±1.5	22.5±1.2	14.0±1.4	15.5±0.9	13.7±1.3
9	5.1±4.1	5.1±4.1	1.1±2.3	1.7±2.7	3.4±2.9	6.2±5.5	2.8±4.0	0.0±0.0
Average	17.9	17.9	19.4	17.4	19.3	17.1	15.7	15.3

Table 3.2. The estimated predictive error averages obtained with a set of well known state-of-the-art algorithms. The best results, in each data set, are marked in grey.

The parameters for the fGkAN, wGkAN, wfGJAN and wbGJAN algorithms are the following:

1. fGkAN with $k = 1$. We have checked that fGkDB obtains the best scores at $k = 1$.

2. wGkAN with $k = n - 1$. Bear in mind that the number of parameters to model a complete graph is only ($\mathcal{O}(n^2)$). With $k = n - 1$, there are no limitations for the wGkAN algorithm. It is not mandatory to limit the structural complexity with the wGkAN algorithm. With $k = n - 1$, there are no limitations for the wGkAN algorithm: We allow each predictor variable to have the maximum number of parents, $n - 1$.
3. wfGJAN and wbGJAN with $r = n$, where r is the maximum number of predictor variables allowed at each joint node. With $r = n$, there are no limitations for the wfGJAN and wbGJAN algorithms: We allow joint nodes with n predictor variables (the maximum) to be constructed.

The experimental results are divided into two subsections. In Section 3.6.1, the estimated classification errors of the algorithms are summarized. Besides, based on these results the algorithms are compared using Friedman plus Shaffer's static procedure as García and Herrera (2008) suggest. Then, following the experimental setup proposed by Kohavi and Wolpert (1996), the bias-variance decomposition of the obtained estimated errors is performed in Section 3.6.2 in order to study the sources of the error of the presented CGN-based classifiers.

3.6.1 Estimated classification error

The results, for each classifier based on CGN paradigm in each data set, have been obtained by a 10-fold cross-validation process in order to estimate the predictive accuracies. The estimated classification error, for each classifier in each data set, is summarized in Table 3.3. Due to the low number of the included data sets in the experimentation, the comparisons between classifier induction algorithms based on the estimated error has been divided in groups of interest. Otherwise, the number of all pairwise comparisons is too high and the corrections imposed to the p-values by the Shaffer's static procedure are too strong to obtain statistical significative results.

Table 3.3 also summarizes two different analyses of the estimated accuracies obtained. The first analysis calculates for each classifier, the average estimated classification error across all data sets. The *Average* row contains the results of the analysis. For example, wGkAN has obtained an average estimated error of 16.2 across all domains (see Table 3.3).

The second analysis is a hypothesis test in order to study whether the best classifier induction algorithm, at each data set, has obtained statistically significant better score values with respect to the rest of the algorithms. For each data set, the algorithm with the best average score is marked as the best: In case of a tie, the algorithm with the smallest standard deviation is marked. Then, based on the estimated predictive accuracies (obtained with each fold of the 10-fold cross-validation process), we establish whether the previously selected algorithm has obtained statistically significantly better results with respect to the rest of algorithms using a non-paired Mann-Whitney

Structures	NB			TAN		kAN		JAN		
# Data Set	GNB	rsGNB	wsGNB	fGTAN	wGTAN	fGkAN	wkAN	wfGJAN	wbGJAN	wCGJAN
1 BALANCE	9.1± 4.2	9.1± 2.9	9.1± 3.0	10.9± 4.5	9.1± 2.2	• 11.8± 3.7	9.1± 2.6	8.3± 4.6	9.1± 3.1	8.3± 3.1
2 BLOCK	• 9.6± 0.8	• 7.3± 0.7	• 5.7± 1.0	• 7.5± 1.4	• 5.2± 0.4	• 7.3± 0.7	4.4± 0.7	4.6± 1.0	5.0± 1.0	5.8± 0.6
3 HABERMAN	25.4± 8.1	25.2± 8.6	25.3± 12.4	24.2± 6.0	24.6± 7.0	24.2± 9.7	24.5± 8.5	24.2± 7.2	24.2± 3.1	23.8± 9.9
4 IRIS	4.0± 5.3	4.0± 3.3	4.0± 3.3	3.7± 3.3	2.0± 3.1	3.7± 4.4	2.0± 3.1	2.0± 3.1	2.0± 4.3	2.0± 3.1
5 LIVER	44.1± 5.6	43.2± 8.4	42.0± 6.0	38.8± 7.3	42.1± 9.1	48.3± 6.4	41.4± 6.0	42.0± 5.7	48.3± 5.7	42.2± 7.4
6 PIMA	23.8± 4.6	23.3± 4.3	23.4± 7.3	23.4± 4.6	23.3± 6.4	24.2± 6.4	22.4± 3.6	23.3± 4.4	22.7± 4.4	23.8± 4.4
7 VEHICLE	• 52.7± 6.7	• 50.9± 5.0	• 43.3± 3.8	• 21.6± 3.4	• 24.9± 4.7	• 21.4± 2.2	• 25.9± 2.8	11.0± 4.0	9.3± 2.6	9.3± 2.6
8 WAVEFORM	• 19.1± 2.4	• 18.2± 2.6	• 17.5± 1.4	• 17.2± 1.8	• 15.1± 0.9	• 17.3± 1.5	• 15.6± 1.0	12.7± 1.4	12.5± 1.8	12.5± 1.8
9 WINE	1.1± 2.3	○ 2.2± 3.7	0.6± 1.7	0.0± 0.0	1.1± 2.3	0.6± 1.7	0.6± 1.7	0.6± 1.7	1.1± 2.3	0.6± 4.4
Average	21.0	20.4	19.0	16.4	16.4	17.6	16.2	14.3	14.9	14.3

Best estimated error in each data set	$\overline{x}\pm\ s$
$\alpha = 5\%$ significance level in a non-paired Mann-Whitney test	•
$\alpha = 10\%$ significance level in a non-paired Mann-Whitney test	○

Table 3.3. Summary of the estimated classification error. The first row of the table contains the type of structures, and the second row, the classifier induction algorithms associated with each structure.

test [Dudewicz and Mishra (1988)]. The study has been performed at $\alpha = 10\%$ and $\alpha = 5\%$ significance levels, represented in Table 3.3 by "$\circ$" and "$\bullet$" symbols, respectively. For example, in the *BLOCK* data set, fTAN has obtained an error significantly worse at $\alpha = 5\%$ than wkAN, which has obtained the best score.

The comparisons between the performance of the classifiers have been performed using a procedure suggested by García and Herrera (2008). The procedure consists of two hypotheses tests. First, Friedman test is performed in order to conclude if there are statistical differences among some of the classifiers included in the comparison (see 1.8.2.2 for further details). In other words, the null hypothesis consists of considering that all the classifiers have the same average performance. Once the initial null hypothesis is rejected Shaffer's static procedure is performed in order to make all the pairwise comparisons between the implied algorithms (see 1.8.2.2 for further details).

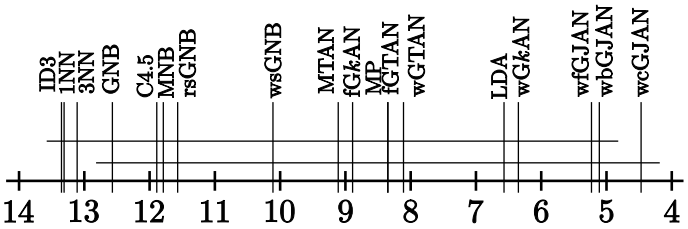
The comparisons between classifier induction algorithms using hypothesis tests has been divided in three groups:

- All the classifier induction algorithms: 1NN, 2NN, ID3, C4.5, LDA, MP, MNB, fMTAN, GNB, frGN, wGNB, fGTAN, wGTAN, fkAN, wkAN, wfGJAN, wbGJAN and wcGJAN.
- Bayesian multinomial network and conditional Gaussian network based classifiers: MNB, MTAN, GNB and fGTAN.
- Conditional Gaussian network based classifier induction algorithms: GNB, frGNB, wGNB, fGTAN, wGTAN, fGkAN, wGkAN, wfGJAN, wbGJAN and wcGJAN.

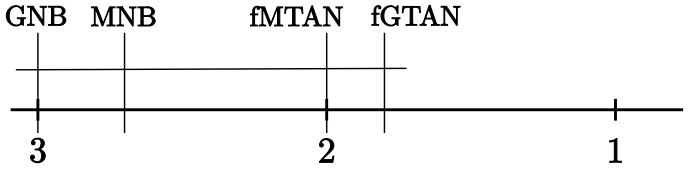
Next we analyze the results obtained with the mentioned Friedman plus Shaffer's static procedure divided by blocks. All the statistical tests have been performed using a test size $\alpha = 0.05$.

Taking all the classifiers into account, the Friedman test rejects the null hypothesis and, thus, it can be concluded that there exist some statistical significant differences between the classifier induction algorithms being considered. Then we apply the Shaffer's static procedure to make all the pairwise comparisons and the results are summarized in Figure 3.2(a). As we noted previously, in order to obtain statistically significative differences, the number of classifiers being considered in this comparison (18) is too high with respect to the number of data sets included in the experimentation (9). Two overlapped clusters of classifiers have been obtained and we can only conclude that wcGJAN is statistically better than ID3 and 1NN taking into account all the pairwise comparisons. On the other hand, this comparison provides the mean rank obtained by each classifier induction algorithm and this rank illustrates the competitive behavior of the proposed classifiers, especially those based on JAN structures (wfGJAN, wbGJAN and wcGJAN).

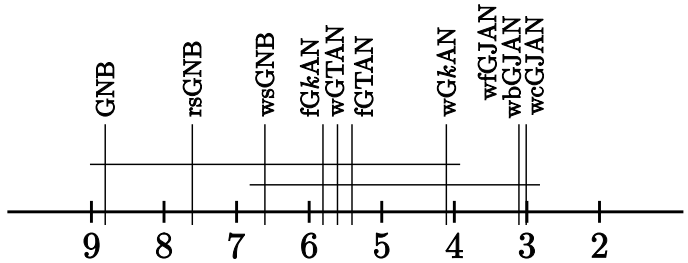
The second study consists of comparing classifier induction algorithms based on both BMN and CGN classifiers. In order to make a fair comparison the naïve Bayes and filter tree augmented naïve Bayes algorithms have been



(a) Comparisons among all the classifier induction algorithms.



(b) Comparisons among BMN and CGN based classifier induction algorithms.



(c) Comparisons among all the provided classifier induction algorithms based on CGN.

Fig. 3.2. These diagrams summarize all the pairwise comparisons performed with the Friedman plus Shaffer's static procedure [García and Herrera (2008)].

selected: MNB, GNB fMTAN and fGTAN. Friedman hypothesis test rejects the null hypothesis which states that MNB, GNB, fMTAN and fGTAN obtains the same average performances. However, Shaffer's static procedure can not detect statistically significant results, which is illustrated in Figure 3.2(b).

The third and last study, consists of comparing the algorithms based on CGN paradigm which have been presented in Section 3.5. Friedman's hypothesis test rejects the null hypothesis and, thus, it can be concluded that there exist some differences between the classifier induction algorithms being considered. The results provided with the Shaffer's static post hoc procedure are summarized in Figure 3.2(c). Due to the low number of data sets with respect to the number of algorithms being compared, there are not many statistically significant results. Shaffer's static procedure is only able to detect two overlapped clusters of algorithms. However, the provided rank illustrates the relative behavior of the presented classifiers. Clearly, NB based algorithms ob-

tain the worst average behavior and JAN structures the best results. Note that JAN structure based classifier algorithms obtains statistically significant better performances than NB based classifiers, except wsGNB. Filter approaches obtain worse average rankings than the wapper alternatives, probably due to the complete structures induced by the filter versions which implies the compulsory addition of some of the included arcs.

3.6.2 Bias-variance decomposition of CGN-based classifiers

In this section, we perform the bias-variance decomposition in order to analyze the different sources of the error of the provided CGN based classifier induction algorithms. For further details on the theory behind the bias plus variance decomposition see Section 1.8.3. The analysis of the sources of the error is useful to explain the behavior of an algorithm and also to study the particularities of a given domain.

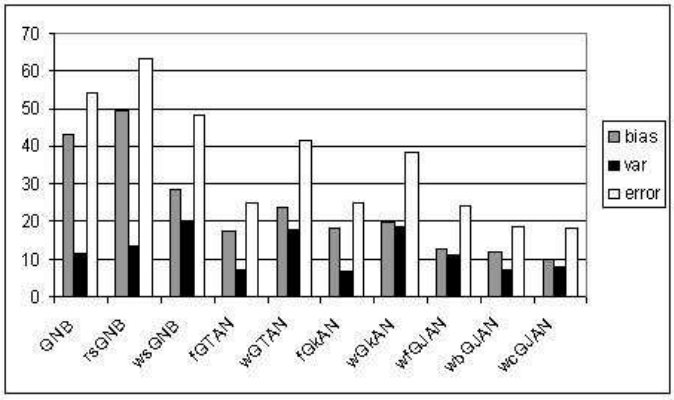
The decompositions have been performed following the suggestions of Kohavi and Wolpert (1996) with parameters $N = 20$ and $m = 1/3|BD|$, where N is the number of training sets, m is its size and $|BD|$ is the size of the data set. We have set $N = 20$ because the bias estimation is precise enough for this value (see Figure 1 of Kohavi and Wolpert (1996)), and $m = 1/3|BD|$ to ensure a minimum training set size which could avoid overfitting problems. Kohavi and Wolpert (1996) choose a set of databases with at least 500 instances in order to ensure accurate estimates of the error. In order to interpret the results, we must take into account that only the *BALANCE*, *BLOCK*, *PIMA*, *VEHICLE* and *WAVEFORM* data sets fulfill this condition (see Table 3.1). Thus, the conclusions obtained with the data sets mentioned are the most important ones.

Table 3.4 shows the results of the decomposition obtained for each classifier in each data set. It also includes an additional row which contains the averages for each classifier across all data sets. For example fGTAN obtains a $bias^2 = 7.0$ and $var = 4.0$ decomposition for *BALANCE*, and an average decomposition across all the data sets of $bias^2 = 17.1$ and $var = 6.5$.

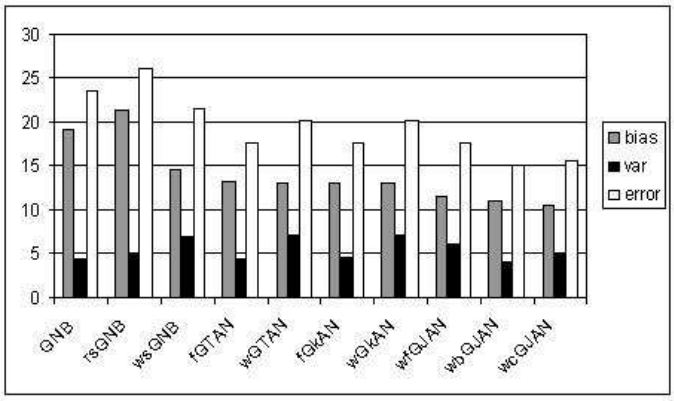
From Table 3.4, one can conclude, on average, that the bias terms of the CGN-based classifiers presented are higher than the variance term. Moreover, the variance is almost constant for different structure complexities. This can be due to the Gaussian assumption and to the low number of parameters needed to model even the most complex classifiers, which can be interpreted as low sensitivity. Besides, it can be seen that, on average (see row Average of the Table 3.4 and Figure 3.3(b)), the bias term decreases with an increase of model complexity, whereas the variance remains almost constant. Comparing the filter and wrapper approaches at each augmented naïve Bayes structure, filter algorithms seems to obtain lower variance but higher bias than wrapper approaches, probably due to the complete and incomplete structures obtained by the filter and wrapper approaches, respectively.

Structures	naïve Bayes			TAN		kDB		Semi		
# Data Set	GNB	rsGNB	wsGNB	fGTAN	wGTAN	fGkAN	wGkAN	wfGJAN	wbGJAN	wcGJAN
1 BALANCE	6.8 ± 3.7	6.0 ± 3.3	7.5 ± 4.5	7.0 ± 4.0	8.2 ± 4.1	7.9 ± 3.9	7.8 ± 3.5	8.5 ± 4.9	9.7 ± 0.6	8.0 ± 3.2
2 BLOCK	6.5 ± 2.3	6.1 ± 1.9	4.1 ± 1.1	5.8 ± 1.9	3.1 ± 1.4	5.1 ± 1.9	3.7 ± 2.0	4.2 ± 2.6	3.5 ± 1.7	3.9 ± 2.1
6 PIMA	21.9 ± 3.6	31.0 ± 1.6	18.9 ± 4.5	21.7 ± 5.5	19.0 ± 5.6	21.5 ± 5.1	22.6 ± 5.3	20.4 ± 5.5	19.6 ± 6.1	18.9 ± 7.5
7 VEHICLE	43.1 ± 11.4	49.7 ± 13.6	28.4 ± 19.9	17.4 ± 7.3	23.7 ± 17.7	18.2 ± 6.7	19.8 ± 18.5	12.9 ± 11.0	11.7 ± 7.0	10.0 ± 8.1
8 WAVEFORM	18.1 ± 0.7	13.9 ± 3.6	13.9 ± 4.7	14.6 ± 3.4	10.6 ± 7.1	13.1 ± 4.4	10.9 ± 6.8	11.6 ± 6.4	10.9 ± 4.9	11.8 ± 4.7
Average	19.3 ± 4.3	21.3 ± 4.8	14.6 ± 7.0	13.3 ± 4.4	12.9 ± 2.7	13.2 ± 4.5	13.0 ± 7.2	11.5 ± 6.1	11.1 ± 4.1	10.5 ± 5.1

Table 3.4. Bias-variance decomposition of the expected misclassification error rate.



(a) *VEHICLE* data set.



(b) Average across all data sets.

Fig. 3.3. Bias-variance decomposition examples. Due to the Gaussian assumption and to the low number of parameters, the bias term tends to dominate the variance of the decomposition of the error on average.

In order to illustrate the behavior of the classifiers taking into account the different complexities, the following behavior must be highlighted. It is illustrated in Figure 3.3(a) and Figure 3.3(b) (which correspond to the rows labeled with *VEHICLE* and *Average* of Table 3.4). Figure 3.3(a) shows that the bias term decreases if the complexity increases. This could be due to the great adjustment of the more complex models, which can approximate the target densities better. The variance term is always lower than the bias. Finally, the variance of the filter algorithms seems to be slightly lower compared to the wrapper algorithms.

Similarly, the behavior of the average across all the data sets at each algorithm, shown in Figure 3.3(b), is consistent with the behavior in Figure

3.3(a): the bias term decreases with the complexity whereas the variance remains almost constant.

3.7 Conclusions and future work

This chapter introduces the conditional Gaussian network paradigm for supervised classification in mixed domains. In order to properly present this paradigm we have provided a set of results concerning the joint, conditional and marginal mixed Gaussian distribution and Gaussian density function. Besides, we provide a set of estimators of information theory quantities under Gaussian assumption. Some of these estimators will be used by the filter classifier induction algorithms presented in this chapter.

We have proposed a battery of filter and wrapper classifier induction algorithms based on CGNs. Most of the algorithms have been adapted from BMN to the CGN paradigm: Gaussian naïve Bayes, wrapper selective Gaussian naïve Bayes, filter Gaussian tree augmented naïve Bayes, wrapper Gaussian tree augmented naïve Bayes, filter Gaussian k-dependent augmented naïve Bayes, wrapper Gaussian k-dependent augmented naïve Bayes, wrapper forward Gaussian joint augmented naïve Bayes, and wrapper backward Gaussian joint augmented naïve Bayes backward. In addition, two novel algorithms have been introduced: *filter ranking Gaussian naïve Bayes*, and *wrapper condensed Gaussian joint augmented naïve Bayes backward*.

The classifiers proposed have been compared in nine data sets by means of the estimated classification error. The family of *joint augmented naïve Bayes structure*-based algorithms obtains the best results among the algorithms proposed. Most of the algorithms based on CGN seem to be quite competitive compared to the state-of-the-art classifiers included in the experimentation. The behavior of the bias and variance terms of the decomposition of the error [Kohavi and Wolpert (1996)] shows that, if the model complexity increases, the bias term decreases and the variance remains almost constant.

The main future work lines consist of adapting the set of classifier induction algorithms for augmented naïve Bayes family of structures to deal with mixed domains. In order to design the novel filter algorithms, a set of estimators for the information theory based quantities under Gaussian assumptions will be proposed for mixed random variables.

Another future work line consists of making an empirical evaluation of alternative estimations to maximum likelihood for learning the parameters of the model. We are considering two approaches: a generative learning of parameters more stable to outliers, and a discriminative learning of parameters.

4

Supervised classification with kernel based Bayesian networks

4.1 Introduction

In previous chapters we have introduced the two most popular Bayesian networks: Bayesian multinomial networks and conditional Gaussian networks. Bayesian multinomial networks have problems handling continuous random variables and it is necessary to discretize them, with the consequent loss of information. Nevertheless, conditional Gaussian networks can directly model mixed random variables assuming that they follow a mixed Gaussian distribution. However, as we noted in Section 1.13, the implied Gaussian assumption can be a very strong constraint in some domains. This chapter introduces a third type of Bayesian network paradigm: *kernel based Bayesian network* (*KBN*). This paradigm can deal directly with mixed random variables without assuming that they follow a mixed Gaussian distribution. Alternatively, kernel based Bayesian networks are based on a non-parametric density estimation technique called *kernel density estimation*.

This chapter presents several non-parametric theoretical results for mixed domains. However, since this dissertation initially focuses on continuous domains, we present a set of classifier induction algorithms for continuous predictor variables. Consequently, the behavior of the presented algorithms is evaluated in continuous domains. It should be noted that most of the results presented in this chapter are based on the work [Pérez et al. (2009)].

The chapter is organized as follows. Section 4.2 introduces the non-parametric multidimensional kernel density estimation, focusing on Gaussian kernels. Then, Section 4.3 presents the mixed Gaussian kernel distribution which is inspired by kernel density estimation and multinomial distribution. This section, analogously to Section 3.2, introduces a set of propositions concerning the marginalization and conditioning of mixed Gaussian kernel distributions. Section 4.4 presents a general purpose non-parametric estimator for the quantities of estimation theory such us, entropy, conditional entropy, mutual information, conditional mutual information and interaction information. In addition, particular estimators of the mixed mutual information and

conditional mutual information are presented in this section. These estimators are used by the classifier induction algorithms introduced in this chapter. Section 4.5 presents a definition for KBN paradigm which is based on mixed Gaussian kernel distribution. Once the paradigm is presented, Section 4.6 presents a set of classifier induction algorithms for continuous domains based on KBN paradigm. Most of these algorithms are adaptations of the filter algorithms presented in Bayesian multinomial network for augmented naïve Bayes structures. Section 4.7 demonstrates the strong consistency of mixed Gaussian kernel distribution, of the provided estimator for the measures of information theory, of KBN given that it is a CI-map and of the flexible classifiers under the same assumption. Section 4.8 empirically studies the proposed classifier induction algorithms by means of artificial and real world data sets. The experimentation includes the error estimation, comparison and the bias plus variance decomposition of the error. Finally, in Section 4.9 we summarize the main contributions presented in this chapter and we indicate the main future work lines concerning the KBN paradigm and the mixed Gaussian kernel distribution.

4.2 Multivariate kernel density estimation

This section presents the multivariate kernel density estimator, which is the multivariate generalization of the estimator introduced in Section 1.13.3. As we explained in Section 1.13, the intuition behind kernel density estimators is that there is a probability mass around the neighborhood of each of the training points, since they have been independently sampled from the underlying true distribution. The extension from the unidimensional form consists of replacing univariate kernel K by a multivariate one, and the window width, h, by a bandwidth matrix H. Let $\boldsymbol{Y} = (Y_1, ..., Y_n)$ be an n-dimensional continuous random variable and let $\mathcal{S}^N = \{\boldsymbol{y}^{(1)}, ..., \boldsymbol{y}^{(N)}\}$ be a set of iid samples of $\boldsymbol{Y}$. From here on, we will omit the superscript N which indicates the size of the training set $\mathcal{S}^N$ for the sake of simplicity. The n-dimensional kernel estimator [Silverman (1986); Wand and Jones (1995)] with the kernel function K of the probability density function for $\boldsymbol{Y}$, $f(\boldsymbol{y})$, based on the data sample $\mathcal{S}$ and the bandwidth matrix H is given by

$$\hat{f}_K(\boldsymbol{y}; \mathcal{S}, H) = \frac{1}{N} \sum_{i=1}^{N} K_H(\boldsymbol{y} - \boldsymbol{y}^{(i)}) \tag{4.1}$$

where H is an $n \times n$ smoothing or bandwidth matrix (BM) and $K_H(\cdot)$ is the kernel function used. The kernel based density estimate $\hat{f}_K(\boldsymbol{y}; \mathcal{S}, H)$ is determined by averaging N kernel densities, $K(\cdot)$, placed at each observation $\boldsymbol{y}^{(i)} \in \mathcal{S}$. It should be noted that the probability mass placed at each observation, $\boldsymbol{y} \in \mathcal{S}$ is the same, $1/N$, as a consequence of the iid assumption. Section 4.3 present the Gaussian kernel density functions, which generalize

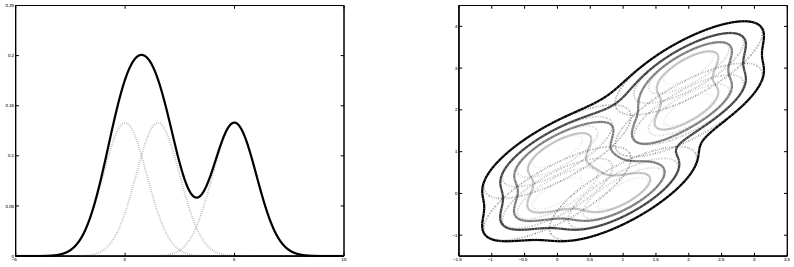
this intuition allowing different mass probabilities (mixing weights) for each observation. The kernel function $K_H(\cdot)$ used is defined as:

$$K_H(\boldsymbol{y}) = |H|^{-1/2} K(H^{-1/2}\boldsymbol{y}) \tag{4.2}$$

assuming that K is an n-dimensional density function. A kernel density estimator is characterized by

1. The kernel density K selected.
2. The bandwidth matrix H used.

H plays the role of scaling factor and is constant for all kernels. It determines the spread of the kernel at each coordinate direction. The kernel density estimate is constructed centering a scaled kernel at each observation. So, the kernel density estimator is a sum of bumps placed at the observations. The kernel function $K(\cdot)$ determines the shape of the bumps. The value of the kernel estimate at the point $\boldsymbol{y}$ is simply the average of the N kernel at that point. One can think of the kernel as spreading a "probability mass" of size $1/N$ associated with each data point in its neighborhood. Combining contributions from each data point means that in regions where there are many observations it is expected that the true density has a relatively large value. The choice of the shape of the kernel function is not particularly important. However, the choice of the value for the bandwidth matrix is very important for density estimation [Wand and Jones (1995)]. Examples for the univariate and bivariate density estimation are shown in Figures 4.1(a) and 4.1(b).



(a) Univariate kernel density estimation using three points.

(b) Bivariate kernel density estimation using five points.

Fig. 4.1. Univariate and bivariate kernel density estimators with Gaussian kernel function. Broken and solid lines represent the contribution of each kernel to the density estimation and the kernel based density estimate respectively.

In order to estimate density functions, the KBN paradigm proposed in this dissertation uses the n-dimensional Gaussian probability density function with identity covariance matrix, $\Sigma = I$, as kernel function:

$$K(\boldsymbol{y}) = (2\pi)^{-n/2} exp(-1/2\boldsymbol{y}^T\boldsymbol{y}) \rightsquigarrow \mathcal{N}(\boldsymbol{\mu} = (0, ..., 0), \Sigma = I) \tag{4.3}$$

It must be noted that, in order to define the KBN paradigm, one can use another kernel function as well, such as Epanechnikov, biweight, triweight, triangular, rectangular or uniform [Silverman (1986); Wand and Jones (1995)].

Thus, the Gaussian kernel $K_H(\boldsymbol{y}-\boldsymbol{y}^{(i)})$ is equivalent to $\mathcal{N}(\boldsymbol{y}; \boldsymbol{\mu} = \boldsymbol{y}^{(i)}, \Sigma = H)$ density function. H is a square symmetric matrix $n \times n$, and it has, in its most general form, $\frac{n(n+1)}{2}$ different parameters. This number of parameters can be very high, even for low dimensional densities. This suggests constraining H to a less general form. For example, if a diagonal matrix is used, only n different parameters must be learned. Besides, as we noted before, the impact of the BM selection is smaller in supervised classification than in density estimation and, thus, the number of parameters could be reduced without increasing the error of the obtained classifiers. Moreover, in order to control the variance of the kernel density estimators and reduce the variance term of the error of the classifier induction algorithms based on kernels [Kohavi (1995b); Pérez et al. (2009)] we suggest to reduce the number of parameters by restricting the BM to a diagonal matrix.

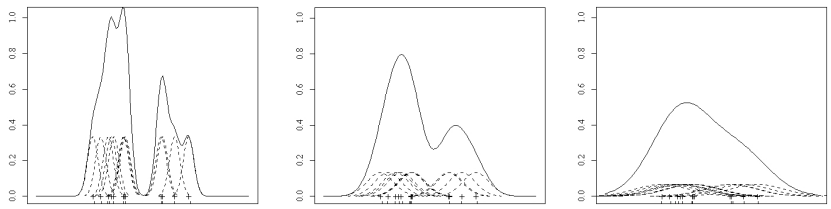
4.2.1 Learning the bandwidth matrix

As we noted in the introduction of the univariate kernel estimator (see Section 1.13.3), the selection of a proper BM $H_{\boldsymbol{Y}}$ is crucial for the density estimation, even more than the kernel function used, $K(\cdot)$ [Delaigle and Gijbels (2002)]. As we noted before, $H_{\boldsymbol{Y}}$ establishes the degree of smoothing of the density function estimation for $\boldsymbol{Y}$, $\hat{f}(\boldsymbol{y}; \mathcal{S}_{\boldsymbol{Y}}, H_{\boldsymbol{Y}})$. However, as we will see in the experimentation of Section 4.8, the impact of the selection of the BM is not so crucial for supervised classification because we are only interested in obtaining an accurate rule $\arg_c \max \rho(c, \boldsymbol{x})$ rather than obtaining a good estimation of $\rho(c, \boldsymbol{x})$, where $\boldsymbol{X} = (\boldsymbol{Y}, \boldsymbol{Z})$. We suggest sacrificing the bias of $\hat{\rho}(\boldsymbol{x}, c)$ as an estimator of $\rho(\boldsymbol{x}, c)$ in order to reduce the variance of the associated classification rule, $\arg_c \max \hat{\rho}(c, \boldsymbol{x})$.

In the univariate case, the smoothing degree depends on a unique parameter, h. Figure 4.2 shows the effect of the parameter h in the estimation of the true density using the same set of training cases. Intuitively, with h near to zero, a noisy estimation is obtained by the *undersmooth* effect (see Figure 4.2(a)). As h increases, the noise in the estimation is reduced and the univariate density begins to approximate the true density, until the optimum* is reached (see Figure 4.2(b)). As h increases, and distances itself from the optimum, the estimation starts to lose details due to the *oversmooth* effect

* The optimum is defined in terms of a loss function. Mean squared error and classification error are usually used in density estimation and classification, respectively. The optimum term is defined as the parameter value which minimizes the selected loss function.

(see Figure 4.2(c)). Finally, as h tends to ∞, the function becomes uniform and the estimator begins to resemble the *nearest neighbor* estimator.



(a) h value under the optimum. (b) Optimum h value. (c) h value over the optimum.

Fig. 4.2. The effect on the density estimation of different smoothing degrees, controlled by the parameter h.

The degree of smoothing is crucial for density estimation [Wand and Jones (1995)], e.g. for minimizing the mean squared error. On the other hand, the influence of the smoothing degree does not have such a crucial role in classification problems under 0-1 loss functions, since, as we noted before, density estimation is only an intermediate step for classifying. For example, we have shown in Chapter 3 that the Gaussian approach can obtain good estimates for classification in many non-Gaussian domains (see the examples provided in Figure 3.1) [Pérez et al. (2006b)]. Intuitively, smoothing degree controls the sensitivity to changes in the training set of the estimator and hence, of the associated classifier induction algorithm. In other words, the smoothing degree controls, somehow, the bias and variance of the estimator $\hat{\rho}(\boldsymbol{x}, c)$, and it indirectly controls the bias and variance terms of the error associated with the rule $\arg_c \max \hat{\rho}(\boldsymbol{x}, c)$. When the smoothing degree tends to zero, flexible complete graph classifier tends to the nearest neighbor classifier (maximum sensitivity). When the smoothing degree tends to infinity, it tends to the Euclidean distance classifier (minimum sensitivity) [Raudys (1991)]. The effect of the smoothing degree in supervised classification has been empirically studied in Sections 4.8.1.2 and 4.8.2.3.

Besides, in order to guarantee an efficient parametrization of a great number of densities, our goal is to use a computationally inexpensive technique to compute the BM, $H_{\boldsymbol{Y}}$. The number of parameters to be estimated for the specification of a full bandwidth matrix is $n(n+1)/2$. This number becomes unmanageable very quickly, which suggests that $H_{\boldsymbol{Y}}$ should be restricted to a simpler form to reduce the computation requirements.

In this work we use the *differential scaled* [Simonoff (1996)] approach. This approach depends on a unique smoothing parameter h:

$$H_{\boldsymbol{Y}} = diag(h_1^2, ..., h_d^2) = h^2 diag(s_1^2, ..., s_d^2) \tag{4.4}$$

where h_i is the smoothing parameter of the variable Y_i, s_i is a dispersion measure (e.g. sample standard deviation of Y_i) and $diag(\cdot)$ represents a diagonal matrix. This parametrization allows different degrees of smoothing in each coordinate direction and it performs an implicit standardization of the variables.

In the majority of practical situations, the principal problem for the selection of an optimum BM is that the real density function is unknown and, thus, it is really difficult to optimize any appropriate loss criteria (a distance function between the real and estimated densities). In this dissertation, we use the *normal rule* [Silverman (1986)]. It selects the BM that minimizes the *mean integrated square error* of the estimated density, under the assumption that the variables follow a multidimensional Gaussian density with identity covariance matrix $\boldsymbol{I}$. Due to the competitive behavior in supervised classification tasks of the Gaussian approach shown in Chapter 3, we consider that this approach is justified from the point of view of bias of the density estimator. Under this assumption we can compute the optimum h as follows [Silverman (1986)]:

$$h = (\frac{4}{(l+2)N})^{\frac{1}{l+4}} \tag{4.5}$$

where l is the dimensionality of the function to be estimated and N is the number of available instances. For example, in spite of the number of continuous variables in the domain, $(y_1, ..., y_n)$, in order to model $f(y_1, ..., y_3)$ we set $l = 3$ in Equation 4.5. However, in some situations it is desirable to use other values of l, such as the number of the random continuous variables in the domain. The computation of h using the normal rule will be explicitly explained at the estimators of the measures of the information theory presented in Section 4.4, at the end of Section 4.5 and at each classifier induction algorithm presented in Section 4.6.

As we noted before, we have chosen the normal rule Silverman (1986) plus differential scaled [Simonoff (1996)] approach because it obtains density estimations that are good enough for classifying (see Sections 4.8.1.2 and 4.8.2.3). Furthermore, we know that the normal rule tends to oversmooth the estimation [Wand and Jones (1995)] and, thus, it could obtain more stable estimations with noisy data, i.e. estimations with less variance. Besides, this approach reduces the number of parameters, and thus, it tends to obtain more stable estimators and classifier induction algorithms with less variance. At the same time, this rule avoids the increase of the high computational requirements related to the classifier induction process (see Table 4.1). It only needs to compute a fixed number of operations independently from the dimension of the density and the number of examples. There are other approaches to decide the smoothing degree for obtaining a good classification performance. The wrapper approach [Kohavi (1995b); Kohavi and John (1997)] could be one of the most interesting procedures from the performance point of view (see Sections 4.8.1.2 and 4.8.2.3). However, it involves intensive computations with flexible classifiers, especially compared with the normal rule.

4.3 Gaussian kernel probability density and mixed Gaussian kernel distribution

The definitions and propositions provided in this section are based on the following multidimensional random variables: $\boldsymbol{X} = (X_1, ..., X_{n+m}) = (\boldsymbol{Y}, \boldsymbol{Z})$ is an $(n+m)$-dimensional mixed random variable, $\boldsymbol{Y} = (Y_1, ..., Y_n)$ is an n-dimensional continuous random variable and $\boldsymbol{Z} = (Z_1, ..., Z_m)$ is an m-dimensional discrete random variable, where $\boldsymbol{Y} \cap \boldsymbol{Z} = \emptyset$. The random variables $\boldsymbol{Y}$ and $\boldsymbol{Z}$ can be partitioned as $\boldsymbol{Y} = (\boldsymbol{U}, \bar{\boldsymbol{U}})$ and $\boldsymbol{Z} = (\boldsymbol{V}, \bar{\boldsymbol{V}})$, where $\boldsymbol{U} = (U_1, ..., U_k) = (Y_{1:\boldsymbol{U}}, ..., Y_{k:\boldsymbol{U}})$ with $k > 0$ and $\boldsymbol{V} = (V_1, ..., V_l) = (Y_{1:\boldsymbol{V}}, ..., Y_{l:\boldsymbol{V}})$ with $l > 0$. These variables will be helpful to define the marginal and conditional functions.

The definitions and propositions provided in this section are based on a data set $\mathcal{S} = \{\boldsymbol{x}^{(1)}, ..., \boldsymbol{x}^{(N)}\} = \{(\boldsymbol{y}^{(1)}, \boldsymbol{z}^{(1)}), ..., (\boldsymbol{y}^{(N)}, \boldsymbol{z}^{(N)})\}$ which can be partitioned by $\boldsymbol{Z}$ into the indexed sets $\mathcal{S}_{\boldsymbol{Y}}(\boldsymbol{z}) = \{\boldsymbol{y}^{(1:\boldsymbol{z})}, ..., \boldsymbol{y}^{(N_{\boldsymbol{z}}:\boldsymbol{z})}\}$ for all $\boldsymbol{z}$, where $\boldsymbol{z}^{(i:\boldsymbol{z})} = \boldsymbol{z}$ and $\sum_{\boldsymbol{z}} N_{\boldsymbol{z}} = N$. For the sake of simplicity and clarity the following nomenclature is used to handle the indexed sets

$$\boldsymbol{y}^{(i)} = \boldsymbol{y}(\boldsymbol{Z})^i = \boldsymbol{y}^{(j)}(\boldsymbol{z}^{(i)}) = \boldsymbol{y}^{(j:\boldsymbol{z}^{(i)})} \text{ for } i = 1, ..., N \text{ and } j : \boldsymbol{z}^{(i)} = i$$
$$\boldsymbol{y}^{(i)}(\boldsymbol{z}) = \boldsymbol{y}^{(i:\boldsymbol{z})} = \boldsymbol{y}^{(j)} \text{ for } i = 1, ..., N_{\boldsymbol{z}} \text{ and } j = i : \boldsymbol{z}$$

The projections of a set $\mathcal{S}_{\boldsymbol{Y}} = \{\boldsymbol{y}^{(1)}, ..., \boldsymbol{y}^{(N)}\}$ into the random variable $\boldsymbol{U} \subset \boldsymbol{Y}$ is denoted as $(\mathcal{S}_{\boldsymbol{Y}})_{\boldsymbol{U}} = \mathcal{S}_{\boldsymbol{U}} = \{\boldsymbol{u}^{(1)}, ..., \boldsymbol{u}^{(N)}\}$ or simply as $\mathcal{S}_{\boldsymbol{U}}$, where $\boldsymbol{y}^{(i)} = (\boldsymbol{u}^{(i)}, \bar{\boldsymbol{u}}^{(i)})$ for all i.

We now present a set of results concerning the probability density function represented by the kernel density estimator, fixed the training set and the bandwidth matrixes. The results provided will be useful to analyze the mathematical properties of the estimators for the measures of information theory given in Section 4.4, the KBN network paradigm and the classifiers based on KBN presented in Section 4.6.

We call *Gaussian kernel probability density function* (*GK probability density function*) to a probability density function inspired in kernel density estimation, fixed the training set $\mathcal{S}_{\boldsymbol{Y}} = \{\boldsymbol{y}^{(1)}, ..., \boldsymbol{y}^{(N)}\}$, the bandwidth matrix $H_{\boldsymbol{Y}}$ and the mixing weights $\boldsymbol{\pi}_{\boldsymbol{Y}} = (\pi^1_{\boldsymbol{Y}}, ..., \pi^N_{\boldsymbol{Y}})$ associated to each Gaussian component, $\hat{f}(\boldsymbol{y}; \mathcal{S}_{\boldsymbol{Y}}, \boldsymbol{\pi}_{\boldsymbol{Y}}, H_{\boldsymbol{Y}})$. Note that Kernel density estimation is a particular case of GK probability density function where the mixing weight associated to the instance $\boldsymbol{y}^{(i)}$ is $\pi^i_{\boldsymbol{Y}} = 1/N$, for $i = 1, ..., N$.

GK probability density function can be viewed as a particular case of the *finite Gaussian mixture density function* [Titterington et al. (1985); McLachlan and Peel (2000)] when N is finite, for which all the compounds have the same covariance matrix $\Sigma = H_{\boldsymbol{Y}}$. In the framework of Gaussian mixture density function, the set $\mathcal{S}_{\boldsymbol{Y}}$ and the distribution $\boldsymbol{\pi}_{\boldsymbol{Y}}$ are interpreted as the set of means and mixture weights for the Gaussian components, respectively [Titterington et al. (1985)]. It should be highlighted that the results provided in this section regarding the GK probability density functions can be applied to

finite Gaussian mixture density function taking into account the constraints imposed to the covariance matrixes.

Following the intuitions behind mixed Gaussian distribution (which generalizes the Gaussian probability density function to mixed domains) we propose *mixed Gaussian kernel distribution*, which generalizes Gaussian kernel probability density function to mixed domains. This section changes the point of view of the kernel density estimation and considers the training sets and bandwidth matrixes as a set of parameters. In other words, the sets are considered as sets of mean vectors of the compounds of the associated mixture of Gaussian densities and the bandwidth matrixes as their covariance matrixes.

This section, analogously to Section 3.2, formally introduces the joint, marginal and conditional forms of the Gaussian kernel probability density and mixed Gaussian kernel distribution functions. All the results provided in this section can be directly obtained by applying the definitions and propositions introduced in Section 3.2 to each particular Gaussian kernel function taking into account that they share the same bandwidth matrix.

Definition 4.1 (Joint Gaussian kernel) $\boldsymbol{Y} = (Y_1, ..., Y_n)$ *is said to follow an* n-dimensional Gaussian kernel probability density function *based on* $\mathcal{S}_{\boldsymbol{Y}} = \{\boldsymbol{y}^{(1)}, ..., \boldsymbol{y}^{(N)}\}$, $\boldsymbol{\pi}_{\boldsymbol{Y}} = (\pi^1_{\boldsymbol{Y}}, ..., \pi^N_{\boldsymbol{Y}})$ *and* $H_{\boldsymbol{Y}}$, $f(\boldsymbol{y}) \rightsquigarrow \mathcal{GK}(\boldsymbol{y}; \mathcal{S}_{\boldsymbol{Y}}, \boldsymbol{\pi}_{\boldsymbol{Y}}, H_{\boldsymbol{Y}})$, *if its joint probability density function is given by:*

$$f(\boldsymbol{y}; \mathcal{S}_{\boldsymbol{Y}}, \boldsymbol{\pi}_{\boldsymbol{Y}}, H_{\boldsymbol{Y}}) = \sum_{i=1}^{N} \pi^i_{\boldsymbol{Y}} \mathcal{N}(\boldsymbol{y}; \boldsymbol{\mu} = \boldsymbol{y}^{(i)}, \Sigma = H_{\boldsymbol{Y}})$$

where $\mathcal{N}(\boldsymbol{y}; \boldsymbol{\mu} = \boldsymbol{y}^{(i)}, \Sigma = H_{\boldsymbol{Y}})$ *denotes the value of the Gaussian probability density function with parameters* $\boldsymbol{\mu} = \boldsymbol{y}^{(i)}$ *and* $\Sigma = H_{\boldsymbol{Y}}$ *at the point* $\boldsymbol{y}$, *and* $\pi^i_{\boldsymbol{Y}}$ *is the mixing weight of the* i*-th Gaussian component.*

Thus, a n-dimensional kernel density estimation with a fixed bandwidth matrix based on Gaussian kernel is a particular case of Gaussian kernel probability function, where $\pi^i_{\boldsymbol{Y}} = 1/N$ for $i = 1, ..., N$. Besides, when N is finite, Gaussian kernel probability density function can be understood as a finite mixture of Gaussian densities with equal covariance matrix for each Gaussian component.

Note that for $H_{\boldsymbol{Y}}$ diagonal, $H_{\boldsymbol{Y}} = diag(h_1^2, ..., h_n^2)$, the joint density is given by

$$f(\boldsymbol{y}; \mathcal{S}_{\boldsymbol{Y}}, \boldsymbol{\pi}_{\boldsymbol{Y}}, H_{\boldsymbol{Y}}) = \sum_{i=1}^{N} \pi^i_{\boldsymbol{Y}} \prod_{j=1}^{n} \mathcal{N}(\boldsymbol{y}; \boldsymbol{y}^{(i)}_j, h_j^2) \tag{4.6}$$

Proposition 4.1 (Marginal Gaussian kernel) *Let* $\boldsymbol{Y} = (Y_1, ..., Y_n)$ *follow an* n*-dimensional Gaussian kernel probability density function based on* $\mathcal{S}_{\boldsymbol{Y}} = \{\boldsymbol{y}^{(1)}, ..., \boldsymbol{y}^{(N)}\}$, $\boldsymbol{\pi}_{\boldsymbol{Y}} = (\pi^1_{\boldsymbol{Y}}, ..., \pi^N_{\boldsymbol{Y}})$ *and* $H_{\boldsymbol{Y}}$, $f(\boldsymbol{y}) \rightsquigarrow \mathcal{GK}(\boldsymbol{y}; \mathcal{S}_{\boldsymbol{Y}}, \boldsymbol{\pi}_{\boldsymbol{Y}}, H_{\boldsymbol{Y}})$, *and let* $\boldsymbol{U} = (U_1, ..., U_k) \subseteq \boldsymbol{Y}$ *and* $\bar{\boldsymbol{U}} = \boldsymbol{Y} \setminus \boldsymbol{U}$. *The marginal joint density function of* $\boldsymbol{U}$, *obtained by marginalizing* $f(\boldsymbol{y})$ *on* $\bar{\boldsymbol{U}}$, *follows an* k*-dimensional*

Gaussian kernel probability density function, $f(\boldsymbol{u}) = \int_{-\infty}^{\infty} f(\boldsymbol{y})d\bar{\boldsymbol{u}} \rightsquigarrow \mathcal{GK}(\boldsymbol{u}; \mathcal{S}_{\boldsymbol{U}}, \boldsymbol{\pi}_{\boldsymbol{U}}, H_{\boldsymbol{U}})$, *where*

$$\begin{aligned}\mathcal{S}_{\boldsymbol{U}} &= (\mathcal{S}_{\boldsymbol{Y}})_{\boldsymbol{U}} = \{\boldsymbol{u}^{(1)}, ..., \boldsymbol{u}^{(N)}\}\\ \boldsymbol{\pi}_{\boldsymbol{U}} &= \boldsymbol{\pi}_{\boldsymbol{Y}}\\ H_{\boldsymbol{U}} &= (H_{\boldsymbol{Y}})_{\boldsymbol{U}}\end{aligned}$$

Proof. The proof of this proposition is as follows:

$$\begin{aligned}f(\boldsymbol{u}) &= \int_{-\infty}^{\infty} f(\boldsymbol{u}, \bar{\boldsymbol{u}})d\bar{\boldsymbol{u}}\\ &= \int_{-\infty}^{\infty} \sum_{i=1}^{N} \pi_{\boldsymbol{Y}}^{i} \mathcal{N}(\boldsymbol{u}, \bar{\boldsymbol{u}}; \boldsymbol{y}^{(i)}, H_{\boldsymbol{Y}})d\bar{\boldsymbol{u}}\\ &= \sum_{i=1}^{N} \pi_{\boldsymbol{Y}}^{i} \int_{-\infty}^{\infty} \mathcal{N}(\boldsymbol{u}, \bar{\boldsymbol{u}}; \boldsymbol{y}^{(i)}, H_{\boldsymbol{Y}})d\bar{\boldsymbol{u}}\\ &= \sum_{i=1}^{N} \pi_{\boldsymbol{U}}^{i} \mathcal{N}(\boldsymbol{u}; \boldsymbol{u}^{(i)}, (H_{\boldsymbol{Y}})_{\boldsymbol{U}})\\ &\rightsquigarrow \mathcal{GK}(\boldsymbol{u}; \mathcal{S}_{\boldsymbol{U}}, \boldsymbol{\pi}_{\boldsymbol{U}}, H_{\boldsymbol{U}})\end{aligned}$$

where the fourth equality is given by Proposition 3.1. ■

It should be noted that the number of the parameters required is reduced under marginalization. The mixing weights $\boldsymbol{\pi}_{\boldsymbol{Y}} = (\pi_{\boldsymbol{Y}}^{1}, ..., \pi_{\boldsymbol{Y}}^{N})$ are preserved in the marginal, $\boldsymbol{\pi}_{\boldsymbol{U}} = \boldsymbol{\pi}_{\boldsymbol{Y}}$.

Proposition 4.2 (Conditional Gaussian kernel) *Let* $\boldsymbol{Y} = (Y_1, ..., Y_n)$ *follow an* n*-dimensional Gaussian kernel probability density function,* $f(\boldsymbol{y}) \rightsquigarrow \mathcal{GK}(\boldsymbol{y}; \mathcal{S}_{\boldsymbol{Y}}, \boldsymbol{\pi}_{\boldsymbol{Y}}, H_{\boldsymbol{Y}})$, *and let* $\boldsymbol{U} = (U_1, ..., U_k) \subseteq \boldsymbol{Y}$ *and* $\bar{\boldsymbol{U}} = \boldsymbol{Y} \setminus \boldsymbol{U}$. *The conditional joint density function of* $\boldsymbol{U}$ *conditioned to* $\bar{\boldsymbol{U}} = \bar{\boldsymbol{u}}$, *follows an* l*-dimensional joint Gaussian kernel density function with parameters* $\mathcal{S}_{\boldsymbol{U}|\bar{\boldsymbol{u}}}$, $\boldsymbol{\pi}_{\boldsymbol{U}|\bar{\boldsymbol{u}}} = (\pi_{\boldsymbol{U}|\bar{\boldsymbol{u}}}^{1}, ..., \pi_{\boldsymbol{U}|\bar{\boldsymbol{u}}}^{N})$ *and* $H_{\boldsymbol{U}|\bar{\boldsymbol{U}}}$, $f(\boldsymbol{u} \mid \bar{\boldsymbol{u}}) = \frac{f(\boldsymbol{u}, \bar{\boldsymbol{u}})}{\int_{-\infty}^{\infty} f(\boldsymbol{u}, \bar{\boldsymbol{u}})d\boldsymbol{u}} \rightsquigarrow \mathcal{GK}(\boldsymbol{u}; \mathcal{S}_{\boldsymbol{U}|\bar{\boldsymbol{u}}}, \boldsymbol{\pi}_{\boldsymbol{U}|\bar{\boldsymbol{u}}}, H_{\boldsymbol{U}|\bar{\boldsymbol{U}}})$, *where*

$$\begin{aligned}\mathcal{S}_{\boldsymbol{U}|\bar{\boldsymbol{u}}} &= \{\boldsymbol{u}^{(1)}|\bar{\boldsymbol{u}}, ..., \boldsymbol{u}^{(N)}|\bar{\boldsymbol{u}}\}\\ \boldsymbol{u}^{(i)}|\bar{\boldsymbol{u}} &= \boldsymbol{u}^{(i)} + H_{\boldsymbol{U},\bar{\boldsymbol{U}}} H_{\bar{\boldsymbol{U}}}^{-1}(\bar{\boldsymbol{u}} - \bar{\boldsymbol{u}}^{(i)})\\ \pi_{\boldsymbol{U}|\bar{\boldsymbol{u}}}^{i} &= \frac{\pi_{\boldsymbol{Y}}^{i} \mathcal{N}(\bar{\boldsymbol{u}}; \bar{\boldsymbol{u}}^{(i)}, H_{\bar{\boldsymbol{U}}})}{\sum_{j=1}^{N} \pi_{\boldsymbol{Y}}^{j} \mathcal{N}(\bar{\boldsymbol{u}}; \bar{\boldsymbol{u}}^{(j)}, H_{\bar{\boldsymbol{U}}})}\\ H_{\boldsymbol{U}|\bar{\boldsymbol{U}}} &= H_{\boldsymbol{U}} - H_{\boldsymbol{U},\bar{\boldsymbol{U}}} H_{\bar{\boldsymbol{U}}}^{-1} H_{\bar{\boldsymbol{U}},\boldsymbol{U}}\end{aligned}$$

Proof. This proposition can be proved as follows:

$$
\begin{aligned}
f(\boldsymbol{u}|\bar{\boldsymbol{u}}) &= \frac{f(\boldsymbol{u},\bar{\boldsymbol{u}})}{\int_{-\infty}^{\infty} f(\boldsymbol{u},\bar{\boldsymbol{u}})d\boldsymbol{u}} \\
&= \frac{\sum_{i=1}^{N} \pi_{\boldsymbol{Y}}^{i} \mathcal{N}((\boldsymbol{u},\bar{\boldsymbol{u}});(\boldsymbol{u}^{(i)},\bar{\boldsymbol{u}}^{(i)}),H_{\boldsymbol{Y}})}{\sum_{i=1} \pi_{\boldsymbol{Y}}^{i} \mathcal{N}(\bar{\boldsymbol{u}};\bar{\boldsymbol{u}}^{(N)},H_{\bar{\boldsymbol{U}}})} \\
&= \sum_{i=1}^{N} \frac{\pi_{\boldsymbol{Y}}^{i} \mathcal{N}(\bar{\boldsymbol{u}};\bar{\boldsymbol{u}}^{(i)},H_{\bar{\boldsymbol{U}}})}{\sum_{j=1}^{N} \pi_{\boldsymbol{Y}}^{j} \mathcal{N}(\bar{\boldsymbol{u}};\bar{\boldsymbol{u}}^{(j)},H_{\bar{\boldsymbol{U}}})} \mathcal{N}(\boldsymbol{u};\boldsymbol{u}^{(i)}|\bar{\boldsymbol{u}},H_{\boldsymbol{U}|\bar{\boldsymbol{U}}}) \\
&= \sum_{i=1}^{N} \pi_{\boldsymbol{U}|\bar{\boldsymbol{u}}}^{i} \mathcal{N}(\boldsymbol{u};\boldsymbol{u}^{(i)}|\bar{\boldsymbol{u}},H_{\boldsymbol{U}|\bar{\boldsymbol{U}}}) \\
&\rightsquigarrow \mathcal{GK}(\boldsymbol{u};\mathcal{S}_{\boldsymbol{U}|\bar{\boldsymbol{u}}},\boldsymbol{\pi}_{\boldsymbol{U}|\bar{\boldsymbol{u}}},H_{\boldsymbol{U}|\bar{\boldsymbol{U}}})
\end{aligned}
$$

where the first equality is given by Definitions 1.14 and 1.15, and the second and third equalities hold due to Proposition 4.2 and Proposition 3.2, respectively. The explicit forms of the parameters $\boldsymbol{u}^{(i)}|\bar{\boldsymbol{u}}$ and $H_{\boldsymbol{U}|\bar{\boldsymbol{U}}}$ are described in Proposition 3.2. ■

Note that for $H_{\boldsymbol{Y}}$ diagonal, $H_{\boldsymbol{Y}} = diag(h_1^2,...,h_n^2)$, the parameters of $f(\boldsymbol{u}|\bar{\boldsymbol{u}})$ for any $\bar{\boldsymbol{u}}$ are given by $\mathcal{S}_{\boldsymbol{U}|\bar{\boldsymbol{u}}} = (\mathcal{S}_{\boldsymbol{Y}})_{\boldsymbol{U}}$ and $H_{\boldsymbol{U}|\bar{\boldsymbol{u}}} = H_{\boldsymbol{U}}$.

From here on, for the sake of brevity, we will refer to the joint Gaussian kernel probability density function as *Gaussian kernel distribution* (*GK distribution*).

We now generalize these results to multidimensional mixed random variables following the intuitions behind *mixed Gaussian distribution* (*MG distribution*). For this purpose we introduce what we call *mixed Gaussian kernel distribution* (*MGK distribution*).

Definition 4.2 (Joint mixed Gaussian kernel) *Let* $\boldsymbol{X} = (\boldsymbol{Y},\boldsymbol{Z}) = (X_1,...,X_{n+m})$ *be an* $(n+m)$*-dimensional mixed random variable, where* $\boldsymbol{Y} = (Y_1,...,Y_n)$ *is an* n*-dimensional continuous random variable and* $\boldsymbol{Z} = (Z_1,...,Z_m)$ *is an* m*-dimensional discrete random variable.* $\boldsymbol{X}$ *is said to follow an* $(n+m)$-dimensional mixed Gaussian kernel distribution *based on* $\mathcal{S}_{\boldsymbol{Y}}(\boldsymbol{z}) = \{\boldsymbol{y}^{(1:\boldsymbol{z})},...,\boldsymbol{y}^{(N_{\boldsymbol{z}}:\boldsymbol{z})}\}$, $\boldsymbol{\pi}_{\boldsymbol{Y}}(\boldsymbol{z}) = (\pi_{\boldsymbol{Y}}^1(\boldsymbol{z}),...,\pi_{\boldsymbol{Y}}^N(\boldsymbol{z}))$ *and* $H_{\boldsymbol{Y}}(\boldsymbol{z})$ *if its generalized joint probability distribution is given by:*

$$\rho(\boldsymbol{x}) = f(\boldsymbol{y}|\boldsymbol{z})p(\boldsymbol{z}) \rightsquigarrow \mathcal{MGK}(\boldsymbol{x};p(\boldsymbol{z}),\mathcal{S}_{\boldsymbol{Y}}(\boldsymbol{z}),\boldsymbol{\pi}_{\boldsymbol{Y}}(\boldsymbol{z}),H_{\boldsymbol{Y}}(\boldsymbol{z})) \tag{4.7}$$

for $p(\boldsymbol{z}) > 0$*, where* $p(\boldsymbol{z})$ *follows an* m*-dimensional multinomial probability mass function,* $p(\boldsymbol{z}) = Pr(\boldsymbol{Z} = \boldsymbol{z})$*, and* $f(\boldsymbol{y}|\boldsymbol{z})$ *follows an* n*-dimensional joint Gaussian kernel density function based on* $\mathcal{S}_{\boldsymbol{Y}}(\boldsymbol{z})$*,* $\boldsymbol{\pi}_{\boldsymbol{Y}}(\boldsymbol{z})$ *and* $H(\boldsymbol{z})$*, for all* $\boldsymbol{z}$*,* $f(\boldsymbol{y}|\boldsymbol{z}) \rightsquigarrow \mathcal{GK}(\boldsymbol{y};\mathcal{S}_{\boldsymbol{Y}}(\boldsymbol{z}),\boldsymbol{\pi}_{\boldsymbol{Y}}(\boldsymbol{z}),H(\boldsymbol{z}))$.

It should be highlighted that the definition of MGK distribution is similar to the definition of MG distribution, replacing Gaussian densities by GK distributions (see Definition 3.2). If $H_{\boldsymbol{Y}}(\boldsymbol{z})$ is diagonal for any $\boldsymbol{z}$, $H_{\boldsymbol{Y}}(\boldsymbol{z}) = diag(h_1(\boldsymbol{z})^2,...,h_n(\boldsymbol{z})^2)$, then we have that

$$f(\boldsymbol{y}|\boldsymbol{z}) = \sum_{i=1}^{N_{\boldsymbol{z}}} \pi_{\boldsymbol{Y}}^{i}(\boldsymbol{z}) \prod_{j=1}^{n} \mathcal{N}(\boldsymbol{y}_j; \boldsymbol{y}_j^{(i:\boldsymbol{z})}, h_j(\boldsymbol{z})^2) \tag{4.8}$$

Similarly to a MG distribution, the marginal of a MGK distribution is not always a MGK distribution. The marginalization of MGK distributions over continuous random variables always follows a MGK distribution but, on the other hand, the marginalization of MGK distributions over discrete random variables, in general, leads to variable bandwidth mixed Gaussian kernel density functions. The following two propositions formally introduce these facts giving the closed forms for the marginal and conditional MGK distributions.

Proposition 4.3 (Marginal mixed Gaussian kernel I) *Let $\boldsymbol{X} = (\boldsymbol{Y}, \boldsymbol{Z}) = (X_1, ..., X_{n+m})$ be an $(n + m)$-dimensional mixed random variable, being $\boldsymbol{Y} = (Y_1, ..., Y_n)$ an n-dimensional continuous random variable and $\boldsymbol{Z} = (Z_1, ..., Z_m)$ an m-dimensional discrete random variable, and let $\boldsymbol{U} = (U_1, ..., U_l) \subseteq \boldsymbol{Y}$ be an l-dimensional continuous random variable being $\bar{\boldsymbol{U}} = \boldsymbol{Y} \setminus \boldsymbol{U}$ its complement in $\boldsymbol{Y}$. If $\boldsymbol{X}$ follows an $(n + m)$-dimensional mixed Gaussian kernel distribution based on $\mathcal{S}(\boldsymbol{z})$, $\boldsymbol{\pi}_{\boldsymbol{Y}}(\boldsymbol{z}) = (\pi_{\boldsymbol{Y}}^{1}(\boldsymbol{z}), ..., \pi_{\boldsymbol{Y}}^{N_{\boldsymbol{z}}}(\boldsymbol{z}))$ and $H(\boldsymbol{z})$, $\rho(\boldsymbol{x}) \rightsquigarrow \mathcal{MGK}(\boldsymbol{x}; p(\boldsymbol{z}), \mathcal{S}_{\boldsymbol{Y}}(\boldsymbol{z}), \boldsymbol{\pi}_{\boldsymbol{Y}}(\boldsymbol{z}), H_{\boldsymbol{Y}}(\boldsymbol{z}))$, then $(\boldsymbol{U}, \boldsymbol{Z})$ follows the mixed Gaussian kernel distribution:*

$$\rho(\boldsymbol{u}, \boldsymbol{z}) = \int_{-\infty}^{\infty} \rho(\boldsymbol{x}) d\bar{\boldsymbol{u}} \rightsquigarrow \mathcal{MGK}((\boldsymbol{u}, \boldsymbol{z}); p(\boldsymbol{z}), \mathcal{S}_{\boldsymbol{U}}(\boldsymbol{z}), \boldsymbol{\pi}_{\boldsymbol{Y}}(\boldsymbol{z}), H_{\boldsymbol{U}}(\boldsymbol{z})) \tag{4.9}$$

where

$$\begin{aligned} \mathcal{S}_{\boldsymbol{U}}(\boldsymbol{z}) &= (\mathcal{S}_{\boldsymbol{Y}}(\boldsymbol{z}))_{\boldsymbol{U}} = \{\boldsymbol{u}^{(1:\boldsymbol{z})}, ..., \boldsymbol{u}^{(N_{\boldsymbol{z}}:\boldsymbol{z})}\} \\ \boldsymbol{\pi}_{\boldsymbol{U}}(\boldsymbol{z}) &= \boldsymbol{\pi}_{\boldsymbol{Y}}(\boldsymbol{z}) \\ H_{\boldsymbol{U}}(\boldsymbol{z}) &= (H_{\boldsymbol{Y}}(\boldsymbol{z}))_{\boldsymbol{U}} \end{aligned}$$

for all $\boldsymbol{z}$.

Proof.

$$\begin{aligned} \int_{-\infty}^{\infty} \rho(\boldsymbol{y}, \boldsymbol{z}) d\bar{\boldsymbol{u}} &= \int_{-\infty}^{\infty} p(\boldsymbol{z}) f(\boldsymbol{y}|\boldsymbol{z}) d\bar{\boldsymbol{u}} \\ &= p(\boldsymbol{z}) \int_{-\infty}^{\infty} f(\boldsymbol{y}|\boldsymbol{z}) d\bar{\boldsymbol{u}} \end{aligned} \tag{4.10}$$

where by Definition 4.2 $f(\boldsymbol{y}|\boldsymbol{z})$ is a GK distribution for any $\boldsymbol{z}$, $f(\boldsymbol{y}|\boldsymbol{z}) \rightsquigarrow \mathcal{GK}(\boldsymbol{y}; \mathcal{S}_{\boldsymbol{Y}}(\boldsymbol{z}), \boldsymbol{\pi}_{\boldsymbol{Y}}(\boldsymbol{z}), H_{\boldsymbol{Y}}(\boldsymbol{z}))$. Then, for any $\boldsymbol{z}$, by Proposition 4.1, we have that $\int_{-\infty}^{\infty} f(\boldsymbol{y}|\boldsymbol{z}) d\bar{\boldsymbol{u}} = f(\boldsymbol{u}|\boldsymbol{z})$ follows a GK distribution with parameters $\mathcal{S}_{\boldsymbol{U}}(\boldsymbol{z}) = (\mathcal{S}_{\boldsymbol{Y}}(\boldsymbol{z}))_{\boldsymbol{U}} = \{\boldsymbol{u}^{(1:\boldsymbol{z})}, ..., \boldsymbol{u}^{(N_{\boldsymbol{z}}:\boldsymbol{z})}\}$, $\boldsymbol{\pi}_{\boldsymbol{U}}(\boldsymbol{z}) = \boldsymbol{\pi}_{\boldsymbol{Y}}(\boldsymbol{z})$ and $H_{\boldsymbol{U}}(\boldsymbol{z}) = (H_{\boldsymbol{Y}}(\boldsymbol{z}))_{\boldsymbol{U}}$. ■

It should be noted that, similar to the case of GK density functions, the number of the parameters required is reduced under marginalization over continuous random variables. Moreover, the mixed weights $\boldsymbol{\pi}_{\boldsymbol{Y}}(\boldsymbol{z}) = (\pi^1_{\boldsymbol{Y}}(\boldsymbol{z}), ..., \pi^{N_{\boldsymbol{z}}}_{\boldsymbol{Y}}(\boldsymbol{z})$ are preserved in the marginal, $\boldsymbol{\pi}_{\boldsymbol{U}}(\boldsymbol{z}) = \boldsymbol{\pi}_{\boldsymbol{Y}}(\boldsymbol{z})$ for all $\boldsymbol{z}$.

Proposition 4.4 (Marginal mixed Gaussian kernel II) *Let $\boldsymbol{X} = (\boldsymbol{Y}, \boldsymbol{Z}) = (X_1, ..., X_{n+m})$ be an $(n+m)$-dimensional mixed random variable which follows a mixed Gaussian kernel distribution based on $\mathcal{S}_{\boldsymbol{Y}}(\boldsymbol{z}) = \{\boldsymbol{y}^{(1:\boldsymbol{z})}, ..., \boldsymbol{y}^{(N_{\boldsymbol{z}}:\boldsymbol{z})}\}$, $\boldsymbol{\pi}_{\boldsymbol{Y}}(\boldsymbol{z}) = (\pi^1_{\boldsymbol{Y}}(\boldsymbol{z}), ..., \pi^{N_{\boldsymbol{z}}}_{\boldsymbol{Y}}(\boldsymbol{z}))$ and $H(\boldsymbol{z})$, $\rho(\boldsymbol{x}) \rightsquigarrow \mathcal{MGK}(\boldsymbol{x}; p(\boldsymbol{z}), \mathcal{S}_{\boldsymbol{Y}}(\boldsymbol{z}), \boldsymbol{\pi}_{\boldsymbol{Y}}(\boldsymbol{z}), H(\boldsymbol{z}))$ being $\boldsymbol{Y} = (Y_1, ..., Y_n)$ an n-dimensional continuous random variable and $\boldsymbol{Z} = (Z_1, ..., Z_m)$ an m-dimensional discrete random variable, let $\boldsymbol{V} = (V_1, ..., V_l) \subseteq \boldsymbol{Z}$ be an l-dimensional discrete random variable being $\bar{\boldsymbol{V}} = \boldsymbol{Z} \setminus \boldsymbol{V}$ its complement in $\boldsymbol{Z}$. If the bandwidth matrix $H(\boldsymbol{v}, \bar{\boldsymbol{v}})$ is constant for any $\bar{\boldsymbol{v}}$ then $(\boldsymbol{Y}, \boldsymbol{V})$ follows the mixed Gaussian distribution*

$$\rho(\boldsymbol{y}, \boldsymbol{v}) \rightsquigarrow \mathcal{MGK}((\boldsymbol{y}, \boldsymbol{v}); p(\boldsymbol{v}), \mathcal{S}_{\boldsymbol{Y}}(\boldsymbol{v}), \boldsymbol{\pi}_{\boldsymbol{Y}}(\boldsymbol{v}), H_{\boldsymbol{Y}}(\boldsymbol{v}))$$

where

$$\begin{aligned}
p(\boldsymbol{v}) &= \sum_{\bar{\boldsymbol{v}}} p(\boldsymbol{v}, \bar{\boldsymbol{v}}) \\
\mathcal{S}_{\boldsymbol{Y}}(\boldsymbol{v}) &= \bigcup_{\bar{\boldsymbol{v}}} \mathcal{S}_{\boldsymbol{Y}}(\boldsymbol{v}, \bar{\boldsymbol{v}}) \\
\boldsymbol{y}^i(\boldsymbol{v}) &= \boldsymbol{y}(\boldsymbol{v}, \bar{\boldsymbol{V}})^{i:\boldsymbol{v}} \\
\pi^i_{\boldsymbol{Y}}(\boldsymbol{v}) &= p(\bar{\boldsymbol{v}}^{(i:\boldsymbol{v})}|\boldsymbol{v}) \pi_{\boldsymbol{Y}}(\boldsymbol{v}, \bar{\boldsymbol{V}})^{i:\boldsymbol{v}} \\
H_{\boldsymbol{Y}}(\boldsymbol{v}) &= H_{\boldsymbol{Y}}(\boldsymbol{v}, \bar{\boldsymbol{v}}) \text{ for any } \bar{\boldsymbol{v}}
\end{aligned}$$

Proof. The proof of this proposition is as follows

$$\begin{aligned}
\sum_{\bar{\boldsymbol{v}}} \rho(\boldsymbol{y}, \boldsymbol{z}) &= \sum_{\bar{\boldsymbol{v}}} p(\boldsymbol{v}, \bar{\boldsymbol{v}}) \sum_{i=1}^{N_{\boldsymbol{v},\bar{\boldsymbol{v}}}} \pi^i_{\boldsymbol{Y}}(\boldsymbol{v}, \bar{\boldsymbol{v}}) \mathcal{N}(\boldsymbol{y}; \boldsymbol{y}^{(i)}(\boldsymbol{v}, \bar{\boldsymbol{v}}), H_{\boldsymbol{Y}}(\boldsymbol{v}, \bar{\boldsymbol{v}})) \\
&= p(\boldsymbol{v}) \sum_{\bar{\boldsymbol{v}}} p(\bar{\boldsymbol{v}}|\boldsymbol{v}) \sum_{i=1}^{N_{\boldsymbol{v},\bar{\boldsymbol{v}}}} \pi^i_{\boldsymbol{Y}}(\boldsymbol{v}, \bar{\boldsymbol{v}}) \mathcal{N}(\boldsymbol{y}; \boldsymbol{y}^{(i)}(\boldsymbol{v}, \bar{\boldsymbol{v}}), H_{\boldsymbol{Y}}(\boldsymbol{v})) \\
&= p(\boldsymbol{v}) \sum_{i=1}^{N_{\boldsymbol{v}}} p(\bar{\boldsymbol{v}}^{(i:\boldsymbol{v})}|\boldsymbol{v}) \pi_{\boldsymbol{Y}}(\boldsymbol{v}, \bar{\boldsymbol{V}})^{i:\boldsymbol{v}} \mathcal{N}(\boldsymbol{y}; \boldsymbol{y}(\boldsymbol{v}, \bar{\boldsymbol{V}})^{i:\boldsymbol{v}}, H_{\boldsymbol{Y}}(\boldsymbol{v})) \\
&= p(\boldsymbol{v}) \sum_{i=1}^{N_{\boldsymbol{v}}} \pi^i_{\boldsymbol{Y}}(\boldsymbol{v}) \mathcal{N}(\boldsymbol{y}; \boldsymbol{y}^{(i)}(\boldsymbol{v}), H_{\boldsymbol{Y}}(\boldsymbol{v}))
\end{aligned}$$

where the first equality holds by Definition 4.2, the second equality is due to $H(\boldsymbol{v}) = H(\boldsymbol{v}, \bar{\boldsymbol{v}})$ for any $\bar{\boldsymbol{v}}$ and the third equality consist of a rearrangements of the terms of the second equality. Note that $\pi_{\boldsymbol{Y}}(\boldsymbol{v}, \bar{\boldsymbol{V}})^{i:\boldsymbol{v}} = \pi^j_{\boldsymbol{Y}}(\boldsymbol{v}, \bar{\boldsymbol{v}}^{(i:\boldsymbol{v})})$ and $\boldsymbol{y}(\boldsymbol{v}, \bar{\boldsymbol{V}})^{i:\boldsymbol{v}} = \boldsymbol{y}^{(j)}(\boldsymbol{v}, \bar{\boldsymbol{v}}^{(i:\boldsymbol{v})})$ where $j : \boldsymbol{v}, \bar{\boldsymbol{v}}^{(i:\boldsymbol{v})} = i : \boldsymbol{v}$ for $i = 1, ..., N_{\boldsymbol{v}}$ and $N_{\boldsymbol{v}} = \sum_{\bar{\boldsymbol{v}}} N_{\boldsymbol{v},\bar{\boldsymbol{v}}}$. ■

In the particular case of $\pi^i(\boldsymbol{v},\bar{\boldsymbol{v}}) = 1/N_{\boldsymbol{v},\bar{\boldsymbol{v}}}$ for $i = 1,...,N_{\boldsymbol{v},\bar{\boldsymbol{v}}}$ and $p(\boldsymbol{v},\bar{\boldsymbol{v}}) = N_{\boldsymbol{v},\bar{\boldsymbol{v}}}/N$, for any $(\boldsymbol{v},\bar{\boldsymbol{v}})$, the parameter $\pi^i_{\boldsymbol{Y}}(\boldsymbol{v})$ for $i = 1,...,N_{\boldsymbol{v}}$ is simply given by:

$$\pi^i_{\boldsymbol{Y}}(\boldsymbol{v}) = \frac{N_{\boldsymbol{v},\bar{\boldsymbol{v}}^{(i:\boldsymbol{v})}}}{N_{\boldsymbol{v}}} * \frac{1}{N_{\boldsymbol{v},\bar{\boldsymbol{v}}^{(i:\boldsymbol{v})}}} = \frac{1}{N_{\boldsymbol{v}}} \tag{4.11}$$

It should be noted that, by the conditional independence $CI(\boldsymbol{Y};\bar{\boldsymbol{V}}|\boldsymbol{V})$ (see Definition 1.17), for any $\bar{\boldsymbol{v}}$, the next equality holds:

$$f(\boldsymbol{y}|\boldsymbol{v}) = f(\boldsymbol{y}|\boldsymbol{v},\bar{\boldsymbol{v}}) \tag{4.12}$$

and, thus, $H_{\boldsymbol{Y}}(\boldsymbol{v},\bar{\boldsymbol{v}})$ is constant for any $\bar{\boldsymbol{v}}$. Therefore, Proposition 4.4 can used to obtain the marginal over $\bar{\boldsymbol{V}}$ of a MG distribution for $\boldsymbol{X}$ when $CI(\boldsymbol{U};\bar{\boldsymbol{V}}|\boldsymbol{V})$.

Similarly to MG distribution, and in contrast to the situation concerning marginals in MGK distribution, conditioning with any subset of variables preserves the MGK distribution. The following proposition gives the closed forms for the parameters of a conditional MGK distribution obtained from a MGK distribution.

Proposition 4.5 (Conditional mixed Gaussian kernel) *Let $\boldsymbol{X} = (\boldsymbol{Y},\boldsymbol{Z}) = (X_1,...,X_{n+m})$ be an $(n+m)$-dimensional mixed random variable which follows a mixed Gaussian kernel distribution based on $\mathcal{S}(\boldsymbol{z})$, $\boldsymbol{\pi}_{\boldsymbol{Y}}(\boldsymbol{z}) = (\pi^1_{\boldsymbol{Y}}(\boldsymbol{z}),...,\pi^N_{\boldsymbol{Y}}(\boldsymbol{z}))$ and $H(\boldsymbol{z})$, $\rho(\boldsymbol{x}) \rightsquigarrow \mathcal{MGK}(\boldsymbol{x};p(\boldsymbol{z}),\mathcal{S}(\boldsymbol{z}),\boldsymbol{\pi}_{\boldsymbol{Y}}(\boldsymbol{z}),H(\boldsymbol{z}))$, being $\boldsymbol{Y} = (Y_1,...,Y_n)$ an n-dimensional continuous random variable and $\boldsymbol{Z} = (Z_1,...,Z_m)$ an m-dimensional discrete random variable, and let $\boldsymbol{U} = (U_1,...,U_l) \subseteq \boldsymbol{Y}$ be an l-dimensional continuous random variable and $\boldsymbol{V} = (V_1,...,V_k) \subseteq \boldsymbol{Z}$ be a k-dimensional discrete random variable, being $\bar{\boldsymbol{U}} = \boldsymbol{Y} \setminus \boldsymbol{U}$ and $\bar{\boldsymbol{V}} = \boldsymbol{Z} \setminus \boldsymbol{V}$ their complements in $\boldsymbol{Y}$ and $\boldsymbol{Z}$, respectively. For each value $(\bar{\boldsymbol{u}},\bar{\boldsymbol{v}})$, $(\boldsymbol{U},\boldsymbol{V})$ follows a multivariate mixed Gaussian kernel distribution, $\rho(\boldsymbol{u},\boldsymbol{v}|\bar{\boldsymbol{u}},\bar{\boldsymbol{v}}) \rightsquigarrow \mathcal{MGK}(\boldsymbol{u},\boldsymbol{v};p(\boldsymbol{v}|\bar{\boldsymbol{u}},\bar{\boldsymbol{v}}),\mathcal{S}_{\boldsymbol{U}|\bar{\boldsymbol{u}}}(\boldsymbol{v},\bar{\boldsymbol{v}}),\boldsymbol{\pi}_{\boldsymbol{U}|\bar{\boldsymbol{u}}}(\boldsymbol{v},\bar{\boldsymbol{v}}),H_{\boldsymbol{U}|\bar{\boldsymbol{U}}}(\boldsymbol{v},\bar{\boldsymbol{v}}))$, where*

$$\begin{aligned}
p(\boldsymbol{v}|\bar{\boldsymbol{u}},\bar{\boldsymbol{v}}) &= p(\boldsymbol{v}|\bar{\boldsymbol{v}})\frac{\mathcal{GK}(\bar{\boldsymbol{u}};\mathcal{S}(\boldsymbol{v},\bar{\boldsymbol{v}})_{\bar{\boldsymbol{U}}},p_{\boldsymbol{Y}|\boldsymbol{v},\bar{\boldsymbol{v}}}(i),H(\boldsymbol{v},\bar{\boldsymbol{v}})_{\bar{\boldsymbol{U}}})}{\sum_{\boldsymbol{v}'}p(\boldsymbol{v}'|\bar{\boldsymbol{v}})\mathcal{GK}(\bar{\boldsymbol{u}};\mathcal{S}(\boldsymbol{v}',\bar{\boldsymbol{v}})_{\bar{\boldsymbol{U}}},p_{\boldsymbol{Y}|\boldsymbol{v}',\bar{\boldsymbol{v}}}(i),H(\boldsymbol{v}',\bar{\boldsymbol{v}})_{\bar{\boldsymbol{U}}})} \\
\mathcal{S}_{\boldsymbol{U}|\bar{\boldsymbol{u}}}(\boldsymbol{v},\bar{\boldsymbol{v}}) &= \{\boldsymbol{u}^{(1:\boldsymbol{z})}|\bar{\boldsymbol{u}},...,\boldsymbol{u}^{(N_{\boldsymbol{z}}:\boldsymbol{z})}|\bar{\boldsymbol{u}}\} \\
\boldsymbol{u}^{(i:\boldsymbol{v},\bar{\boldsymbol{v}})}|\bar{\boldsymbol{u}} &= \boldsymbol{u}^{(i:\boldsymbol{z})} + H(\boldsymbol{z})_{\boldsymbol{U},\bar{\boldsymbol{U}}}(H(\boldsymbol{z})_{\bar{\boldsymbol{U}}})^{-1}(\bar{\boldsymbol{u}} - \bar{\boldsymbol{u}}^{(i:\boldsymbol{z})}) \\
\pi^i_{\boldsymbol{U}|\bar{\boldsymbol{u}}}(\boldsymbol{v},\bar{\boldsymbol{v}}) &= \frac{\pi^i_{\boldsymbol{Y}}(\boldsymbol{z})\mathcal{N}(\bar{\boldsymbol{u}};\bar{\boldsymbol{u}}^{(i:\boldsymbol{z})},H(\boldsymbol{z})_{\bar{\boldsymbol{U}}})}{\sum_{j=1}^{N_{\boldsymbol{z}}}\pi^j_{\boldsymbol{Y}}(\boldsymbol{z})\mathcal{N}(\bar{\boldsymbol{u}};\bar{\boldsymbol{u}}^{(j:\boldsymbol{z})},H(\boldsymbol{z})_{\bar{\boldsymbol{U}}})} \\
H_{\boldsymbol{U}|\bar{\boldsymbol{U}}}(\boldsymbol{v},\bar{\boldsymbol{v}}) &= H(\boldsymbol{z})_{\boldsymbol{U}} - H(\boldsymbol{z})_{\boldsymbol{U},\bar{\boldsymbol{U}}}(H(\boldsymbol{z})_{\bar{\boldsymbol{U}}})^{-1}H(\boldsymbol{z})_{\bar{\boldsymbol{U}},\boldsymbol{U}}
\end{aligned}$$

Proof.

$$\rho(\boldsymbol{u},\boldsymbol{v}|\bar{\boldsymbol{u}},\bar{\boldsymbol{v}}) = p(\boldsymbol{v}|\bar{\boldsymbol{u}},\bar{\boldsymbol{v}})f(\boldsymbol{u}|\bar{\boldsymbol{u}},\boldsymbol{z})$$

where $\boldsymbol{z} = (\boldsymbol{v},\bar{\boldsymbol{v}})$.

For any $\boldsymbol{z}$, $f(\boldsymbol{u}|\bar{\boldsymbol{u}},\boldsymbol{z})$ can be seen as the conditional form of $f(\boldsymbol{u},\bar{\boldsymbol{u}}|\boldsymbol{z})$. Thus, for any $\boldsymbol{z}$, by Proposition 4.2, we have that $f(\boldsymbol{u}|\bar{\boldsymbol{u}},\boldsymbol{v},\bar{\boldsymbol{v}}) \rightsquigarrow \mathcal{GK}(\boldsymbol{u};\mathcal{S}_{\boldsymbol{U}|\bar{\boldsymbol{u}}}(\boldsymbol{z}),\boldsymbol{\pi}_{\boldsymbol{U}|\bar{\boldsymbol{u}}}(\boldsymbol{z}),H_{\boldsymbol{U}|\bar{\boldsymbol{U}}}(\boldsymbol{z}))$, where

$$\mathcal{S}_{\boldsymbol{U}|\bar{\boldsymbol{u}}}(\boldsymbol{z}) = \{\boldsymbol{u}^{(1:\boldsymbol{z})}|\bar{\boldsymbol{u}}, ..., \boldsymbol{u}^{(N_{\boldsymbol{z}}:\boldsymbol{z})}|\bar{\boldsymbol{u}}\}$$
$$\boldsymbol{u}^{(i:\boldsymbol{z})}|\bar{\boldsymbol{u}} = \boldsymbol{u}^{(i:\boldsymbol{z})} + H(\boldsymbol{z})_{\boldsymbol{U},\bar{\boldsymbol{U}}}(H(\boldsymbol{z})_{\bar{\boldsymbol{U}}})^{-1}(\bar{\boldsymbol{u}} - \bar{\boldsymbol{u}}^{(i:\boldsymbol{z})})$$
$$\pi^i_{\boldsymbol{U}|\bar{\boldsymbol{u}}}(\boldsymbol{z}) = \frac{\pi^i_{\boldsymbol{Y}}(\boldsymbol{z})\mathcal{N}(\bar{\boldsymbol{u}}; \bar{\boldsymbol{u}}^{(i:\boldsymbol{z})}, H(\boldsymbol{z})_{\bar{\boldsymbol{U}}})}{\sum_{j=1}^{N_{\boldsymbol{z}}} \pi^j_{\boldsymbol{Y}}(\boldsymbol{z})\mathcal{N}(\bar{\boldsymbol{u}}; \bar{\boldsymbol{u}}^{(j:\boldsymbol{z})}, H(\boldsymbol{z})_{\bar{\boldsymbol{U}}})}$$
$$H_{\boldsymbol{U}|\bar{\boldsymbol{U}}}(\boldsymbol{z}) = H(\boldsymbol{z})_{\boldsymbol{U}} - H(\boldsymbol{z})_{\boldsymbol{U},\bar{\boldsymbol{U}}}(H(\boldsymbol{z})_{\bar{\boldsymbol{U}}})^{-1}H(\boldsymbol{z})_{\bar{\boldsymbol{U}},\boldsymbol{U}}$$

Given $\bar{\boldsymbol{u}}$ and $\bar{\boldsymbol{v}}$, the term $p(\boldsymbol{v}|\bar{\boldsymbol{u}}, \bar{\boldsymbol{v}})$ is given by the following normalized quantity:

$$p(\boldsymbol{v}|\bar{\boldsymbol{u}}, \bar{\boldsymbol{v}}) = \frac{\rho(\bar{\boldsymbol{u}}, \boldsymbol{v}, \bar{\boldsymbol{v}})}{\sum_{\boldsymbol{v}'} \rho(\bar{\boldsymbol{u}}, \boldsymbol{v}', \bar{\boldsymbol{v}})} = p(\boldsymbol{v}|\bar{\boldsymbol{v}})\frac{f(\bar{\boldsymbol{u}}|\boldsymbol{v}, \bar{\boldsymbol{v}})}{\sum_{\boldsymbol{v}'} p(\boldsymbol{v}'|\bar{\boldsymbol{v}})f(\bar{\boldsymbol{u}}|\boldsymbol{v}', \bar{\boldsymbol{v}})}$$

where, by Proposition 4.3, $\rho(\bar{\boldsymbol{u}}, \boldsymbol{v}, \bar{\boldsymbol{v}}) = \rho(\bar{\boldsymbol{u}}, \boldsymbol{z}) \rightsquigarrow \mathcal{MGK}(\bar{\boldsymbol{u}}, \boldsymbol{z}; p(\boldsymbol{z}), \mathcal{S}(\boldsymbol{z})_{\bar{\boldsymbol{U}}}, \boldsymbol{\pi}_{\boldsymbol{Y}}(\boldsymbol{z}), H(\boldsymbol{z})_{\bar{\boldsymbol{U}}})$. Thus, by Definition 4.2, for each value $(\boldsymbol{v}, \bar{\boldsymbol{v}}) = (\boldsymbol{z})$, $f(\bar{\boldsymbol{u}}|\boldsymbol{z})$ is distributed according to $\mathcal{GK}(\bar{\boldsymbol{u}}; \mathcal{S}(\boldsymbol{z})_{\bar{\boldsymbol{U}}}, \boldsymbol{\pi}_{\boldsymbol{Y}}(\boldsymbol{z}), H(\boldsymbol{z})_{\bar{\boldsymbol{U}}})$. ■

Note that when $H_{\boldsymbol{Y}}(\boldsymbol{z})$ is diagonal, $H_{\boldsymbol{Y}}(\boldsymbol{z}) = diag(h_1(\boldsymbol{z})^2, ..., h_n(\boldsymbol{z})^2)$, we have that the parameters are simply given by $\mathcal{S}_{\boldsymbol{U}|\bar{\boldsymbol{u}}}(\boldsymbol{z}) = (\mathcal{S}_{\boldsymbol{Y}}(\boldsymbol{z}))_{\boldsymbol{U}}$ and $H_{\boldsymbol{U}|\bar{\boldsymbol{U}}}(\boldsymbol{z}) = (H_{\boldsymbol{Y}}(\boldsymbol{z}))_{\boldsymbol{U}}$, for any $\boldsymbol{z}$.

The results concerning MGK distributions can be used in order to adapt the classifier induction algorithms presented in Section 4.6 to mixed domains.

4.4 MGK distribution based estimators for measures of information theory

This section introduces a set of non-parametric estimators for the following measures of information theory: entropy, conditional entropy, mutual information, conditional mutual information and interaction information. Then it presents the explicit formulation of a set of quantities that are used by the classifier induction algorithms introduced in Section 4.6. The estimators are based on Gaussian kernel density estimation and multinomial distribution. Note that these estimators can be thought of as being based on MGK distribution. This point of view will be useful in order to analyze the mathematical properties of the obtained quantities.

The notation and the random variables used in this section were introduced at the beginning of Section 4.3. However, in this section, we need to consider the more general partition $\boldsymbol{X} = (X_1, ..., X_{m+n}) = (\boldsymbol{X}_1, ..., \boldsymbol{X}_d)$, where $\boldsymbol{X}_i = (\boldsymbol{Y}_i, \boldsymbol{Z}_i)$, $\bigcup_{i=1}^d \boldsymbol{Y}_i = \boldsymbol{Y}$ and $\bigcup_{i=1}^d \boldsymbol{Z}_i = \boldsymbol{Z}$, for the definition of interaction information.

All the measures based on information theory introduced in Chapter 1, Section 1.10 can be expressed as an average over X of the logarithm of a function $g(\boldsymbol{x}_1, ..., \boldsymbol{x}_d)$, that is:

$$t(\boldsymbol{X}_1; ...; \boldsymbol{X}_d) = E_{\boldsymbol{X}}[\log g(\boldsymbol{x}_1; ...; \boldsymbol{x}_d)] = E_{\boldsymbol{Z}}E_{\boldsymbol{Y}|\boldsymbol{z}}[\log g(\boldsymbol{x}_1; ...; \boldsymbol{x}_d)]$$

The function $g(\boldsymbol{x}_1, ..., \boldsymbol{x}_d)$ is defined in terms of a quotient of products of marginal and conditional functions obtained from $\rho(\boldsymbol{x})$. Depending on the measure of the information theory, the function g is defined as:

- Entropy, $H(\boldsymbol{X}) = H(\boldsymbol{Y}, \boldsymbol{Z})$: $g_H(\boldsymbol{x}) = \frac{1}{\rho(\boldsymbol{y},\boldsymbol{z})}$.
- Conditional entropy, $H(\boldsymbol{U}, \boldsymbol{V}|\bar{\boldsymbol{U}}, \bar{\boldsymbol{V}})$: $g_H(\boldsymbol{u}, \boldsymbol{v}; \bar{\boldsymbol{u}}, \bar{\boldsymbol{v}}) = \frac{1}{\rho(\boldsymbol{u},\boldsymbol{v}|\bar{\boldsymbol{u}},\bar{\boldsymbol{v}})}$.
- Mutual information, $I(\boldsymbol{U}, \boldsymbol{V}; \bar{\boldsymbol{U}}, \bar{\boldsymbol{V}})$: $g_I(\boldsymbol{u}, \boldsymbol{v}; \bar{\boldsymbol{u}}, \bar{\boldsymbol{v}}) = \frac{\rho(\boldsymbol{u},\bar{\boldsymbol{u}},\boldsymbol{v},\bar{\boldsymbol{v}})}{\rho(\boldsymbol{u},\boldsymbol{v})\rho(\bar{\boldsymbol{u}},\bar{\boldsymbol{v}})}$.
- Interaction information, $I(\boldsymbol{X}_1; ...; \boldsymbol{X}_d)$: $g_I(\boldsymbol{x}_1; ...; \boldsymbol{x}_d) = \prod_{(\boldsymbol{U},\boldsymbol{V}) \subseteq \{\boldsymbol{X}_1,...,\boldsymbol{X}_d\}} (-1)^{l-|(\boldsymbol{U},\boldsymbol{V})|} \rho(\boldsymbol{u}, \boldsymbol{v})$ where $|(\boldsymbol{U}, \boldsymbol{V})|$ represents the number of variables $\boldsymbol{X}_i$ included in $(\boldsymbol{U}, \boldsymbol{V})$.

A natural non-parametric estimator of the general measure $t(\boldsymbol{X}_1; ...; \boldsymbol{X}_d)$ can be given by

$$\begin{aligned}
\hat{t}(\boldsymbol{X}_1; ...; \boldsymbol{X}_d) &= \sum_{\boldsymbol{z}} \frac{p(\boldsymbol{z})}{N_{\boldsymbol{z}}} \sum_{i=1}^{N_z} \log \hat{g}((\boldsymbol{y}_1^{(i:\boldsymbol{z})}, \boldsymbol{z}_1^{(i:\boldsymbol{z})}); ...; (\boldsymbol{y}_d^{(i:\boldsymbol{z})}, \boldsymbol{z}_d^{(i:\boldsymbol{z})})) \\
&= \frac{1}{N} \sum_{\boldsymbol{z}} \sum_{i=1}^{N_{\boldsymbol{z}}} \log \hat{g}((\boldsymbol{y}_1^{(i:\boldsymbol{z})}, \boldsymbol{z}_1); ...; (\boldsymbol{y}_d^{(i:\boldsymbol{z})}, \boldsymbol{z}_d)) \\
&= \frac{1}{N} \sum_{i=1}^{N} \log \hat{g}((\boldsymbol{y}_1^{(i)}, \boldsymbol{z}_1^{(i)}); ...; (\boldsymbol{y}_d^{(i)}, \boldsymbol{z}_d^{(i)})) \qquad (4.13)
\end{aligned}$$

where $\hat{g}(\boldsymbol{x}_1; ...; \boldsymbol{x}_d)$ is obtained by modeling each factor of the form $\rho(\boldsymbol{x}_i|\boldsymbol{x}_j)$ using a mixed Gaussian kernel distribution for $(\boldsymbol{X}_i, \boldsymbol{X}_j) = (\boldsymbol{Y}_i, \boldsymbol{Y}_j, \boldsymbol{Z}_i, \boldsymbol{Z}_j)$, $\rho(\boldsymbol{y}_i, \boldsymbol{y}_j, \boldsymbol{z}_i, \boldsymbol{z}_j) \rightsquigarrow \mathcal{MGK}(\boldsymbol{y}_i, \boldsymbol{y}_j, \boldsymbol{z}_i, \boldsymbol{z}_j; p(\boldsymbol{z}_i, \boldsymbol{z}_j), \mathcal{S}_{\boldsymbol{Y}_i,\boldsymbol{Y}_j}(\boldsymbol{z}_i, \boldsymbol{z}_j), \boldsymbol{\pi}_{\boldsymbol{Y}_i,\boldsymbol{Y}_j}(\boldsymbol{z}_i, \boldsymbol{z}_j), H_{(\boldsymbol{Y}_i,\boldsymbol{Y}_j)}(\boldsymbol{z}_i, \boldsymbol{z}_j))$ where $\pi^i_{\boldsymbol{Y}_i,\boldsymbol{Y}_j}(\boldsymbol{z}_i, \boldsymbol{z}_j) = 1/N_{(\boldsymbol{z}_i,\boldsymbol{z}_j)}$.

The general purpose estimator presented in Equation 4.13 inherits the convergence properties of the kernel density estimation (see Section 4.7 for further details). Moreover, Equation 4.13 estimates the expectance of the function $\log(\hat{g}(\boldsymbol{x}_1, ..., \boldsymbol{x}_d))$, when the instances of the training set are independent and identically distributed [Bishop (2006)]. The estimation of the expectance becomes exact in the limit, as $N \to \infty$ [Bishop (2006)]. Another proposal based on kernels can be found in [Moon et al. (1995)]. Thus, we think that the estimators based on Equation 4.13 are suitable to estimate the measures of information theory such us entropy, conditional entropy, mutual information, conditional mutual information and interaction information (see Section 4.7).

The estimator for the conditional mutual information between two continuous multivariate random variables $\boldsymbol{U}$ and $\bar{\boldsymbol{U}}$ given $\boldsymbol{Z}$ is given by:

$$\begin{aligned}
&\hat{I}(\boldsymbol{U}, \bar{\boldsymbol{U}}|\boldsymbol{Z}) \qquad (4.14) \\
&= \frac{1}{N} \sum_{i=1}^{N} \log \frac{\hat{f}(\boldsymbol{y}^{(i)}; \mathcal{S}_{\boldsymbol{Y}}(\boldsymbol{z}^{(i)}), \boldsymbol{\pi}_{\boldsymbol{Y}}(\boldsymbol{z}), H_{\boldsymbol{Y}}(\boldsymbol{z}^{(i)}))}{\hat{f}(\boldsymbol{u}^{(i)}; \mathcal{S}_{\boldsymbol{U}}(\boldsymbol{z}^{(i)}), \boldsymbol{\pi}_{\boldsymbol{U}}(\boldsymbol{z}), H_{\boldsymbol{U}}(\boldsymbol{z}^{(i)}))\hat{f}(\bar{\boldsymbol{u}}^{(i)}; \mathcal{S}_{\bar{\boldsymbol{U}}}(\boldsymbol{z}^{(i)}), \boldsymbol{\pi}_{\bar{\boldsymbol{U}}}(\boldsymbol{z}), H_{\bar{\boldsymbol{U}}}(\boldsymbol{z}^{(i)}))}
\end{aligned}$$

where $H_{\boldsymbol{U}}(\boldsymbol{z}) = H_{\boldsymbol{Y}}(\boldsymbol{z})_{\boldsymbol{U}}$, $H_{\bar{\boldsymbol{U}}}(\boldsymbol{z}) = H_{\boldsymbol{Y}}(\boldsymbol{z})_{\bar{\boldsymbol{U}}}$, $\mathcal{S}_{\boldsymbol{U}}(\boldsymbol{z}) = \mathcal{S}_{\boldsymbol{Y}}(\boldsymbol{z})_{\boldsymbol{U}}$, $\mathcal{S}_{\bar{\boldsymbol{U}}}(\boldsymbol{z}) = \mathcal{S}_{\boldsymbol{Y}}(\boldsymbol{z})_{\boldsymbol{U}}$ and $\pi^i_{\boldsymbol{U}}(\boldsymbol{z}) = \pi^i_{\bar{\boldsymbol{U}}}(\boldsymbol{z}) = \pi^i_{\boldsymbol{Y}}(\boldsymbol{z}) = 1/N_{\boldsymbol{z}}$ for $i = 1, ..., N_{\boldsymbol{z}}$ by Proposition

4.5. We suggest computing the bandwidth matrix $H_{\boldsymbol{Y}}(\boldsymbol{z})$ using the normal rule plus differential scale approach with $l = |\boldsymbol{Y}| = |\boldsymbol{U}| + |\bar{\boldsymbol{U}}| = n$.

The estimator for the mutual information between a multivariate continuous variable $\boldsymbol{Y}$ and a multivariate discrete variable $\boldsymbol{Z}$ is given by:

$$\hat{I}(\boldsymbol{Y}, \boldsymbol{Z}) = \frac{1}{N} \sum_{i=1}^{N} \log \frac{\hat{f}(\boldsymbol{y}^{(i)}; \mathcal{S}_{\boldsymbol{Y}}(\boldsymbol{z}^{(i)}), \boldsymbol{\pi}_{\boldsymbol{Y}}(\boldsymbol{z}^{(i)}), H_{\boldsymbol{Y}}(\boldsymbol{z}^{(i)}))}{\hat{f}(\boldsymbol{y}^{(i)}; \mathcal{S}_{\boldsymbol{Y}}, \boldsymbol{\pi}_{\boldsymbol{Y}}, H_{\boldsymbol{Y}})} \tag{4.15}$$

where $\pi^i_{\boldsymbol{Y}}(\boldsymbol{z}) = 1/N_{\boldsymbol{z}}$ for $i = 1, ..., N_{\boldsymbol{z}}$ and $\pi^i_{\boldsymbol{Y}} = 1/N$ for $i = 1, ..., N$. In addition, $H_{\boldsymbol{Y}}(\boldsymbol{z})$ and $H_{\boldsymbol{Y}}$ can be estimated from $\mathcal{S}_{\boldsymbol{Y}}(\boldsymbol{z})$ and $\mathcal{S}_{\boldsymbol{Y}}$, respectively. In this case, we suggest the use of the normal rule plus differential scaled approach with $l = |\boldsymbol{Y}| = n$ to compute both $H_{\boldsymbol{Y}}(\boldsymbol{z})$ and $H_{\boldsymbol{Y}}$. It should be noted that both $\hat{f}(\boldsymbol{y}|\boldsymbol{z}; \mathcal{S}_{\boldsymbol{Y}}(\boldsymbol{z}), \boldsymbol{\pi}_{\boldsymbol{Y}}(\boldsymbol{z}), H_{\boldsymbol{Y}}(\boldsymbol{z}))$ and $\hat{f}(\boldsymbol{y}; \mathcal{S}_{\boldsymbol{Y}}, \boldsymbol{\pi}_{\boldsymbol{Y}}, H_{\boldsymbol{Y}})$ can not be GK distributions in general (see Proposition 4.4). Thus, this estimator can be dependent on linear transformations of the implied continuous random variables, and therefore it is advisable to transform the continuous random variables $Y_1, ..., Y_n$ in order to have similar dispersion values.

The estimators presented in Equations 4.14 and 4.15 have been presented for multidimensional random variables and they can be used, in their unidimensional form, by the classifier induction algorithms based on KBN paradigm presented in Section 4.6.

4.5 Kernel based Bayesian network

This section presents the third class of PGMs we are interested in, *kernel based Bayesian networks* (*KBN*) [Pérez et al. (2009)]. KBN can deal with mixed random variables while breaking with the strong Gaussian assumption made by CGNs. We propose a definition of the KBN paradigm based on MGK distributions.

Let us consider a PGM for an $(n + m)$-dimensional mixed random variable $\boldsymbol{X} = (X_1, ..., X_{n+m})$ with the proper subsets $\boldsymbol{Y} = (Y_1, ..., Y_n) = (X_{1:\boldsymbol{Y}}, ..., X_{n:\boldsymbol{Y}})$ and $\boldsymbol{Z} = (Z_1, ..., Z_m) = (X_{1:\boldsymbol{Z}}, ..., X_{m:\boldsymbol{Z}})$ as described in the beginning of Section 4.3. If (i) no discrete random variables have continuous parents, $\boldsymbol{Pc}(\boldsymbol{s})_{i:\boldsymbol{Z}} \cap \boldsymbol{Pa}(\boldsymbol{s})_{i:\boldsymbol{Z}} = \emptyset$ for $1 \leq i \leq m$, and (ii) the variable $\boldsymbol{X}$ follows a mixed kernel Gaussian distribution, $\rho(\boldsymbol{x}) \rightsquigarrow \mathcal{MKG}(\boldsymbol{x}; p(\boldsymbol{z}), \mathcal{S}_{\boldsymbol{Y}}(\boldsymbol{z}), \boldsymbol{\pi}_{\boldsymbol{Y}}(\boldsymbol{z}), H_{\boldsymbol{Y}}(\boldsymbol{z}))$ (see Definition 4.2), then the PGM for $\boldsymbol{X}$ is called *kernel based Bayesian network* (*KBN*) for $\boldsymbol{X}$. It should be highlighted that the definitions of CGN and KBN paradigms impose the same structural constraint. Thus, the structures of KBN paradigm are also decomposable [Lauritzen (1992)].

The graphical factorization of the generalized joint probability distribution for $\boldsymbol{X}$ encoded by a KBN for $\boldsymbol{X}$ is similar to the CGN paradigm but considering MGK distribution instead of MG distribution (see Equation 3.10).

We want to note that there are different ways of applying the normal rule (see Equation 4.5) to the local factors in order to obtain a KBN

which factorizes $\rho(\boldsymbol{x})$. We recommend fixing l taking into account the maximum number of continuous random variables implied in local densities $f(y_i|\boldsymbol{pc}(\boldsymbol{s})_{i:\boldsymbol{Y}}, \boldsymbol{pd}(\boldsymbol{s})_{i:\boldsymbol{Y}})$ for all i, $|Y_i| + |\boldsymbol{Pc}(\boldsymbol{s})_{i:\boldsymbol{Y}}|$. For example, in the case of TAN structures we recommend setting $l = 2$. Another approach could be to fix $l = n$.

4.6 Classifiers based on kernel based Bayesian networks: flexible classifiers

Since this dissertation initially focuses on continuous domains, this section and the experimental results presented in Section 4.8 are focused on continuous domains, that is, domains with only continuous predictor variables, being the class the single discrete variable. Thus, the domains are defined in terms of the random variables $(\boldsymbol{Y}, C) = (Y_1, ..., Y_n, C)$. Note that in a KBN for $(\boldsymbol{Y}, C)$ with an augmented naive Bayes structure, the parameters of the local factor $f(y_i|pc_i, c)$ can be obtained from $\mathcal{S}_{\boldsymbol{Y}}(c)$, $\boldsymbol{\pi}_{\boldsymbol{Y}}(c)$ and $H_{\boldsymbol{Y}}(c)$ for all c using Propositions 4.3 and 4.5, for $i = 1, ..., n$. Fortunately, the main difference between BMNs, CGNs and KBNs is related to the factors with continuous random variables. Thus, continuous domains can be considered appropriate to highlight the main differences between the three Bayesian network paradigms.

This section presents a set of classifier induction algorithms based on KBN for augmented naïve Bayes family of structures, ordered by their structural complexity: flexible naïve Bayes, flexible tree-augmented naïve Bayes, flexible k-dependent augmented naive Bayes and flexible complete graph Bayesian classifier. Examples of their different structure complexities are shown in Figure 2.4. Most of the algorithms have been adapted from BMN to the novel KBN paradigm.

4.6.1 Flexible naïve Bayes

The *naïve Bayes* classifier induction algorithm (*NB*) [Duda and Hart (1973); Langley et al. (1992); Minsky (1961)] learns the complete naïve Bayes structure, which is known *a priori*. Originally, NB classifier was introduced for the BMN (multinomial NB or mNB), and it was adapted to the KBN paradigm (flexible NB or fNB) by John and Langley (1995). This classifier has been recently used by Lerner and Lawrence (2001), Bouckaert (2004) and Lerner (2004). It must be noted that our version of the fNB uses the normal rule [Silverman (1986)] plus differential scaling with $l = 1$ for computing h, instead of the heuristic proposed by John and Langley (1995).

For further details on NB structures and NB classifier based on BMN paradigm see Section 2.5.2 and Section 2.5.3.1, respectively.

4.6.2 Flexible tree-augmented naïve Bayes

The tree-augmented naïve Bayes structures (TAN structures) break with the strong independence assumption made by NB structures, allowing probabilistic dependencies among predictors.

This section presents the novel adaptation to the KBN paradigm of fMTAN algorithm [Friedman et al. (1997)], originally introduced for BMN. The algorithm proposed by Friedman et al. (1997) follows the general outline of Chow and Liu's procedure [Chow and Liu (1968)], but instead of using the mutual information between two variables, it uses conditional mutual information between predictors given the class variable to construct the maximal weighted spanning tree. In order to adapt this algorithm to continuous domains, we estimate the mutual information between every pair of continuous predictor variables conditioned to the class variable $I(Y_i, Y_j|C)$, using the estimator proposed in Equation 4.14. In order to compute the bandwidth matrix using the normal rule plus differential scaling approach, we have set $l = 2$ in our experiments.

For further details on TAN structures and fMTAN classifier induction see Sections 2.5.2 and 2.5.3.2, respectively.

4.6.3 Flexible *k*-dependent augmented naïve Bayes

The k-dependent augmented naive Bayes structure (kAN structure) extends TAN structures allowing a maximum of k predictor parents plus the class for each predictor variable (NB, TAN and CG structures are equivalent to kDB structures with $k = 0$, $k = 1$ and $k = n - 1$, respectively).

This section introduces the novel adaptation of fMkDB classifier induction algorithm [Sahami (1996)] to the KBN paradigm. We call this adaptation *flexible k-dependent augmented naïve Bayes* (*fkAN*). This algorithm is a greedy approach which uses the class conditioned mutual information between each pair of predictor variables $I(Y_i, Y_j \mid C)$ and the mutual information between the class and each predictor $I(Y_i, C)$ to lead the structure search process. In order to adapt this algorithm to the KBN paradigm, we compute the amounts of mutual information $I(Y_i, Y_j \mid C)$ and $I(Y_i, C)$, using the estimators proposed in Equations 4.14 and 4.15, respectively. In order to compute the bandwidth matrix using the normal rule, we have set $l = k + 1$.

For further details on kAN structures and fMkAN see Sections 2.5.2 and 2.5.3.3, respectively.

4.6.4 Flexible complete graph classifier: Parzen window classifier

All the complete graphs represent the same factorization of the joint distribution, a factorization without any simplification derived from the conditional (in)dependence statements. We can assume any of the complete acyclic graphs for classification because they are Markov equivalent [Chickering (2002)].

Therefore, the structure of the *flexible complete graph* classifier (fCG) can be fixed randomly, provided that there are no cycles. We can assume any complete kAN structure with $k \geq n-1$. It should be noted that fCG is equivalent to the Parzen window classifier [Fukunaga (1972); Parzen (1962); Rosenblatt (1956)].

4.6.5 Storage and computational complexity

KBN based classifiers require more storage space and have greater computational costs than CGN based classifiers for learning and classifying new instances, respectively. First, we introduce the requirements associated with the structural learning, and then the requirements associated with the parametric learning. Finally, the requirements related to each of the presented classifier induction algorithms based on KBN are presented.

In order to obtain the structure to be modeled from data (structural learning), a subset of the classifier induction algorithms proposed in this section[†] estimates the mutual information between every pair of variables conditioned to the class, $I(Y_i, Y_j|C)$, and the mutual information between each variable and the class, $I(Y_i, C)$. The number of operations involved for computing $I(Y_i; Y_j|C)$, using the estimator given by Equation 4.14, is $\mathcal{O}(\sum_c N_c^2)$, where N_c is the number of instances in the partition of the training set induced by $C = c$. Thus, fTAN and fkAN algorithms require to compute $\mathcal{O}(n^2 \sum_c N_c^2)$ operations, where n is the number of continuous predictor variables included in the model. On the other hand, the computational cost for estimating $I(Y_i; C)$, with the estimator proposed in Equation 4.15, is $\mathcal{O}(N^2)$. Therefore, fkAN algorithm requires $\mathcal{O}(nN^2)$ computations additionally.

For modeling a classifier based on the KBN paradigm (using differentially scaled plus Normal rule approach) from continuous data given its structure (parametric learning), every instance in the training set and the variances of the predictors conditioned on each class label $C = c$ are stored. Note that the mixing weights are not stored because they can be obtained directly from data. Therefore, a classifier with kAN structure stores $\mathcal{O}(Nn+r(n+1))$ values (cases and parameters) to proceed with classification ($k = 0$ for NB, $k = 1$ for TAN and $k = n-1$ for CG). In order to classify a new instance, the same KBN based classifier requires to compute $\mathcal{O}(Nn(k+1))$ operations. Table 4.1 summarizes the storage and computational complexity for both CGN and KBN based classifiers given a kDB structure.

Therefore, the number of operations for learning each flexible classifier proposed is: $\mathcal{O}(Nn)$ for fNB [John and Langley (1995)] and fCG, $\mathcal{O}(n^2 \sum_c N_c^2)$ for fTAN, and $\mathcal{O}(n^2 \sum_c N_c^2 + nN^2)$ for fkAN. Due to the high computational requirements for learning the classifier and for classifying new instances, it can be said that flexible classifiers are more suitable for low and medium sized data sets.

[†] The structure induced with fNB and fCG classifier induction algorithms are given *a priori* and, therefore, their computational cost can be considered constant.

CGN based		KBN based	
Training space	Testing time	Training space	Testing time
$\mathcal{O}(rn^2)$	$\mathcal{O}(rn(k+1))$	$\mathcal{O}(Nn + r(n+1))$	$\mathcal{O}(Nn(k+1))$

Table 4.1. Storage and computational requirements of a classifier with a kAN structure based on both KBN paradigm, with the differential scaled bandwidth matrix plus the normal rule approach, and CGN paradigm. The domain has n continuous predictor variables and r classes, the training set has N cases. The computational requirements have been obtained for classifying a new instance.

4.7 Consistency of MGK distribution based results

In this section we discuss the asymptotic properties of MGK distribution as an estimator of the true underlying distribution given a training set $\mathcal{S}^N = \{\boldsymbol{x}^{(1)}, ..., \boldsymbol{x}^{(N)}\}$. This section is based on the random variables introduced in Section 4.3. It is also based on the definition of *strong pointwise consistency* and the theorems and lemmas presented by John and Langley (1995).

Definition 4.3 (Strong pointwise consistency) *If $\rho(\boldsymbol{x})$ is a generalized probability distribution for $\boldsymbol{X}$ and $\hat{\rho}_N(\boldsymbol{x})$ is an estimation of $\rho(\boldsymbol{x})$ based on N independent and identically distributed instances, $\mathcal{S}^N$, then $\hat{\rho}_N(\boldsymbol{x})$ is strongly pointwise consistent if $\hat{\rho}_N(\boldsymbol{x}) \to \rho(\boldsymbol{x})$ almost surely for all $\boldsymbol{x}$; i.e., for every $\epsilon > 0$, $p(\lim_{N\to\infty} |\hat{\rho}_N(\boldsymbol{x}) - \rho(\boldsymbol{x})| < \epsilon) = 1$.*

Note that we have modified the definition of strong pointwise consistency to take into account the mixed nature of the random variable $\boldsymbol{X}$. We consider the particular case of a MGK distribution for $\boldsymbol{X}$ learned from $\mathcal{S}^N$ when the bandwidth matrix is computed using the differential scaling plus normal rule approach. Moreover, we consider the particular case where the distribution $p(\boldsymbol{z})$ is estimated from $\mathcal{S}^N$ using a multinomial distribution with maximum likelihood parameters as assumed in [John and Langley (1995)]. Next, we proved the consistency of MGK distributions, KBN paradigm and the estimators of the measures of information theory based on MGK distributions.

John and Langley (1995) demonstrate, based on Casella and Berger (1990), the consistency of kernel density estimation for modeling continuous variables $\boldsymbol{Y}$ under some conditions regarding the estimator of the bandwidth matrix $H_{\boldsymbol{Y}}$ (*strong consistency for reals* theorem [John and Langley (1995)]). It must be noted that kernel density estimate using Gaussian kernel with the normal rule plus differential scaling approach satisfies the constraints of the strong consistency for reals. Thus, a GK density function for $\boldsymbol{Y}$ based on $\mathcal{S}^N$, $\hat{f}(\boldsymbol{y}) \rightsquigarrow GK(\boldsymbol{y}; \mathcal{S}^N_{\boldsymbol{Y}}, \boldsymbol{\pi}_{\boldsymbol{Y}}, H_{\boldsymbol{Y}})$ where $\pi^i_{\boldsymbol{Y}} = 1/N$ for $i = 1, .., N$ is a strong pointwise consistent estimator of $f(\boldsymbol{y})$.

Then, John and Langley (1995) demonstrate, based on Devroye (1983), the consistency of the multinomial distribution for $\boldsymbol{Z}$ as an estimator of $p(\boldsymbol{z})$ when the parameters are estimated by maximum likelihood. Furthermore, they demonstrate the consistency of the product of strongly consistent estimates (*consistency of products* [John and Langley (1995)]) and the related

consistency of the quotient (see the proof of Theorem 3 in John and Langley (1995)). Following, based on these lemmas and theorems, we demonstrate the strong consistency of MGK distribution for $\boldsymbol{X}$ as an estimator based on $\mathcal{S}^N$.

Lemma 4.1 (Strong consistency of MGK distribution) *The MGK estimator for the generalized probability function for* $\boldsymbol{X}$ *based on* $\mathcal{S}^N$, $\hat{\rho}_N(\boldsymbol{x}) \rightsquigarrow \mathcal{MGK}(\boldsymbol{x}; p(\boldsymbol{z}),$
$\mathcal{S}^N_{\boldsymbol{Y}}(\boldsymbol{z}), \boldsymbol{\pi}_{\boldsymbol{Y}}(\boldsymbol{z}), H_{\boldsymbol{Y}}(\boldsymbol{z}))$ *is strongly consistent.*

Proof.

$$\hat{\rho}_N(\boldsymbol{x}) = \hat{f}_N(\boldsymbol{y}|\boldsymbol{z})\hat{p}_N(\boldsymbol{z}) \rightarrow f(\boldsymbol{y}|\boldsymbol{z})p(\boldsymbol{z}) = \rho(\boldsymbol{x})$$

where the second equality holds due to the strong consistency for reals, strong consistency for nominals and strong consistency for the product. ■

Corollary 4.1 (Consistency of local density functions of KBN) *The estimates of the local factor* $f(y_i|\boldsymbol{pc}(\boldsymbol{s})_{i:\boldsymbol{Y}}, \boldsymbol{pd}(\boldsymbol{s})_{i:\boldsymbol{Y}})$ *based on* $\mathcal{S}^N$ *in the KBN framework,* $\hat{f}(y_i|\boldsymbol{pc}(\boldsymbol{s})_{i:\boldsymbol{Y}}, \boldsymbol{pd}(\boldsymbol{s})_{i:\boldsymbol{Y}}) \rightsquigarrow MGK(y_i; \mathcal{S}_{(\boldsymbol{Y}_i|\boldsymbol{pc}(\boldsymbol{s})_{i:\boldsymbol{Y}})}(\boldsymbol{pd}(\boldsymbol{s})_{i:\boldsymbol{Y}})$, $\boldsymbol{\pi}_{(\boldsymbol{Y}_i|\boldsymbol{pc}(\boldsymbol{s})_{i:\boldsymbol{Y}})}(\boldsymbol{pd}(\boldsymbol{s})_{i:\boldsymbol{Y}}), H_{(\boldsymbol{Y}_i|\boldsymbol{Pc}(\boldsymbol{s})_{i:\boldsymbol{Y}})}(\boldsymbol{pd}(\boldsymbol{s})_{i:\boldsymbol{Y}}))$, *is strongly consistent.*

This corollary is proved by the consistency of MGK distribution, strong consistency for the quotient and Proposition 4.5, taking into account that $\hat{f}(y_i|\boldsymbol{pd}(\boldsymbol{s})_{i:\boldsymbol{Y}}, \boldsymbol{pc}(\boldsymbol{s})_{i:\boldsymbol{Y}}) = \frac{\hat{f}(y_i, \boldsymbol{pd}(\boldsymbol{s})_{i:\boldsymbol{Y}}, \boldsymbol{pc}(\boldsymbol{s})_{i:\boldsymbol{Y}})}{\hat{f}(\boldsymbol{pd}(\boldsymbol{s})_{i:\boldsymbol{Y}}, \boldsymbol{pc}(\boldsymbol{s})_{i:\boldsymbol{Y}})}$ and $f(y_i|\boldsymbol{pd}(\boldsymbol{s})_{i:\boldsymbol{Y}}, \boldsymbol{pc}(\boldsymbol{s})_{i:\boldsymbol{Y}}) = \frac{f(y_i, \boldsymbol{pd}(\boldsymbol{s})_{i:\boldsymbol{Y}}, \boldsymbol{pc}(\boldsymbol{s})_{i:\boldsymbol{Y}})}{f(\boldsymbol{pd}(\boldsymbol{s})_{i:\boldsymbol{Y}}, \boldsymbol{pc}(\boldsymbol{s})_{i:\boldsymbol{Y}})}$.

Corollary 4.2 (Consistency of KBN) *The KBN estimate of the generalized probability function* $\rho(x)$ *based on* $\mathcal{S}^N$, $\hat{\rho}_N(\boldsymbol{x}) \rightsquigarrow \mathcal{MGK}(\boldsymbol{x}; p(\boldsymbol{z}), \mathcal{S}^N_{\boldsymbol{Y}}(\boldsymbol{z}), \boldsymbol{\pi}_{\boldsymbol{Y}}(\boldsymbol{z}),$ $H_{\boldsymbol{Y}}(\boldsymbol{z}))$ *is strongly consistent if the structure* $\boldsymbol{s}$ *of KBN is a CI-map.*

This corollary is proved by the consistency of MGK distribution, the strong consistency of local density functions, by the definition of CI map, and the strong consistency of the product. Note that a CI-map for $\boldsymbol{X}$ captures all the conditional dependencies included in $\rho(\boldsymbol{x})$. Besides, this corollary implies, by the consistency of the quotient, that the flexible classifiers which are a CI map of $\rho(\boldsymbol{x}, c)$ are strongly consistent.

Corollary 4.3 (Consistency of $\hat{t}(\boldsymbol{X}_1; ...; \boldsymbol{X}_k)$) *The estimates of the measures of information theory* $t(\boldsymbol{X}_1; ...; \boldsymbol{X}_k)$, $\hat{t}(\boldsymbol{X}_1; ...; \boldsymbol{X}_k)$, *is strongly consistent.*

This corollary is proved in two steps. Firstly, the consistency of $\hat{g}(\boldsymbol{X}_1; ...; \boldsymbol{X}_k)$ is proved due to the consistency of MGK distribution, the consistency of the product and consistency of the quotient. Note that $g(\boldsymbol{X}_1; ...; \boldsymbol{X}_k)$ is a function composed of the product and quotients of generalized probability functions defined over $\boldsymbol{X}_s \subseteq \{\boldsymbol{X}_1, ..., \boldsymbol{X}_k\}$. Then, the consistency of $\hat{t}(\boldsymbol{X}_1; ...; \boldsymbol{X}_k)$ is proved because Equation 4.13 exactly estimates the expectance of the function $\log(\hat{g}(\boldsymbol{x}_1, ..., \boldsymbol{x}_k))$ in the limit, as $N \rightarrow \infty$, when the instances of the training set are independent and identically distributed [Bishop (2006)].

4.8 Experimental results

This section presents the experimental results of previously introduced flexible classifiers.

First, in Section 4.8.1, we study the behavior of the flexible classifiers and their sensitivity to changes in the smoothing degree using artificial domains. Then, in Section B., we present a set of results in 21 *UCI repository* continuous data sets [Asuncion and Newman (2007)]. These results include a comparison of the classifiers using the Friedman plus Shaffer's static post-hoc test [García and Herrera (2008)] based on the estimated errors obtained in the data sets from UCI repository (Section 4.8.2.2). This section also includes the study of the effect of the smoothing degree on the performance of flexible classifiers (Section 4.8.2.3), and the bias plus variance decomposition of the expected error [Kohavi and Wolpert (1996)] (Section 4.8.2.4).

4.8.1 Artificial data sets

This section, which includes two subsections, illustrates the behavior of the flexible classifiers using artificial domains. Section 4.8.1.1 studies the classification performance of the NB and TAN classifiers based on the BMN, CGN and KBN paradigms, by means of six *ad-hoc* designed representative artificial data sets. The use of the flexible classifiers which model the correlation between predictors is justified, presenting their advantages. Section 4.8.1.2 studies the effect of the smoothing degree in the performance of the flexible classifiers by means of four artificial domains.

At each artificial domain a parameter λ is fixed in order to guarantee a Bayes error of $\epsilon_B = 0.1$. We define the error of a classifier M under the winner-takes-all rule [Duda et al. (2000)] as:

$$\epsilon_M = \int_{-\infty}^{\infty} f(\boldsymbol{y})(1 - p(c^*|\boldsymbol{y}))d\boldsymbol{y} \tag{4.16}$$

where $p(\cdot)$ and $f(\cdot)$ are the true probability distribution and density functions of the domain respectively, and $c^* = arg_c max[p_M(c|\boldsymbol{y})]$ under the winner-takes-all rule. The Bayes error ϵ_B is defined as the error of the Bayes classifier, which is obtained by the decision $c^* = arg_c max[p(c|\boldsymbol{y})]$ under the winner-takes-all rule.

4.8.1.1 Error estimation

This section illustrates the differences between NB and TAN classifiers based on the BMN (plus supervised discretization [Fayyad and Irani (1993)]), CGN and KBN paradigms. For this purpose, we have designed six representative artificial domains with two continuous predictors, $\boldsymbol{Y} = (Y_1, Y_2)$, and two equally probable classes, $c \in \{0, 1\}$ with $p(C = 0) = 0.5$. It must be noted that TAN,

kDB (at $k > 0$ values), and complete graph structures are equivalent in the bivariate domains. Taking into account the dependencies between predictor variables (correlated or non-correlated) and four types of densities (Gaussian, Gaussian mixture, chi square and chi square mixture), we propose the following six domains:

- Domains with non-correlated Gaussian predictors (ncG).
- Domains with correlated Gaussian predictors (cG).
- Domains with non-correlated Gaussian mixture predictors (ncGM).
- Domains with correlated Gaussian mixture predictors (cGM).
- Domains with non-correlated chi square predictors (ncCh).
- Domains with correlated chi square mixture predictors (cChM).

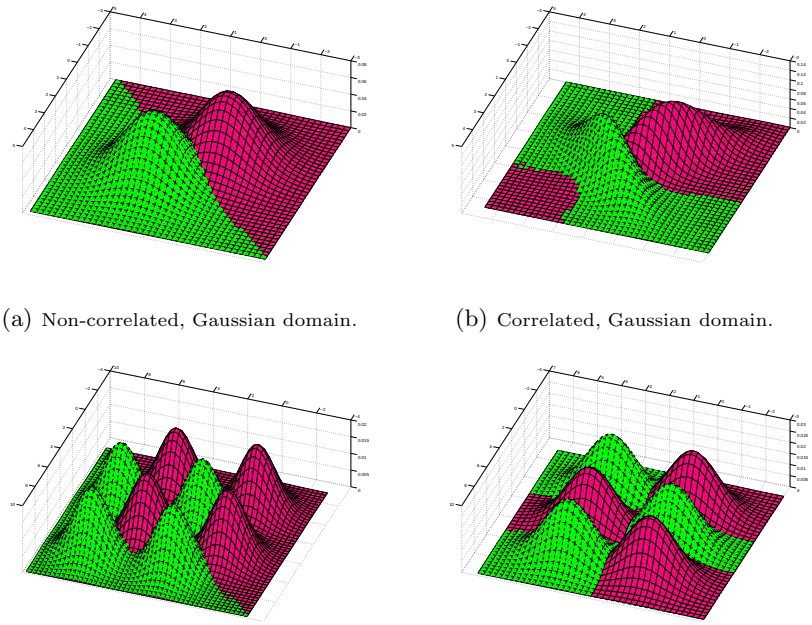
We have used the mixture of two or three densities for each class (ncGM, cGM and cChM) so as to model non-Gaussian densities. Note that these densities are different to a single Gaussian density. The chi square domains (ncCh and cChM) are based on the following pseudo-chi square density:

$$f_{chi}(y; y^0, \kappa) = \begin{cases} \frac{1}{2^{\kappa/2}\Gamma(\kappa/2)} y^{(\kappa/2)-1} e^{-(y-y^0)/2} & y - y^0 > 0 \\ 0 & \text{otherwise} \end{cases} \tag{4.17}$$

where κ specifies the number of degrees of freedom. In order to experiment with long tailed densities, we have set $\kappa = 4$. Besides, in contrast to the symmetry of the Gaussian density, pseudo-chi square has a skewness of $\sqrt{8/\kappa}$. The bivariate version of pseudo-chi square is defined as $f_{chi}(y_1, y_2; (y_1^0, y_2^0), \kappa) = f_{chi}(y_1; y_1^0, \kappa) \cdot f_{chi}(y_2; y_2^0, \kappa)$.

We have created the training and test sets by simulation for each of the exposed domains. Each simulated data set is defined by means of its associated generalized probability function $\rho(\boldsymbol{y}, c) = \rho(y_1, y_2, c) = p(c) f(y_1, y_2|c)$. In order to obtain a domain with dependent variables given the class, we use a function $f(y_1, y_2|c)$ which can not be decomposed into $f(y_1|c)f(y_2|c)$. The ncG, cG, ncGM, cGM, ncCh and cChM domains are defined by the following densities $f(y_1, y_2|c)$:

- The ncG data set is generated from the density functions defined by $f(y_1, y_2|c) = f(y_1|c)f(y_2|c)$ where $f(y_1|C = 0) \equiv f(y_2|0) \rightsquigarrow N(\mu = 0, \sigma^2 = 1)$ and $f(y_1|1) \equiv f(y_2|1) \rightsquigarrow N(\lambda, 1)$ with $\lambda = 1.8$ (see Figure 4.3(a)).
- The cG data set is defined by the density functions $f(y_1, y_2|0) \rightsquigarrow N(\boldsymbol{\mu} = (0, 0), \Sigma = [1, 0.75; 0.75, 1]$ and $f(y_1, y_2|1) \rightsquigarrow N((\lambda, \lambda), [1, -0.75; -0.75, 1])$ with $\lambda = 1.4$ (see Figure 4.3(b)).
- The ncGM data set is defined by the mixture functions $f(y_1|0) \equiv f(y_2|0) \rightsquigarrow 0.5N(0, 1)+0.5N(\lambda, 1)$ and $f(y_1|1) \equiv f(y_2|1) \rightsquigarrow 0.5N(\lambda/2, 1)+0.5N(3\lambda/2, 1)$ with $\lambda = 4.7$ (see Figure 4.3(c)).
- The cGM data set is defined by the mixture functions $f(y_1, y_2|0) \rightsquigarrow 1/3(N((0, 0), I = [1, 0; 0, 1])+N((\lambda, \lambda), I)+N((0, 2\lambda), I))$ and $f(y_1, y_2|1) \rightsquigarrow$



(a) Non-correlated, Gaussian domain.

(b) Correlated, Gaussian domain.

(c) Non-correlated, Gaussian mixture domain.

(d) Correlated, Gaussian mixture domain.

Fig. 4.3. Visualization of the artificial domains based on Gaussian and mixture of Gaussian densities. At each artificial domain $f(\boldsymbol{y}, c = 0)$ is represented with broken lines and a lighter surface, and $f(\boldsymbol{y}, 1)$ with continuous solid lines and a darker surface.

$1/3(N((\lambda, 0), I) + N((0, \lambda), I) + N((\lambda, 2\lambda), I))$ with $\lambda = 3.4$ (see Figure 4.3(d)).

- The ncCh data set is generated from the density functions defined by $f(y_1, y_2|0) \rightsquigarrow f_{chi}(y_1, y_2; (y_1^0, y_2^0) = (0, 0), \kappa = 4)$ and $f(y_1, y_2|1) \rightsquigarrow f_{chi}(y_1, y_2; (\lambda, \lambda), 4)$ with $\lambda = 3.5$ (see Figure 4.3(a)).
- The cChM data set is defined by the mixture functions $f(y_1, y_2|0) \rightsquigarrow 1/3[f_{chi}(y_1, y_2; (0, 0), 4) + f_{chi}(y_1, y_2; (\lambda, \lambda), 4) + f_{chi}(y_1, y_2; (0, 2\lambda), 4)]$ and $f(y_1, y_2|1) \rightsquigarrow 1/3[f_{chi}(y_1, y_2; (0, \lambda), 4) + f_{chi}(y_1, y_2; (\lambda, 0), 4) + f_{chi}(y_1, y_2; (\lambda, 2\lambda), 4)]$ with $\lambda = 7.9$ (see Figure 4.3(d)).

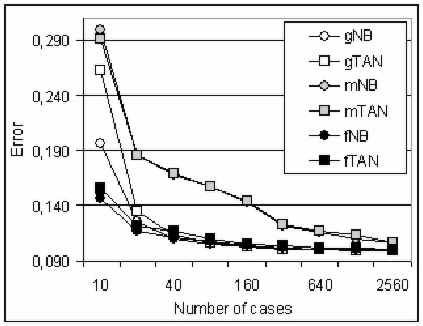
It must be noted that the ncCh and cChM domains are similar to the ncG and cGM (see Figure 4.3), replacing Gaussian by pseudo-chi square densities.

In order to study the errors of different classifiers (mNB, mTAN, gNB, gTAN, fNB and fTAN) with different training set sizes, at each artificial domain proposed, we have sampled training sets with 10, 20, 40, 80, 160, 320, 640, 1280 and 2560 cases. The experiment has been repeated 50 times for

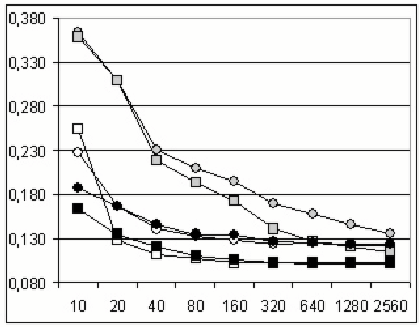
estimating the errors in each training set size. All the classifiers learned have been tested on an independent data set of 3000 cases. The variables have been discretized using the entropy algorithm [Fayyad and Irani (1993)] for each training-test pair, learning the discretization policies from the training set. The evolution of the errors estimated, with each classifier learned, is presented in Figure 4.4.

From the results shown in Figure 4.4, the following conclusions can be obtained:

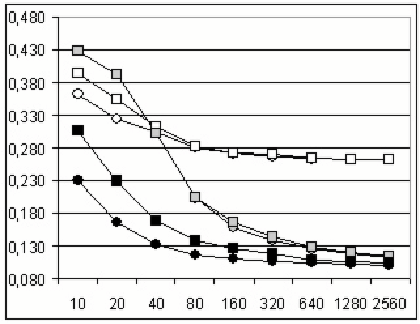
- In the ncG domain (see Figure 4.3(a)) all the classifiers reach the Bayes error. Gaussian and flexible classifiers behave similarly (see Figure 4.4(a)). They reach the Bayes error with training sizes of 160 or greater. This suggests that ncG domains can be modeled similarly for classifying using the kernel density estimation and the parametric Gaussian approach. The multinomial classifiers show a slower learning curve.
- In the cG domain (see Figure 4.3(b)) the classifiers which do not model the correlation between predictors (gNB, mNB and fNB) seem to behave somewhat worse than gTAN and fTAN (see Figure 4.4(b)). Gaussian and fTAN reach the Bayes error with a training size of 320 or greater. By contrast, mTAN does not reach the Bayes error but behaves a little better than NB classifiers. These results suggest that cG domains could generally be better modeled by Gaussian and flexible classifiers which consider the correlation between predictor variables. Thus, gTAN and fTAN show a better behavior than gNB and fNB respectively.
- In the ncGM domain (see Figure 4.3(c)) the flexible classifiers behave notably better than Gaussian based classifiers (see Figure 4.4(c)). Flexible and multinomial classifiers reach the Bayes error, but flexible classifiers converge to the Bayes error quicker. Flexible classifiers can clearly model the ncGM domains better than Gaussian classifiers.
- In the cGM domain (see Figure 4.3(d)) fTAN classifier behaves clearly better than the rest of the classifiers (see Figure 4.4(d)). This result suggests the importance of modeling the correlation between predictors using kernel based density estimation for the cGM domains. Multinomial TAN plus discretization converges to the same error value as NB models. Each variable is discretized independently, in a univariate way. Thus, the discretization algorithm [Fayyad and Irani (1993)] loses useful information about the class conditional dependencies between predictor variables. It must be noted that the most used discretization algorithms are univariate [Dougherty et al. (1995)].
- The results in the ncCh domain are similar to those obtained in the ncGM. The flexible classifiers clearly behave better than Gaussian based classifiers (see Figure 4.4(e)). Flexible and multinomial classifiers reach the Bayes error but multinomial classifiers converge to the Bayes error quicker.
- The results in the cChM domain are similar to those obtained in the cGM. In the cChM domain, fTAN classifier behaves clearly better than



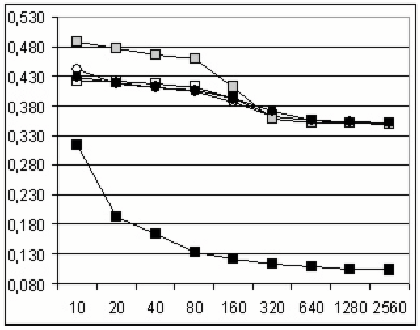
(a) Non-correlated, Gaussian domain, ncG.



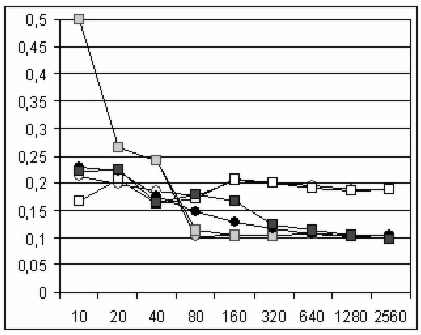
(b) Correlated, Gaussian domain, cG.



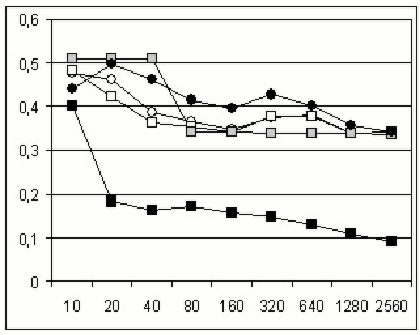
(c) Non-correlated, Gaussian mixture domain, ncGM.



(d) Correlated, Gaussian mixture domain, cGM.



(e) Non-correlated, chi square domain, ncCh.



(f) Correlated, chi square mixture domain, cChM

Fig. 4.4. The graphics represent the evolution, with respect to the number of cases, of the errors of different classifiers (mNB, mTAN, gNB, gTAN, fNB and fTAN) in each of the artificial domains proposed.

the rest of the classifiers (see Figure 4.4(f)). These results also suggest the importance of modeling the correlation between predictors with flexible

classifiers. Multinomial TAN plus discretization converges to the same error value as NB models. The discretization algorithm [Fayyad and Irani (1993)] loses useful information once again, due to its univariate nature.

Taking into account the four continuous domains presented, at each structural complexity level, the flexible classifiers seem to be at least as equally suitable as the Gaussian and multinomial classifiers (plus univariate supervised discretization [Fayyad and Irani (1993)]) for supervised classification in continuous domains. Flexible classifiers which model the true dependencies between predictor variables reach the Bayes error in the artificial domains proposed.

In Gaussian domains, the flexible classifiers show a learning curve similar to the Gaussian classifiers. Multinomial classifiers have a slower learning curve than flexible classifiers at each type of the proposed domain (except in the ncCh domain). fTAN, which models correlations between variables, obtains acceptable errors (less than 15%) with training set sizes of 20 in the ncG and cG domains, with 40 cases in the ncGM and cGM domains, and with 160 cases in the ncCh and cChM domains. Besides, it reaches the Bayes error with less than 1280 training cases in most of the domains. This suggests that flexible classifiers which model the true dependencies between variables could perform well in multi-modal domains, even with small sample sizes.

4.8.1.2 Analysis of the smoothing degree

This section illustrates the effect of the smoothing degree in the performance of flexible classifiers. For this purpose, we have designed four representative univariate artificial domains.

- Gaussian domain (Ga).
- Gaussian mixture domain (GaM).
- Chi square domain (Ch).
- Chi square mixture domain (ChM).

It must be noted that all the presented flexible classifiers are equivalent in the univariate domains.

The training and test sets of each of the four univariate artificial domains have been created by simulation. Each simulated data set is defined by means of its associated generalized probability function $\rho(y, c) = p(c)f(y|c)$. The domains have two equally probable classes, $c \in \{0, 1\}$ with $p(C = 0) = 0.5$. They are defined by the following densities $f(y|c)$:

- Ga domain: $f(y|C = 0) \rightsquigarrow N(y; \mu = 0, \sigma^2 = 1)$ and $f(y|1) \rightsquigarrow N(y; \lambda, 1)$ with $\lambda = 2.5$.
- GaM domain: $f(y|0) \rightsquigarrow 1/2[N(y; 0, 1)+N(y; 2\lambda, 1)]$ and $f(y|1) \rightsquigarrow 1/2[N(y; \lambda, 1) + N(y; 3\lambda, 1)]$ with $\lambda = 3.0$.
- Ch domain: $f(y|0) \rightsquigarrow f_{chi}(y; y^0 = 0, \kappa = 4)$ and $f(y|1) \rightsquigarrow f_{chi}(y; \lambda, 4)$ with $\lambda = 5.8$.

- ChM domain: $f(y|0) \rightsquigarrow 1/2[f_{chi}(y;0,4) + f_{chi}(y;2\lambda,4)]$ and $f(y|1) \rightsquigarrow f_{chi}(y;\lambda,4) + f_{chi}(y;3\lambda,4)]$ with $\lambda = 6.8$.

In order to study the errors of flexible classifiers with different training set sizes, at each artificial domain proposed, we have sampled training sets with 10, 25, 100 and 250 cases. The experiments have been repeated 50 times to estimate the errors for each training set size. We have generated 50 training sets for each train size and 50 test sets with 3000 cases. The variables have been discretized independently for each training-test pair. The discretization policies and the classifiers have been learned using the training set.

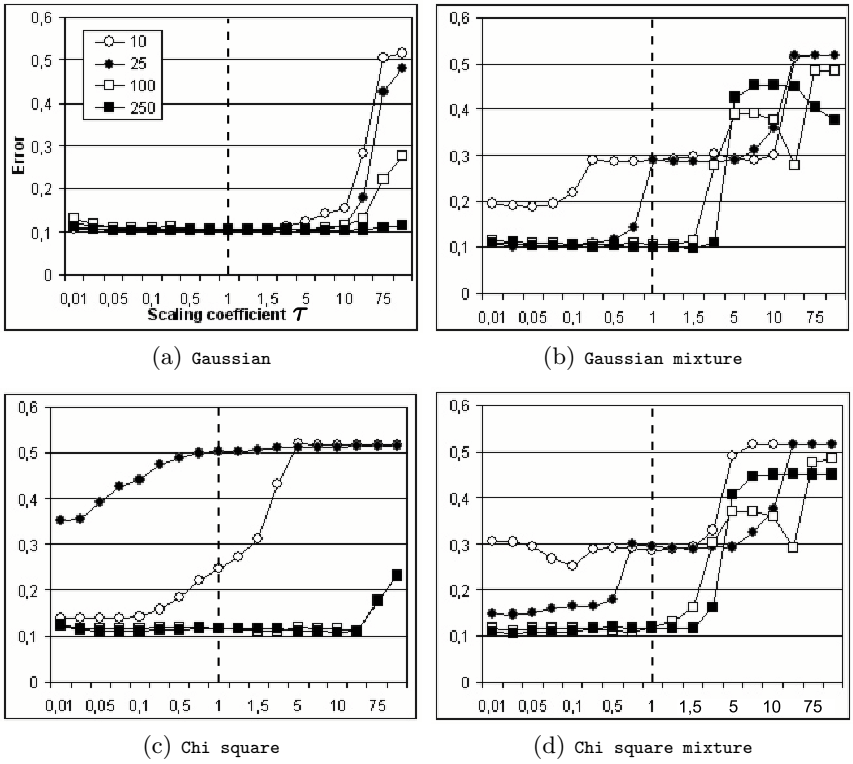
At each experiment we compute the smoothing parameter h using the normal rule plus differential scaled approach with $l = 1$ (see Section 4.2.1). Then, the bandwidth matrix $H = h^2$ is scaled using the following 18 coefficients, $\tau = \{0.01, 0.025, 0.05, 0.075, 0.1, 0.25, 0.5, 0.75, 1, 1.25, 1.5, 2.5, 5, 7.5, 10, 25, 75, 100\}$. For each scaled value of h^2 the error of the associated flexible classifier is estimated. Note that this scaling process can be understood as a wrapper optimization of the smoothing parameter, h. The evolution of the estimated errors with respect to the smoothing degree, τ, in the proposed artificial domains (Ga, GaM, Ch and ChM) are presented in Figure 4.5.

It should be noted that the study does not take into account the oversmooth effect of the parameter l in the normal rule plus differential scaled approach. However, when Equation 4.5 is analyzed, it is clear that the effect of the parameter l is lower than the parameter τ of our experimentation. For example, with $N = 25$ when $l = 1$ is set, we obtain $h = 0.37$, and if we set $l = 10^6$, $h \simeq 1$.

The standard deviation, σ, is the most common measure of statistical dispersion, measuring how widely spread the values in a data set are. The standard deviation is very sensitive to outliers. Note that the data obtained from long tailed densities generally presents outliers, e.g. chi square with $\kappa = 4$. This sensitivity is increased when the training set has few cases. It also increases when the density presents more than one mode, such as GaM and ChM domains. Under these conditions, the deviation overestimates the spread of the data. We say that the standard deviation is overestimating the spread if the interval $[\mu - \sigma, \mu + \sigma]$ contains more than 90% of the cases.

The normal rule tends to oversmooth the estimations. Taking into account that the standard deviation tends to overestimate the spread of the data, the effect of the oversmoothing of the normal rule is increased. Next, the results presented in Figure 4.4 are analyzed, taking into account the properties of the standard deviation and the normal rule:

- In the Ga domain (see Figure 4.5(a)) the standard deviation obtains a good measure of the spread of the data. In this case the normal rule obtains appropriate density estimations to classify.
- In the GaM domain (see Figure 4.5(b)) the deviation is overestimating the spread, especially with small training sets (10, 25). Given enough cases



(a) Gaussian (b) Gaussian mixture

(c) Chi square (d) Chi square mixture

Fig. 4.5. The effect of the smoothing degree, τh^2, in the performance of the flexible classifiers. Note that the scale coefficient τ amplifies or reduces the smoothing degree h^2 obtained using the normal rule, e.g. for $\tau = 1.25$ the smoothing degree given by h^2 is amplified 1.25 times. The errors obtained using the normal rule, $\tau = 1$, are marked using a dashed vertical line. This figure shows the competitive behavior of the normal rule in supervised classification compared to other smoothing degrees, $\tau \neq 1$.

(more than 25), the normal rule models appropriate densities with classification purposes.

- In the Ch domain (see Figure 4.5(c)) the deviation is overestimating the spread, especially with small training sets (10, 25). The normal rule obtains good estimations when the training set has enough cases (more than 25).
- In the ChM domain (see Figure 4.5(d)) the deviation overestimates the spread of the data, especially with small training sets. Besides, the normal rule increases the oversmoothing effect. In spite of the oversmoothing, the normal rule obtains appropriate density estimations with classification purposes when the training set has enough cases (more than 25).

These results suggest that it could be useful to compute a spread measure which is more robust to outliers than the standard deviation, e.g. the interquartile range. The normal rule tends to obtain enough good density estimations with classification purposes if the deviation does not overestimate the spread of the training data. Besides, flexible classifiers seem to have a good classification behavior even when the optimum smoothing degree is not used. Moreover, they seem to be quite insensitive to the optimum smoothing parameter, especially when the training set has enough cases (more than 25 in the univariate case). Note that, for training sizes of 100 and 250, Figure 4.5 presents error curves close to the Bayes error (0.1) at different smoothing degrees, τ: in Ga domain the error curves are close to the Bayes error for $\tau \in [0.01, 50]$ (see Figure 4.5(a)), in GaM domain for $\tau \in [0.01, 2.5]$ (see Figure 4.5(b)), in Ch domain for $\tau \in [0.01, 50]$ (see Figure 4.5(c)) and in ChM domain for $\tau \in [0.01, 2.5]$ (see Figure 4.5(d)).

On the other hand, Figure 4.5 suggests that the performance of the flexible classifiers could be improved when the optimization of the smoothing degree is performed in a wrapper way, especially for small databases.

4.8.2 UCI data sets

This section is divided in three parts. In Section 4.8.2.2 the classification error (see Equation 4.16) of the presented algorithms is estimated for each data set. Moreover, the results of two classifiers based on Bayesian networks (multinomial and Gaussian NB and TAN), k-nearest neighbour with $k = \{1, 3\}$, ID3 and C4.5 classification trees, quadratic discriminant analysis and multilayer perceptron are presented. Then, based on these results, flexible classifiers and the benchmarks included are compared across the UCI data sets by means of Friedman plus Shaffer's static post-hoc tests, as proposed in García and Herrera (2008). Finally, in Section 4.8.2.4, and in order to study the nature of the error of flexible classifiers, we perform the bias plus variance decomposition of the error [Kohavi and Wolpert (1996)] in a subset of the UCI data sets included.

4.8.2.1 Main characteristics and preprocessing

The results have been obtained in 21 *UCI repository* data sets [Asuncion and Newman (2007)], which only contain continuous predictor variables without missing values. As we indicate at the beginning of this section, we think that the continuous domains can be useful in order to highlight the main differences between BMN, CGN and KBN paradigms. Besides continuous domains allow the use of the most complex models, such as fCG. Note that fCG can be considered prohibitive for mixed domains with many discrete variables because the number of parameters of the model increases exponentially with the number of variables. The main characteristics of the data sets included are summarized in Table 5.2.

Label	r	n	N	Name of the data set
Balance	3	4	625	Balance Scale Weight & Distance
Block	5	10	5474	Page Block's Classification
Haberman	2	3	307	Haberman's Survival
Image	7	18	2310	Image Segmentation
Ionosphere	2	34	351	Johns Hopkins University Ionosphere
Iris	3	4	150	Iris Plant
Letter	26	16	20000	Letter Image Recognition
Liver	2	6	345	Bupa Liver Disorders
MFeatures	10	649	2000	Multiple Feature Digit
Musk	2	166	6598	Musk Clean2
Pendigits	10	16	10992	Pen-Digit Recognition of Handwritten Digits
Pima	2	8	768	Pima Indians Diabetes
Satellite	6	36	6435	Landsat Satellite
Sonar	2	60	208	Sonar, Mines VS Rocks
Spambase	2	57	4601	Spam e-mail database atributes
Thyroid	3	5	215	Thyroid Disease Records
Vehicle	4	19	846	Vehicle Silhouetes
Vowel	11	10	990	Vowel Recognition
Waveform	3	21	5000	Waveform Data Generation
Wine	3	13	179	Wine Recognition
Yeast	10	9	1484	Yeast

Table 4.2. Main characteristics of the data sets: the number of different values of the class variable (r), the number of predictor variables (n), and the number of instances (N).

In order to interpret the results, we must take into account that most data sets of the UCI repository are already preprocessed [Kohavi (1995b)]: in the data sets included, there are few irrelevant or redundant variables, and little noise [van der Putten and van Someren (2004)]. Therefore, it is more difficult to obtain statistically significant differences between the results of the algorithms in these types of data sets.

Some of the included data sets have a high-dimensional feature space (see the value n in Table 5.2). We have decided to reduce their dimensionality by a preprocessing stage to avoid the computational problems related with the flexible classifiers presented (especially for computing the conditioned mutual information among each pair of predictors, $I(Y_i; Y_j|C)$).

The preprocess stage is performed for each training-test pair separately. In order to estimate the error (Sections 4.8.2.2 and 4.8.2.3) at each data set, ten training-test pairs are generated using a 10-fold cross validation procedure. In the bias plus variance decomposition (Section 4.8.2.4), ten training-test pairs are generated. Firstly, in order to reduce the computational requirements of the algorithms proposed (especially for fTAN and fkDB), we have performed a feature selection process. If the number of predictors n is greater than 50, the dimensionality is reduced to 50 predictor variables. With this purpose, the mutual information with the class, $I(Y_i; C)$, is computed for each variable, Y_i, using the training set. Then, variables are sorted in descendant order of $I(Y_i; C)$ and the first 50 variables are selected. So as to perform the experimentation with the discrete classifiers, training and testing sets are discretized following the entropy algorithm proposed by Fayyad and Irani (1993), using the discretization policy learned in the training set.

4.8.2.2 Classification error estimation

The classification error has been estimated for the following flexible classifiers: flexible NB (fNB), flexible TAN (fTAN), flexible kDB for $k = 2$ (f2AN) and $k = n/2$ (fn.5AN), and flexible complete graph (fCG).

The benchmark classifiers selected are: Gaussian NB and TAN [Pérez et al. (2006b)] (gNB and gTAN), multinomial NB and TAN [Friedman et al. (1997)] (mNB and mTAN), k-nearest neighbour with $k = 1, 3$ [Cover and Hart (1967)] (k-NN) as lazy classifiers, ID3 [Quinlan (1986)] and C4.5 [Quinlan (1993)] as classification trees, and quadratic discriminant analysis (QDA) and multilayer perceptron [Rosenblatt (1959)] (MP) as discriminant functions. All of them, except gTAN [Pérez et al. (2006b)] and mTAN [Friedman et al. (1997)], are implemented in Weka 3.4.3 [Witten and Frank (2005)].

In order to estimate the classification error, for each flexible classifier and benchmark, at each data set, a stratified 10-fold cross-validation process has been performed. Stratified cross-validation obtains estimations with less variance than the standard cross-validation procedure [Diamantidis et al. (2000); Kohavi (1995a); Witten and Frank (2005)]. The validation was performed in a paired way, that is, the classifiers are trained and tested in the same train-test pairs. The estimated classification errors of the benchmarks and the flexible classifiers are summarized in Tables 4.3 and 4.4 respectively.

Using estimated errors in Tables 4.3 and 4.4, we compare the behavior of the flexible classifiers by means of the Friedman plus Shaffer's static post-hoc test [García and Herrera (2008)]. The results for Shaffer's static post-hoc test are shown by means of the critical difference diagrams introduced by Demšar (2006). These plots show the mean ranks of each model across all the domains in a numbered line. If there are not statistically significant differences between two classifiers, they are connected in the diagram by a straight line. For example, in Figure 4.6(c), at $\alpha = 0.1$ level, fTAN is significantly better than fCG. On the other hand, at $\alpha = 0.05$ the differences are not significant. A set of classifiers conform a cluster when they are connected in the diagram with the same line. Two clusters are significantly different (disjoint) when they are not connected.

Each test has been carried out at two significance levels $\alpha = \{0.1, 0.05\}$. The studies proposed are:

- Comparison of the benchmarks and the flexible classifier induction algorithms: fNB, fTAN, f2AN, fn.5AN and fCG.
- Comparison of the families of multinomial, Gaussian and flexible classifiers using the mNB, mTAN, gNB, gTAN, fNB and fTAN classifier induction algorithms.
- Comparison of the family of the flexible classifiers proposed.

The Friedman test for all the classifiers (benchmarks and flexible classifiers) rejects the null hypothesis at $\alpha = 0.05$. Thus, their errors can not be considered equivalent. The results for Shaffer's static post-hoc test and the

	k-NN		Classification Trees		Discriminant func.		CGN		BMN	
Data set	1-NN	3-NN	ID3	C4.5	QDA	MP	gNB	gTAN	mNB	mTAN
Balance	14.9 ± 3.8	14.7 ± 3.8	29.8 ± 4.6	23.0 ± 4.7	8.3 ± 2.8	9.0 ± 1.9	9.3 ± 1.0	11.5 ± 1.8	27.7 ± 4.8	28.2 ± 4.3
Block	3.8 ± 0.8	3.8 ± 0.8	3.9 ± 0.5	3.1 ± 0.5	6.2 ± 0.7	3.8 ± 0.7	10.1 ± 2.4	7.5 ± 0.8	6.6 ± 1.4	4.4 ± 0.5
Haberman	33.3 ± 3.8	32.0 ± 3.8	27.8 ± 3.8	28.4 ± 7.7	24.8 ± 4.3	26.5 ± 4.3	25.8 ± 4.9	24.5 ± 4.6	27.8 ± 3.8	27.8 ± 3.8
Image	3.1 ± 1.1	4.0 ± 1.2	7.5 ± 2.1	4.0 ± 1.5	19.6 ± 22.1	4.0 ± 1.3	21.1 ± 2.0	20.5 ± 8.5	11.3 ± 2.0	5.9 ± 1.7
Ionosphere	13.1 ± 4.5	14.8 ± 3.3	10.0 ± 6.2	11.7 ± 5.0	13.4 ± 4.3	11.4 ± 6.4	18.0 ± 6.3	7.7 ± 3.6	10.5 ± 4.0	8.6 ± 4.1
Iris	4.7 ± 6.7	5.3 ± 7.2	6.0 ± 8.1	7.3 ± 8.1	2.7 ± 4.4	3.3 ± 6.1	4.0 ± 6.8	3.3 ± 6.1	8.0 ± 7.8	7.3 ± 8.1
Letter	3.9 ± 0.5	4.4 ± 0.5	20.9 ± 0.6	12.1 ± 0.6	11.5 ± 0.7	17.4 ± 1.2	35.9 ± 1.4	28.3 ± 1.1	26.0 ± 1.2	14.5 ± 0.4
Liver	39.1 ± 4.6	37.9 ± 6.2	41.4 ± 2.9	33.3 ± 6.7	39.7 ± 5.7	32.2 ± 7.1	43.7 ± 7.8	39.7 ± 5.6	41.4 ± 2.9	41.4 ± 2.9
MFeatures	20.1 ± 1.7	18.1 ± 1.4	31.7 ± 2.0	23.6 ± 3.9	18.2 ± 2.3	16.9 ± 2.5	23.3 ± 2.7	19.7 ± 1.6	22.6 ± 2.5	20.7 ± 3.0
Musk	4.7 ± 0.9	3.9 ± 0.7	6.6 ± 1.3	4.1 ± 0.7	13.7 ± 1.2	2.6 ± 0.6	20.1 ± 1.8	23.8 ± 1.4	8.8 ± 1.1	6.0 ± 1.2
Pendigit	0.7 ± 0.3	0.7 ± 0.2	13.0 ± 1.3	3.7 ± 0.4	2.2 ± 0.5	6.2 ± 0.5	15.0 ± 1.1	7.3 ± 1.2	12.9 ± 0.9	4.5 ± 0.5
Pima	29.3 ± 4.0	27.2 ± 4.8	27.0 ± 5.4	24.9 ± 4.8	25.6 ± 3.7	25.0 ± 3.6	25.0 ± 3.3	24.6 ± 3.6	23.7 ± 3.7	23.0 ± 3.3
Satellite	9.6 ± 0.9	9.4 ± 1.1	18.9 ± 1.4	13.1 ± 1.4	14.4 ± 0.8	10.4 ± 1.0	20.5 ± 0.9	14.5 ± 1.6	17.9 ± 1.1	11.6 ± 1.4
Sonar	13.0 ± 4.8	16.4 ± 6.9	23.6 ± 7.3	26.0 ± 9.0	23.5 ± 6.7	13.9 ± 6.8	31.7 ± 7.0	31.7 ± 9.7	22.5 ± 9.2	22.5 ± 11.2
Spambase	8.9 ± 1.4	9.8 ± 1.4	9.5 ± 1.3	7.1 ± 1.5	19.3 ± 6.9	7.9 ± 1.3	18.0 ± 1.5	17.2 ± 1.5	10.3 ± 0.9	7.2 ± 0.8
Thyroid	3.3 ± 3.7	5.6 ± 5.4	8.4 ± 4.6	6.9 ± 4.6	3.2 ± 4.6	3.3 ± 3.0	4.2 ± 4.3	3.2 ± 4.6	5.0 ± 5.2	5.0 ± 5.5
Vehicle	30.4 ± 3.4	29.2 ± 2.5	32.7 ± 3.1	30.2 ± 4.5	40.2 ± 3.1	40.5 ± 4.7	54.3 ± 4.7	23.3 ± 4.0	30.6 ± 3.1	20.0 ± 5.3
Vowel	0.9 ± 0.8	3.1 ± 1.8	22.7 ± 4.6	20.4 ± 4.0	11.3 ± 3.0	20.5 ± 4.7	32.6 ± 2.9	22.7 ± 3.6	35.5 ± 4.4	28.1 ± 4.0
Waveform	22.5 ± 2.2	19.5 ± 1.5	30.3 ± 2.3	22.8 ± 2.0	15.1 ± 1.8	15.5 ± 1.7	19.0 ± 0.7	17.6 ± 2.1	18.8 ± 0.8	18.0 ± 2.6
Wine	5.7 ± 4.4	4.0 ± 4.5	6.2 ± 6.9	6.1 ± 5.2	0.6 ± 1.7	2.8 ± 4.5	2.8 ± 3.8	0.6 ± 1.8	1.7 ± 2.6	4.5 ± 4.9
Yeast	48.1 ± 3.8	47.2 ± 3.0	43.5 ± 4.4	44.1 ± 3.7	42.9 ± 3.8	40.6 ± 3.0	42.3 ± 5.0	42.9 ± 4.8	43.7 ± 4.0	43.2 ± 4.1
Average	14.9	14.8	20.1	16.9	17.0	14.9	22.7	18.7	19.7	16.8

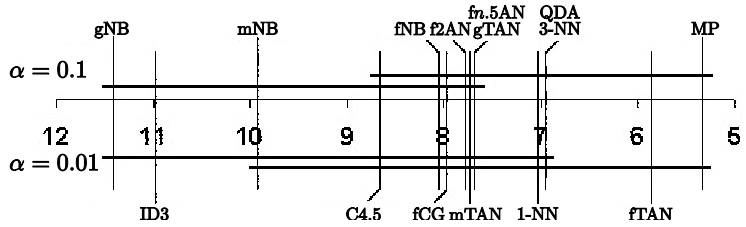
Table 4.3. The estimated errors obtained with a set of well known state-of-the-art algorithms. The best results, in each data set, are marked in grey.

Data set	fNB	fTAN	f2AN	fn.5AN	fCG
Balance	8.3 ± 1.2	10.7 ± 1.9	12.8 ± 2.6	12.8 ± 9.9	1.2 ± 0.7
Block	6.1 ± 0.6	5.2 ± 0.7	5.8 ± 0.9	6.3 ± 0.7	6.2 ± 0.7
Haberman	25.5 ± 5.9	24.8 ± 4.3	26.4 ± 4.7	24.8 ± 4.3	26.4 ± 4.7
Image	18.7 ± 1.4	15.7 ± 1.6	16.5 ± 1.4	18.4 ± 1.6	17.1 ± 1.7
Ionosphere	9.1 ± 4.6	7.1 ± 2.6	7.1 ± 3.2	10.0 ± 4.5	10.5 ± 5.0
Iris	4.0 ± 6.8	4.7 ± 6.7	4.7 ± 6.7	4.7 ± 6.7	3.3 ± 6.1
Letter	28.7 ± 1.2	15.7 ± 0.5	11.4 ± 0.7	7.7 ± 0.8	7.1 ± 0.8
Liver	33.6 ± 7.9	37.4 ± 8.4	37.4 ± 8.3	38.2 ± 6.7	40.8 ± 6.6
MFeatures	22.5 ± 3.4	18.5 ± 2.2	23.1 ± 2.6	18.6 ± 2.3	19.0 ± 2.3
Musk	13.4 ± 1.7	10.7 ± 1.6	18.3 ± 1.8	41.5 ± 1.7	40.0 ± 1.5
Pendigit	14.1 ± 1.0	4.3 ± 0.4	3.1 ± 0.5	3.1 ± 0.4	3.1 ± 0.3
Pima	24.3 ± 4.3	23.1 ± 4.3	25.4 ± 3.3	24.9 ± 4.7	25.8 ± 3.8
Satellite	18.3 ± 0.9	12.0 ± 0.9	12.2 ± 0.7	12.6 ± 0.6	12.2 ± 0.8
Sonar	26.9 ± 8.1	19.7 ± 5.5	24.5 ± 7.1	15.4 ± 5.5	16.8 ± 7.7
Spambase	22.8 ± 1.4	20.3 ± 1.6	25.1 ± 2.6	21.6 ± 1.9	20.9 ± 1.8
Thyroid	3.2 ± 4.6	4.2 ± 5.6	4.2 ± 5.6	4.2 ± 5.6	4.2 ± 5.6
Vehicle	29.7 ± 3.4	31.1 ± 3.6	33.6 ± 2.9	34.6 ± 3.8	35.6 ± 3.4
Vowel	26.0 ± 4.1	10.6 ± 3.9	5.1 ± 2.7	1.9 ± 1.0	1.9 ± 1.5
Waveform	19.3 ± 0.7	18.4 ± 1.7	19.6 ± 1.8	22.3 ± 1.9	20.5 ± 2.0
Wine	2.3 ± 3.7	1.1 ± 2.3	1.1 ± 2.3	1.7 ± 2.6	1.7 ± 2.5
Yeast	40.8 ± 3.9	41.0 ± 3.9	42.2 ± 3.1	42.5 ± 3.6	43.9 ± 3.4
Average	18.9	16.0	17.1	17.5	17.5

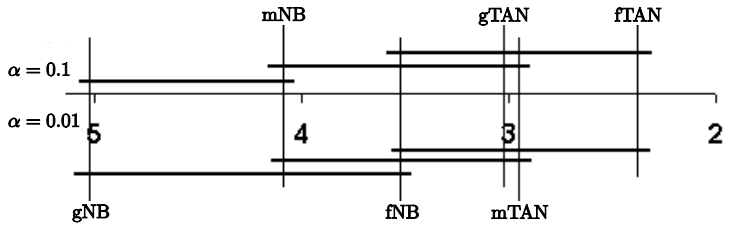
Table 4.4. The estimated errors and their corresponding standard deviations obtained with the presented flexible classifiers: flexible naïve Bayes (fNB), flexible tree-augmented naïve Bayes (fTAN), flexible k-dependent augmented naïve Bayes with $k = 2, n/2$ (f2AN, fn.5AN) and flexible complete graph classifier (fCG). The best results, in each data set, are marked in grey.

average ranks are shown in the critical difference diagrams [Demšar (2006)] in Figure 4.6(a). The analysis is quite similar at $\alpha = \{0.1, 0.05\}$. No disjoint cluster of algorithms is obtained, but the good average rank obtained by fTAN and MP, and the competitive results of the flexible classifiers should be highlighted.

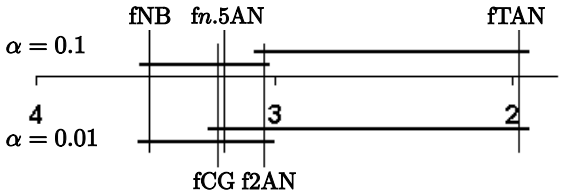
The comparison between the three BN based paradigms (CMN, CGN and KBN), using mNB, mTAN, gNB, gTAN, fNB and fTAN classifiers, shows two types of behaviors: families of NB and TAN classifiers. The Friedman test rejects the null hypothesis at $\alpha = \{0.1, 0.05\}$. The results for the Nemenyi test and the average ranks are shown in Figure 4.6(b). The analysis are similar at $\alpha = \{0.1, 0.05\}$. The different behavior of NB and TAN classifier families, especially at $\alpha = 0.1$, should be noted. Flexible classifiers, at each complexity level, seem to perform better than Gaussian and multinomial classifiers in terms of the average rank. However, the differences are not statistically significant using the Nemenyi test at $\alpha = 0.1$. It should be noted that using the Wilcoxon test, which has a larger discriminative power than Shaffer's static



(a) Flexible classifiers and benchmarks. In order to simplify the diagram, we have decided to include only the clusters associated with the best (MP) and the worst (gNB) classifiers in terms of average ranks.



(b) gNB, gTAN, mNB, mTAN, fNB and fTAN.

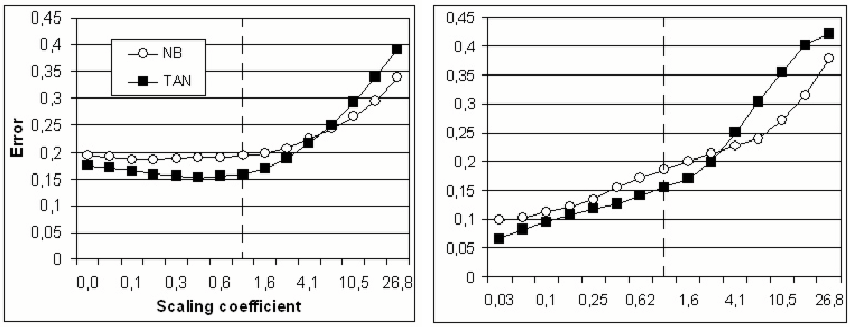


(c) Flexible classifiers.

Fig. 4.6. Critical difference diagrams Demšar (2006) representing Nemenyi post-hoc test and average ranks for different sets of classifiers. The test has been performed at $\alpha = \{0.1, 0.05\}$ significance levels as is suggested in [Demšar (2006)].

procedure, the fTAN shows significantly lower error levels than the mNB, mTAN, gNB, gTAN and fNB classifiers at $\alpha = 0.05$.

The last comparison has been performed between the flexible classifiers proposed. The Friedman test finds statistically significant differences in the overall behavior at $\alpha = \{0.1, 0.05\}$. The results of the Nemenyi test and the average ranks are shown in Figure 4.6(c). While fNB seems to show the worst behavior, fTAN shows the best one. The fTAN classifier obtains significantly better errors than fNB at $\alpha = 0.05$ (see Figure 4.6(c)). Moreover, fTAN obtains the highest number of best results across all data sets (see Table 4.4). The *curse of dimensionality* of the kernel density estimation [Duda et al. (2000); Scott (1992); Silverman (1986)] could explain the decrease of the per-



(a) `Average` (b) `Image`

Fig. 4.7. The graphs show the effect of the smoothing degree in the performance of flexible classifiers. A vertical dashed line indicates the normal rule plus differential scaled smoothing option. Figure 4.7(a) shows the average across all the data sets and it illustrates the good behavior of the normal rule plus differential scaled approach. The behavior of the error shown in Figure 4.7(a) is representative of most data sets included in this study. On the other hand, Figure 4.7(b) shows an abnormal behavior for which the normal rule plus differential scale approach obtains a bad performance.

formance of flexible classifiers while the structural complexity increases, such as from fTAN to fCG (see Table 4.4 and Figure 4.6(c)).

4.8.2.3 Smoothing degree

This section illustrates the effect of the smoothing degree in the flexible classifiers by means of 21 UCI databases (see Table 5.2). The error estimation is performed in the same way as the experimentation of Section 4.8.2.2. In order to study the effect of the smoothing degree, we have scaled the bandwidth matrix, which has been obtained with the normal rule plus differential scaled approach, using the following 15 coefficients $\tau = \{1.6^{-7}, ..., 1.6^{7}\} = \{0.037, ..., 26.843\}$. The coefficients cover a broad range of smoothness in a logarithmic scale. Due to the base 1.6 the jumps from one coefficient to the next are not too big. Note that this scaling process can be understood as a wrapper optimization of the smoothing parameter, h.

Figure 4.7 summarizes the main results of this section. For simplicity, as fTAN and fNB are representative of the flexible classifiers presented, we only include their results.

In the worst case (see Figure 4.7(b)), the differences between the error obtained with the normal rule plus differential scaled approach and the best smoothing degree are 8.6% and 9.0% for fNB and fTAN respectively. This result suggests that, when the optimization of the smoothing degree is performed in a wrapper way, the performance of the flexible classifiers can be

considerably improved in some domains (see Figure 4.7(b)). Note that in Image 1-NN classifier obtains the best performance (see Tables 4.3 and 4.4) and thus the error decreases as h tends to zero.

In spite of this, on average, the error difference with respect to the best smoothing degree value is only 0.7% for both fNB and fTAN (see Figure 4.7(a)). Besides, on average, the flexible classifiers obtain good results with the smoothing coefficient $\tau \in \{1.6^{-6}, ..., 1.6^{1}\}$. Therefore, it can be concluded that the normal rule plus differential scaled approach usually obtains good approximations for classification. Coupled with this, we think that, using a more robust spread measure than standard deviation, the flexible classifiers could obtain better results in some problematic domains, such as `Image`.

4.8.2.4 Bias plus variance decomposition of the expected error

In this section, we perform the bias plus variance decomposition in order to study, in each data set, the behavior of the error [Kohavi and Wolpert (1996)] of the flexible classifiers presented. We define the expected error of the zero-one loss function as:

$$E_M = \int_{-\infty}^{\infty} f(\boldsymbol{y}) \sum_{c=1}^{r} p(c|\boldsymbol{y})(1 - p_M(c|\boldsymbol{y}))d\boldsymbol{y} \tag{4.18}$$

where $p_M(\cdot)$ is the estimation modeled by a classifier and $p(\cdot)$ is the true distribution of the domain. The bias-variance decomposition of the expected error can be useful to explain the behaviors of different algorithms [van der Putten and van Someren (2004)]. For further details on the bias plus variance decomposition of the error see Section 1.8.3.

The decompositions have been performed following the procedure proposed by Kohavi and Wolpert (1996) with parameters $a = 10$ and $b = 1/3|\mathcal{S}|$, where a is the number of training sets $\{\mathcal{S}_1, ..., \mathcal{S}_a\}$, b is its size and $|\mathcal{S}|$ is the size of the entire data set. We have set $a = 10$ because the bias estimation is precise enough for this value (see Figure 1 of Kohavi and Wolpert (1996)), and due to its acceptable computational cost. We have set $b = 1/3|BD|$ to ensure a minimum training set size which could avoid overfitting problems. Kohavi and Wolpert (1996) choose a set of databases with at least 500 instances in order to ensure accurate estimates of the error. So as to obtain more robust conclusions, we have fulfilled this constraint and the following data sets have been selected for the experimentation: `Block`, `Image`, `Letter`, `MFeatures`, `Musk`, `Pima`, `Satellite`, `Vehicle`, `Vowel`, `Waveform` and `Yeast` (see Table 5.2).

Table 4.5 shows the results of the decomposition obtained for each classifier in the selected data sets following the criteria of Kohavi and Wolpert (1996) –at least 500 instances. It also includes an additional row which contains the average values for each classifier across all the data sets selected. For example fTAN obtains a $bias^2 = 11.6$ plus $var = 18.1$ decomposition for `Balance`,

Data set	fNB	fTAN	f2AN	fn.5AN	fCG
Balance	13.3 + 20.3	11.6 + 18.1	10.8 + 16.3	10.8 + 16.3	9.8 + 14.7
Block	3.5 + 1.3	3.2 + 1.3	3.4 + 1.3	3.6 + 1.2	3.7 + 1.2
Image	15.8 + 3.1	14.1 + 2.4	14.3 + 3.0	14.9 + 3.7	14.6 + 3.5
Letter	20.5 + 13.4	12.1 + 10.2	8.4 + 9.6	5.7 + 8.5	5.6 + 8.3
MFeatures	15.6 + 6.5	11.7 + 6.9	11.5 + 7.5	10.9 + 6.9	11.3 + 6.5
Musk	12.3 + 1.4	9.8 + 3.6	14.8 + 6.6	28.8 + 6.9	27.6 + 6.5
Pendigits	11.7 + 3.0	4.0 + 1.6	2.7 + 1.1	2.5 + 1.0	2.7 + 1.0
Pima	16.6 + 11.5	16.4 + 12.6	16.4 + 12.9	16.6 + 12.3	17.3 + 12.0
Satellite	17.3 + 1.3	9.1 + 3.1	8.4 + 4.7	8.3 + 4.8	8.1 + 4.6
Spambase	19.1 + 2.8	19.0 + 2.8	19.0 + 2.8	19.0 + 2.8	19.0 + 2.8
Vehicle	29.3 + 13.5	20.6 + 14.6	20.3 + 15.5	21.0 + 17.0	21.2 + 16.8
Vowel	21.1 + 24.7	11.1 + 17.5	8.9 + 14.7	7.2 + 12.1	6.9 + 11.7
Waveform	17.3 + 3.7	12.5 + 9.5	12.4 + 10.4	13.1 + 11.8	13.7 + 10.3
Yeast	31.1 + 27.2	30.7 + 27.5	31.2 + 27.4	31.2 + 27.1	31.9 + 27.2
Average	17.5 + 9.4	13.3 + 9.4	13.0 + 9.6	13.8 + 9.4	13.8 + 9.1

Table 4.5. Bias plus variance decomposition of the expected misclassification error rate.

and an *average* decomposition across all the data sets of $bias^2 = 13.3$ plus $var = 9.4$. Taking into account the results of Table 4.5, the following four behaviors could be highlighted (see Figure 4.8):

- The behavior of the decomposition with `Vehicle` data set (and in a similar way for `Waveform` and `Satellite`) is as follows (see Figure 4.8(a)):
  - Bias notably decreases in the jump from fNB to fTAN. On the other hand, fTAN, f2AN, fn.5AN and fCG obtain quite similar bias component values.
  - Variance slightly increases with the increase of the complexity.
- The behavior of the decomposition with `Vowel` data set (and in a similar way for `Balance`, `Letter`, `Pendigit` and `Vowel`) is as follows (see Figure 4.8(b)):
  - Bias decreases with the increase of the complexity.
  - Variance also decreases with the increase of the complexity.
- The behavior of the decomposition with `Pima` data set (and in a similar way for `Block`, `Image`, `Spambase` and `Yeast`) is as follows (see Figure 4.8(c)):
  - Bias and variance terms remain almost constant for all the flexible classifiers.
- Taking into account `Balance`, `Block`, `Image`, `Letter`, `MFeatures`, `Musk`, `Pendigits`, `Pima`, `Satellite`, `Spambase`, `Vehicle`, `Vowel`, `Waveform` and `Yeast` data sets, on average, the behavior of the decomposition is as follows (see Figure 4.8(d)):
  - Bias notably decreases in the jump from fNB to fTAN. On the other hand, fTAN, f2DB, fn.5DB and fCG obtain quite similar bias component values of the error. Therefore, it can be concluded that for the

selected data sets, fTAN model is precise enough for estimating the underlying density of the data.
– Variance remains almost constant with the increase of the complexity, and thus, all the models seem to have a similar sensitivity to changes in the training set.

In `Musk` data set the bias component of the fNB is clearly lower than fn.5AN and fCG. The mutual information between predictors conditioned on the class reveals that some predictor variables are independent with respect to each other. Thus, the compulsory addition of arcs could be the reason for the increment of the bias term.

In order to understand the behavior of the variance term, the following two observations should be taken into account:

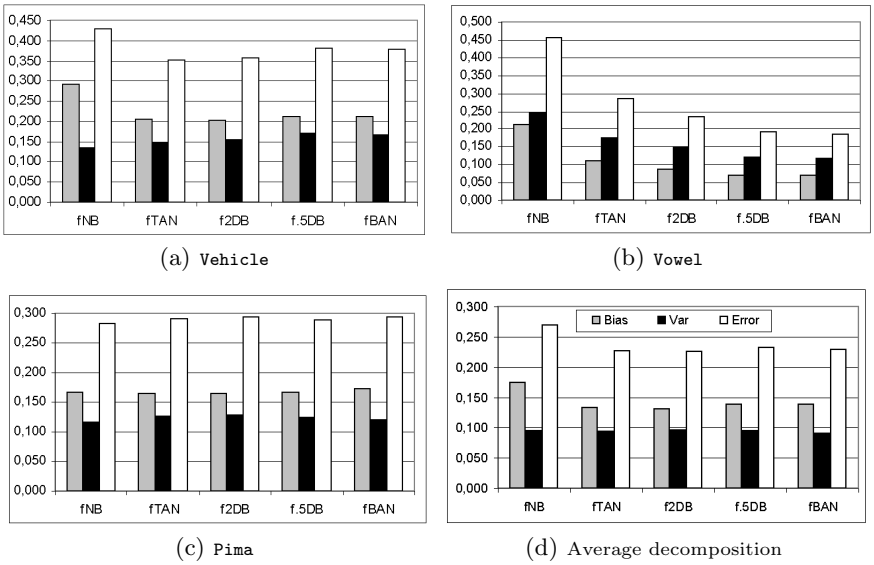
- The number of parameters (see Section 4.6.5) among the flexible classifiers presented is quite similar, and it is almost independent with respect to their structural complexity (see Table 4.1 with $k = 0$ for fNB and $k = d - 1$ for fCG). Thus, from the parametric learning point of view, the sensitivity to the changes in the training set (variance term) should be quite similar among flexible classifiers.
- Given a factorization of the density $f(\boldsymbol{y}|c)$, the probability mass of the density is more spread as fewer dependencies between variables are considered. For example, $f(y_1, y_2|c)$ is sharper than $f(y_1|c)f(y_2|c)$. Therefore, the classifiers which model more dependencies between variables tend to model an *a posteriori* distribution $p(c|\boldsymbol{y})$ with values nearest to 0 or 1 (*degenerated distributions*). If an instance $\boldsymbol{y}^{(i)}$ has a distribution more degenerated than another instance $\boldsymbol{y}^{(j)}$, then its variance should be lower, $var_{\boldsymbol{y}^{(i)}} < var_{\boldsymbol{y}^{(j)}}$ (see Equation 1.43).

The first observation explains the similar variance values for fNB, fTAN, fkDB and fCG classifiers in `Block`, `Image`, `Vehicle`, `MFeatures`, `Pima`, `Satimage` and `Spambase`. The first and the second observations could explain the slight decrease of the variance term as the structural complexity increases in `Vowel`, `Letter` and `Pendigits`.

Generally, among flexible classifiers, those which model correlations between predictor variables seem to be more suitable for modeling the underlying true densities of the variables, keeping at the same time the error due to the variance almost constant. Therefore, in the absence of prior knowledge about the true density underlying the data, it may be advisable to use flexible classifiers which model correlations between predictors.

4.9 Conclusions and future work

This chapter presents the novel *kernel based Bayesian network* (KBN) paradigm. A KBN is a Bayesian network which models the generalized probability distribution for $\boldsymbol{X}$, $\rho(\boldsymbol{x})$, using a mixed Gaussian kernel probability distribution.



(a) `Vehicle` (b) `Vowel`

(c) `Pima` (d) Average decomposition

Fig. 4.8. The graphics represent the evolution of the bias plus variance decomposition of the expected error for different flexible classifiers ordered by their structural complexity. The graphics include the decompositions for three representative data sets (`Vehicle` (a), `Vowel` (b) and `Pima` (c)) and for the average over the selected data sets (d).

Besides, this chapter proposes a general purpose estimator for the quantities of the information theory based on mixed Gaussian kernel distributions. Following the nomenclature introduced by John and Langley (1995), we use the term *flexible classifiers* to name the family of classifiers based on KBN. We have presented the following classifier induction algorithms for KBN: fNB, fTAN, fkAN and fCG.

We have demonstrated the strong consistency of mixed Gaussian kernel probability distributions as estimators of $\rho(\boldsymbol{x})$, of the KBN paradigm provided that it is a CI-map, of the estimators for the quantities of the mutual information $\hat{t}(\boldsymbol{X}_1; ...; \boldsymbol{X}_d)$, and for the flexible classifiers as estimators of $p(c|\boldsymbol{x})$ assuming that they are CI-maps.

The KBN paradigm can be considered as an alternative to Bayesian multinomial networks because they can directly handle continuous variables without needing to discretize them. In addition, it can also be considered as an extension of the conditional Gaussian network paradigm from the point of view of the flexibility of their approaches for estimating densities, because kernel based Bayesian network uses a non-parametric kernel based density estimation instead of a parametric Gaussian one. The kernel based estimation confers better flexibility properties than Gaussian approach in order to fit non-Gaussian densities. In summary, the factorizations codified by a KBN

tends to be better approaches to the true joint density and, therefore, the flexible classifiers tend to obtain error rates closer to the Bayes error. This idea is illustrated in Section 4.8.1 by means of a set of four artificial domains. In the UCI data sets included, flexible classifiers seems to outperform their Gaussian versions, and at least, equal to their multinomial versions. Besides, they obtain competitive results compared to the state-of-the-art algorithms selected.

Among the flexible classifiers proposed, the novel *flexible tree-augmented naïve Bayes* obtains the best ranking average and estimated errors in the selected domains. Besides, taking into account the bias plus variance decomposition of the expected error, it seems to model the true densities much better than the flexible naïve Bayes, and thus, the error due to the bias component is lower. On the other hand, flexible classifiers which model the correlation between predictors obtain quite similar bias components of the error. Moreover, the error due to the variance remains almost constant among flexible classifiers and, therefore, all the models seem to have a similar sensitivity to changes in the training set. In summary, among the flexible classifiers presented, in the absence of prior knowledge about the true density underlying the data, it is generally advisable to use the novel flexible tree-augmented naïve Bayes for supervised classification.

One of the most important disadvantages of the kernel density estimation is related with the computational cost that is required to evaluate an instance. Our main future work line consists of relaxing the strong computational cost required. We have considered approximating the kernel-based estimation by means of semi-parametric approaches, such as, mixture of truncated exponentials [Moral et al. (2002); Romero et al. (2006)], spline approximation [de Boor (2001); Gurwicz and Lerner (2004, 2005)] or Gaussian mixture models [McLachlan and Peel (2000)]. The Gaussian mixture models can be learned by means of the following approaches: EM algorithm [Bilmes (1997); McLachlan and Peel (2000)], projection pursuit algorithm [Aladjem (2002, 2005)], iterative pairwise replacement algorithm [Scott and Szewczyk (2001)] or M-Kernel merging procedure [Zhou et al. (2003)]. These semi-parametric approaches considerably reduce the computational cost: from a number of operations related with the number of training cases to a fixed one. Both approaches will allow us to design and implement other search techniques in the space of possible structures: wrapper search [Kohavi et al. (1997)] guided by the accuracy, or using metaheuristics such as genetic algorithms [Goldberg (1989)] or estimation of distribution algorithms [Larrañaga and Lozano (2002); Lozano et al. (2006)].

Other future lines based on KBN include the experimental study of classifiers in mixed domains and the use of the paradigm for making probabilistic inference. We thought that the paradigm can be specially useful with training sets of short and medium size.

5

Information theory, classification error and tree-augmented structures

This chapter focuses on the relation that seems to exist between some measures of information theory and the classification error of classifiers based on Bayesian networks. We were initially interested in this relation because conditional mutual information and/or mutual information have been used to guide the structural learning of most of filter classifier induction algorithms based on Bayesian networks [Sahami (1996); Friedman et al. (1997)]. The initial motivation of this chapter consist of searching novel alternative scores to conditional mutual information for guiding the structural learning of the tilter classifier induction algorithms presented in this document.

It is known that information theory provides a set of measures which can be interpreted in terms of uncertainty. Intuitively, the uncertainty surrounding the class random variable and the classification error should be related: when the uncertainty of the class increases the classification error tends to increase. This chapter provides a set of empirical evidences and theoretical arguments, which are used to show the connection between classification error and information theory. Besides, based on this knowledge, a discriminative structural learning algorithm of tree-augmented structures proposed by Pernkopf and Bilmes (2005) is theoretically analyzed.

The chapter is organized as follows. Section 5.1 presents the intuitions related to a set of quantities of the information theory, the likelihood and conditional likelihood with respect to the classification error. Then, some of the intuitions are illustrated by means of a set of experiments with tree-augmented naive Bayes structures. In Section 5.2 we analyze the discriminative structural learning of tree-augmented naive Bayes structures proposed by Pernkopf (2005). In the same section we provide a set of experiments which illustrate some intuitions related to the structural learning of classifiers based on BNs. This section includes an empirical comparison between both generative and discriminative learning algorithms of tree-augmented structures when the parameters are learned with a generative approach. Finally, Section 5.3 summarizes the main contributions introduced in this section and indicates the main future work lines.

5.1 Information theory, likelihood and classification error

Let us consider a supervised classification domain defined over the random variables C and $\boldsymbol{X}$, where C is a discrete class random variable with r states, $\{c^1, ..., c^r\}$, and $\boldsymbol{X} = (X_1, ..., X_n)$ are the predictor random variables. In supervised classification we are interested in predicting the state of C given a value of the predictor random variables $\boldsymbol{X} = \boldsymbol{x}$. This task is usually achieved learning a classifier ϕ, that is, a classification rule that assigns class labels to the different values of $\boldsymbol{X}$. The reader may consult Section 1.4 for further details on supervised classification.

As we noted in Section 1.5, the classifier which achieves the lower error is the Bayes classifier, $\phi_B(\boldsymbol{x}) \equiv \arg_c \max p(c|\boldsymbol{x}) \equiv \arg_c \max \rho(\boldsymbol{x}, c)$. Note that in this section we are considering only 0-1 loss functions (see Equation 1.19). The classification error of the Bayes classifier, ϵ_B, is given by Equation 1.15 using the Bayes classifier $\phi_B(\boldsymbol{x})$.

However, in most situations the *a posteriori* distribution of the class given the predictors, $p(c|\boldsymbol{x})$, is unknown and it must be estimated from a training set $\mathcal{S} = \{(\boldsymbol{x}^{(1)}, c^{(1)}), ..., (\boldsymbol{x}^{(N)}, c^{(N)})\}$. This chapter is focused on classifiers $\phi(\boldsymbol{x}) = \arg_c \max \rho_\phi(\boldsymbol{x}, c)$ learned from data $\mathcal{S}$, which are based on Bayesian networks (BNs) for $(\boldsymbol{X}, C)$. These classifiers approach the joint generalized probability function $\rho(\boldsymbol{x}, c)$, by means of the generalized probability function $\rho_\phi(\boldsymbol{x}, c)$ provided by the BN. They classify using the rule $\phi(\boldsymbol{x}) = \arg_c \max \rho(\boldsymbol{x}, c)$. Note that the classification rule $\phi(\boldsymbol{x}) = \arg_c \max \rho_\phi(\boldsymbol{x}, c)$ is equivalent to the rule $\arg_c \max p_\phi(c|\boldsymbol{x})$, where $p_\phi(c|\boldsymbol{x})$ can be obtained from $\rho_\phi(\boldsymbol{x}, c)$. When the learning is focused on modeling $\rho(\boldsymbol{x}, c)$, it is said to be generative or informative learning. Alternatively, when the learning is focused on obtaining a good estimation of $p(c|\boldsymbol{x})$, it is said to be conditional [Jebara (2004)] or discriminative learning [Pernkopf and Bilmes (2004); Santafé et al. (2005)] (see Section 1.6 for further details on generative, conditional and discriminative learning). In the remainder of this chapter we will use the term discriminative as proposed by [Pernkopf and Bilmes (2004); Santafé et al. (2005)]. Discriminative learning is justified from the point of view of the Occam's razor principle: *"It is futile to do with more what can be done with fewer"*. Following a similar idea Vapnik (1998) says *"...one should solve the (classification) problem directly and never solve a more general problem as an intermediate step (such as modeling $\rho(\boldsymbol{x}, c)$)"*. Indeed, generative learning solves a more general problem than discriminative learning because it tries to model $\rho(\boldsymbol{x}, c)$ instead of $p(c|\boldsymbol{x})$.

The BN associated to the classifier ϕ is defined by the pair $(\boldsymbol{s}, \boldsymbol{\Theta_s})$, where $\boldsymbol{s}$ is the structure of the network and $\boldsymbol{\Theta_s}$ is the set of its parameters (see Section 2.4 for further details on BNs). The BN associated to ϕ is usually learned in two steps: the structural learning and the parametric learning. The structural learning defines the qualitative part of the BN for $(\boldsymbol{X}, C)$, that is, the structure $\boldsymbol{s}$. The structure can be interpreted as a set of conditional independence assertions between the implied random variables $(\boldsymbol{X}, C)$ (see

Section 2.2.1 for further details on the conditional independence assertions codified by a BN). In other words, the structural learning defines the factorization of $\rho_\phi(\boldsymbol{x}, c) = \rho(\boldsymbol{x}, c; \boldsymbol{s}, \boldsymbol{\Theta_s}) = p(c; \Theta_c) \prod_{i=1}^{n} \rho(x_i | \boldsymbol{pa}(\boldsymbol{s})_i; \Theta_i)$, where the parameters $\boldsymbol{\Theta} = (\boldsymbol{\Theta}_1, ..., \boldsymbol{\Theta}_n, \boldsymbol{\Theta}_c)$ are not learned. This chapter is focused on structural learning of BN based classifiers. On the other hand, the parametric learning defines the quantitative part of the model, that is, the parameters $\boldsymbol{\Theta_s} = (\boldsymbol{\Theta}_1, ..., \boldsymbol{\Theta}_n, \boldsymbol{\Theta}_c)$ which determine the local probability functions $\rho(x_i | \boldsymbol{pa}_i, c; \boldsymbol{\Theta}_i)$ and $p(c; \boldsymbol{\Theta}_c)$. For further details on BNs in general, and BMNs in particular, the reader may consult Chapter 2.

This section is focused on the structural learning of tree-augmented naive Bayes (TAN) family of structures which are clearly biased towards supervised classification (see Section 2.5.2 for the intuitions behind the augmented-naive Bayes family of structures). The parametric learning is performed using a generative procedure, such as the maximum likelihood.

5.1.1 Intuitions behind information theory

The entropy of the class C, $H(C)$, can be interpreted as a measure of the uncertainty surrounding the class variable C (see Section 1.10 for further interpretations of the entropy). It is intuitive to think that both the uncertainty of the class C and the capability to predict C a priori should be somehow related [Cover and Thomas (1991)]. The entropy $H(C)$ depends on the distribution $p(c)$ and it reaches the maximum when the class labels are equiprobable, that is, when the error of the classification rule $\arg_c \max p(c)$ is maximum. Alternatively, the entropy is minimum (zero) whenever a class label c exists for which $p(c) = 1$ and, therefore, the error of the classification rule $\arg_c \max p(c)$ is also minimum (zero). When the uncertainty surrounding the class variable increases, the error of the classification rule $\arg_c \max p(c)$ seems to increase. Thus, the error and the uncertainty surrounding C should be somehow related and the reduction of the uncertainty of C tends to be beneficial for supervised classification.

The conditional entropy of C given the predictors $\boldsymbol{X}$, $H(C|\boldsymbol{X})$, measures the uncertainty surrounding the class variable when the predictors are known. Following the intuition behind $H(C)$, it could be thought that the conditional entropy $H(C|\boldsymbol{X})$ is given some information about the error associated to the classification rule $\arg_c \max p(c|\boldsymbol{x})$. Moreover, the mutual information between the class and the predictors, $I(\boldsymbol{X}; C) = H(C) - H(C|\boldsymbol{X})$, can be interpreted as the reduction in the uncertainty of C when the predictors $\boldsymbol{X}$ are known (see Section 1.10 for further interpretations). Intuitively, the mutual information $I(\boldsymbol{X}; C)$ tends to be inversely related to the error of the rule $\arg_c \max p(c|\boldsymbol{x})$.

The classifier ϕ defines a factorization $\rho_\phi(\boldsymbol{x}, c)$ through a BN defined by $(\boldsymbol{s}, \boldsymbol{\Theta_s})$. It is possible to define the entropy of the random variables $(\boldsymbol{X}, C)$ assuming that they are distributed according to $\rho_\phi(\boldsymbol{x}, c)$:

$$H_\phi(\boldsymbol{X}, C) = -E_{\hat{\rho}(\boldsymbol{x},c)}[\log(\rho_\phi(\boldsymbol{x}, c))] = -\frac{1}{N}\sum_{i=1}^{N}\log(\rho_\phi(\boldsymbol{x}^{(i)}, c^{(i)})) \tag{5.1}$$

where $\hat{\rho}(\boldsymbol{x}, c) = 1/N$ iff $(\boldsymbol{x}, c) \in \mathcal{S}$, and 0 in the other case, is the empirical distribution given by the training set $\mathcal{S} = \{(\boldsymbol{x}^{(1)}, c^{(1)}), ..., (\boldsymbol{x}^{(N)}, c^{(N)})\}$. Note that in order to indicate explicitly the generalized probability function used to compute the expectance over $(\boldsymbol{X}, C)$, we have changed the notation $E_{(\boldsymbol{x},c)}$ by $E_{\rho(\boldsymbol{x},c)}$. We call ϕ-entropy of $(\boldsymbol{X}, C)$ to $H_\phi(\boldsymbol{X}, C)$ and it depends on both the structure of the model, $\boldsymbol{s}$, and the parameters, $\boldsymbol{\Theta}$. However, we focus our attention on the structural learning and the set of conditional independence statements inferred from the structure of ϕ, $\boldsymbol{s}$. We have removed from the study the parametric learning. From here on we will consider only generative approaches to the parametric learning: the parameters of BMN and CGN based classifiers are learned by maximum likelihood (see Chapters 2 and 3), and the parameters of KBN are learned with the differential scaled plus normal rule approach (see Chapter 4). Therefore, through this chapter, the value of ϕ-entropy only depends on the structure of the model, $\boldsymbol{s}$. The experimental and theoretical results should be interpreted in the light of a generative learning of the parameters.

Analogously to the ϕ-entropy, $H_\phi(\boldsymbol{X}, C)$, we can define other measures of the information theory such us the ϕ-conditional entropy of C given $\boldsymbol{X}$:

$$H_\phi(C|\boldsymbol{X}) = E_{\hat{\rho}(\boldsymbol{x},c)}[p_\phi(c|\boldsymbol{x})] \tag{5.2}$$

where $p_\phi(c|\boldsymbol{x})$ is obtained from $\rho_\phi(\boldsymbol{x}, c)$ by conditioning on $\boldsymbol{X}$. We are interested in finding relations between these quantities based on the generalized probability function $\rho_\phi(\boldsymbol{x}, c)$ and the classification error associated to the classification rule $\phi(\boldsymbol{x}) = \arg_c \max \rho_\phi(\boldsymbol{x}, c)$ taking into account the conditional independencies included in the structure $\boldsymbol{s}$ of the model ϕ. Intuitively, $H_\phi(C|\boldsymbol{X})$ is the measure we are interested in because, assuming the distribution $\rho_\phi(\boldsymbol{x}, c)$, it measures the uncertainty surrounding the class variable C given the predictors $\boldsymbol{X}$. Thus, when learning the structure of the classifier $\phi(\boldsymbol{x})$ we could try to minimize the ϕ-conditional entropy, $H_\phi(C|\boldsymbol{X})$. However, the use of ϕ-mutual information between C and $\boldsymbol{X}$ is useful. The ϕ-mutual information is given by:

$$I_\phi(\boldsymbol{X}; C) = E_{\hat{\rho}(\boldsymbol{x},c)}[\log \frac{\rho_\phi(\boldsymbol{x}, c)}{p_\phi(c)\rho_\phi(\boldsymbol{x})}] \tag{5.3}$$

It can be interpreted as the reduction of the uncertainty of the class C when the predictors $\boldsymbol{X}$ are known.

5.1.2 Intuitions behind likelihood and conditional likelihood

Some generative structural learning algorithms make use of the likelihood of the model ϕ given the data $\mathcal{S}$ [Friedman et al. (1997); Jebara (2004)]:

$$L(\phi|\mathcal{S}) = \prod_{i=1}^{N} \rho_\phi(\boldsymbol{x}^{(i)}, c^{(i)})$$

The likelihood of the model given the data can be interpreted as the probability of the data set $\mathcal{S}$ given the model ϕ, $L(\phi|\mathcal{S}) \equiv p(\mathcal{S}|\phi)$. The probability $p(\mathcal{S}|\phi)$ can be understood as the capability of ϕ for explaining the random variables $(\boldsymbol{X}, C)$ in the data set $\mathcal{S}$.

In order to facilitate the operations, the log likelihood is commonly used:

$$LL(\phi|\mathcal{S}) = \sum_{i=1}^{N} \log(p_\phi(c^{(i)}, \boldsymbol{x}^{(i)}))$$

It should be noted that $\arg_\phi \max L(\phi|\mathcal{S}) \equiv \arg_\phi \max LL(\phi|\mathcal{S})$.

Some of the most common criteria based on the likelihood are: likelihood maximization [Friedman et al. (1997)], minimum description length [Lam and Bacchus (1994); Rissanen (1978)] and Bayesian metrics [Cooper and Herskovits (1992); Heckerman et al. (1995)]. These approximations can be considered generative because they try to maximize the likelihood $LL(\phi|\mathcal{S})$ under some constraints. Therefore, they try to accurately model $p(\boldsymbol{x}, c)$ (see Proposition 5.1, Proposition 5.2 and Corollary 5.1). However, the likelihood shows some problems for the supervised classification tasks [Friedman et al. (1997); Jebara (2004)]:

$$LL(\phi|\mathcal{S}) = \sum_{i=1}^{N} \log(p_\phi(c^{(i)}|\boldsymbol{x}^{(i)})) + \log(\rho_\phi(\boldsymbol{x}^{(i)})) \tag{5.4}$$

The first term is directly related with supervised classification and it is usually referred to as conditional likelihood. The second one, named marginal log likelihood, is only relevant for modeling the (in)conditional dependence relationships among predictive variables $\boldsymbol{X}$ but it is irrelevant for classification purposes [Jebara (2004); Friedman et al. (1997)]. Moreover, when the number of variables, n, grows $p(\boldsymbol{x})$ tends to take lower values and, on the other hand, $p(c|\boldsymbol{x})$ tends to take values of the same magnitude. Therefore, the marginal log likelihood term becomes more important than conditional likelihood as n grows [Friedman et al. (1997)] and this is clearly detrimental for supervised classification. Besides, focusing on the structural learning process, maximum likelihood increases monotonically with respect to the conditional dependencies modeled. In other words, maximum likelihood criteria tends to induct complete structures. Nevertheless, the maximization of the log likelihood score does not necessarily obtain classifiers with lower error rates.

On the other hand, maximum conditional likelihood criterion could be more appropriate than maximum likelihood for supervised classification problems [Jebara (2004); Grossman and Domingos (2004)]:

$$CL(\phi|\mathcal{S}) = \prod_{i=1}^{N} p_\phi(c^{(i)}|\boldsymbol{x}^{(i)})$$

The conditional likelihood is given by the conditional probability of the class variable given the predictors, $p(c|\boldsymbol{x})$ [Jebara (2004)]. It treats the class variable C in the same special way as supervised classification. Analogously to the likelihood, the conditional likelihood can be interpreted as the likelihood of the random variable C when the predictor variables $\boldsymbol{X}$ are known given the data set $\mathcal{S}$. The structural learning process guided by the conditional likelihood can be considered discriminative because it tries to find a set of arcs which accurately model the conditional distribution $p(c|\boldsymbol{x})$ rather than $\rho(\boldsymbol{x}, c)$ (see Proposition 5.3, Proposition 5.4 and Corollary 5.2). The logarithm of $CL(\phi|\mathcal{S})$ is known as conditional log likelihood, $CLL(\phi|\mathcal{S})$, and it corresponds to the first term of Equation 5.4. The maximization of conditional log likelihood is directly related with the minimization of the ϕ-conditional entropy $H_\phi(C|\boldsymbol{X})$ (see Proposition 5.3). From the point of view of the information theory, minimizing the conditional entropy is equivalent to reducing the uncertainty of the class variable when the predictors are known. This is clearly beneficial for classification.

Usually, the structural learning is constrained to a subfamily of structures, e.g. TAN structures. In this case, two situations are possible. If the true distribution of the domain, $\rho(\boldsymbol{x}, c)$, can be represented by the selected subfamily of structures, e.g. TAN structures, generative learning may learn classifiers with a competitive performance. On the other hand, if $\rho(\boldsymbol{x}, c)$ can not be represented by the selected subfamily of structures, generative structural learning should perform worse than discriminative structural learning. The idea is generalizable to both parametric and structural learning of BNs (see [Santafé (2007)] for theoretical results).

Following, we present a set of theoretical results related to the intuitions provided in the previous sections.

5.1.3 Theoretical results

This subsection provides a set of theoretical results which relate the previously introduced measures. Besides, we show that these quantities are related to Kullback-Leibler divergence [Cover and Thomas (1991)]. We have presented the results only for discrete random variables for the sake of simplicity and clarity but, these results are generalizable to mixed domains. Some of the provided results are adapted from Friedman et al. (1997).

Proposition 5.1 *Given a training set* $\mathcal{S} = \{(\boldsymbol{x}^{(1)}, c^{(1)}), ..., (\boldsymbol{x}^{(N)}, c^{(N)})\}$ *and a classifier* ϕ *learned from* $\mathcal{S}$ *given by* $(\boldsymbol{s}, \boldsymbol{\Theta_s})$*, the* ϕ*-entropy of* $(\boldsymbol{X}, C)$ *verifies that*

$$H_\phi(\boldsymbol{X}, C) = -\frac{1}{N} LL(\phi|\mathcal{S}) \tag{5.5}$$

Proof.

$$\begin{aligned} H_\phi(\boldsymbol{X}, C) &= -E_{\hat{p}(\boldsymbol{x},c)}[\log p_\phi(\boldsymbol{x}, c)] \\ &= -\sum_{\boldsymbol{x},c} \hat{p}(\boldsymbol{x}, c) \log p_\phi(\boldsymbol{x}, c) \\ &= -\frac{1}{N} \sum_{i=1}^{N} \log p_\phi(\boldsymbol{x}^{(i)}, c^{(i)}) \\ &= -\frac{1}{N} LL(\phi|\mathcal{S}) \end{aligned}$$

. ■

Thus, given a training set $\mathcal{S}$, minimizing $H_\phi(\boldsymbol{X}, C)$ is equivalent to maximizing $LL(\phi|\mathcal{S})$.

Proposition 5.2 *Given a training set* $\mathcal{S} = \{(\boldsymbol{x}^{(1)}, c^{(1)}), ..., (\boldsymbol{x}^{(N)}, c^{(N)})\}$ *and a classifier* ϕ *learned from* $\mathcal{S}$ *given by* $(\boldsymbol{s}, \boldsymbol{\Theta_s})$*, the Kullback-Leibler divergence* $D_{KL}(\hat{p}(\boldsymbol{x}, c)||p_\phi(\boldsymbol{x}, c))$ *verifies that*

$$D_{KL}(\hat{p}(\boldsymbol{x}, c)||p_\phi(\boldsymbol{x}, c)) \propto H_\phi(\boldsymbol{X}, C) \tag{5.6}$$

Proof.

$$\begin{aligned} D_{KL}(\hat{p}(\boldsymbol{x}, c)||p_\phi(\boldsymbol{x}, c)) &= E_{\hat{p}(\boldsymbol{x},c)}[\log \frac{\hat{p}(\boldsymbol{x}, c)}{p_\phi(\boldsymbol{x}, c)}] \\ &= E_{\hat{p}(\boldsymbol{x},c)}[\log \hat{p}(\boldsymbol{x}, c)] - E_{\hat{p}(\boldsymbol{x},c)}[\log p_\phi(\boldsymbol{x}, c)] \\ &= k + H_\phi(\boldsymbol{X}, C) \end{aligned}$$

where k is a constant given $\mathcal{S}$. ■

Corollary 5.1 *Given a training set* $\mathcal{S} = \{(\boldsymbol{x}^{(1)}, c^{(1)}), ..., (\boldsymbol{x}^{(N)}, c^{(N)})\}$ *and a classifier* ϕ *learned from* $\mathcal{S}$ *given by* $(\boldsymbol{s}, \boldsymbol{\Theta_s})$*, the Kullback-Leibler divergence* $D_{KL}(\hat{p}(\boldsymbol{x}, c)||p_\phi(\boldsymbol{x}, c))$ *verifies that*

$$D_{KL}(\hat{p}(\boldsymbol{x}, c)||p_\phi(\boldsymbol{x}, c)) \propto LL(\phi|\mathcal{S}) \tag{5.7}$$

The proof is straightforward in the light of Proposition 5.1 and Theorem 5.2.

Thus, maximizing the log likelihood, or equivalently minimizing the ϕ-entropy, is equivalent to minimizing the Kullback-Leibler divergence between the empirical and the modeled distributions, $\hat{p}(\boldsymbol{x}, c)$ and $p_\phi(\boldsymbol{x}, c)$.

Proposition 5.3 *Given a training set* $\mathcal{S} = \{(\boldsymbol{x}^{(1)}, c^{(1)}), ..., (\boldsymbol{x}^{(N)}, c^{(N)})\}$ *and a classifier* ϕ *learned from* $\mathcal{S}$ *given by* $(\boldsymbol{s}, \boldsymbol{\Theta_s})$*, the* ϕ*-conditional entropy verifies that*

$$H_\phi(C|\boldsymbol{X}) = -\frac{1}{N} CLL(\phi|\mathcal{S}) \tag{5.8}$$

Proof.

$$\begin{aligned} H_\phi(C|\boldsymbol{X}) &= -E_{\hat{p}(\boldsymbol{x},c)}[\log p_\phi(c|\boldsymbol{x})] \\ &= -\sum_{\boldsymbol{x},c} \hat{p}(\boldsymbol{x},c) \log p_\phi(c|\boldsymbol{x}) \\ &= -\frac{1}{N}\sum_{i=1}^{N} \log p_\phi(c^{(i)}|\boldsymbol{x}^{(i)}) \\ &= -\frac{1}{N} CLL(\phi|\mathcal{S}) \end{aligned}$$

. ■

Thus, given a training set $\mathcal{S}$, minimizing $H_\phi(C|\boldsymbol{X})$ is equivalent to maximizing $CLL(\phi|\mathcal{S})$.

Proposition 5.4 *Given a training set $\mathcal{S} = \{(\boldsymbol{x}^{(1)}, c^{(1)}), ..., (\boldsymbol{x}^{(N)}, c^{(N)})\}$ and a classifier ϕ learned from $\mathcal{S}$ given by $(\boldsymbol{s}, \boldsymbol{\Theta}_{\boldsymbol{s}})$, the expectance of the Kullback-Leibler divergence $E_{\hat{p}(\boldsymbol{x})}[D_{KL}(\hat{p}(c|\boldsymbol{x})||p_\phi(c|\boldsymbol{x}))]$ verifies that*

$$E_{\hat{p}(\boldsymbol{x})}[D_{KL}(\hat{p}(c|\boldsymbol{x})||p_\phi(c|\boldsymbol{x}))] \propto H_\phi(C|\boldsymbol{X}) \tag{5.9}$$

where $\hat{p}(\boldsymbol{x})$ has been obtained by marginalizing $\hat{p}(\boldsymbol{x}, c)$ over C.

Proof.

$$\begin{aligned} E_{\hat{p}(\boldsymbol{x})}[D_{KL}(\hat{p}(c|\boldsymbol{x})||p_\phi(c|\boldsymbol{x}))] &= E_{\hat{p}(\boldsymbol{x})}[E_{\hat{p}(c|\boldsymbol{x})}[\log \frac{\hat{p}(c|\boldsymbol{x})}{p_\phi(c|\boldsymbol{x})}]] \\ &= E_{\hat{p}(\boldsymbol{x},c)}[\log \frac{\hat{p}(c|\boldsymbol{x})}{p_\phi(c|\boldsymbol{x})}] \\ &= E_{\hat{p}(\boldsymbol{x},c)}[\log \hat{p}(c|\boldsymbol{x})] - E_{\hat{p}(\boldsymbol{x},c)}[\log p_\phi(c|\boldsymbol{x})] \\ &= k + H_\phi(C|\boldsymbol{X}) \end{aligned}$$

where k is a constant given $\mathcal{S}$. ■

Corollary 5.2 *Given a training set $\mathcal{S} = \{(\boldsymbol{x}^{(1)}, c^{(1)}), ..., (\boldsymbol{x}^{(N)}, c^{(N)})\}$ and a classifier ϕ learned from $\mathcal{S}$ given by $(\boldsymbol{s}, \boldsymbol{\Theta}_{\boldsymbol{s}})$, the expected Kullback-Leibler divergence $E_{\hat{p}(\boldsymbol{x})}[D_{KL}(\hat{p}(\boldsymbol{x}, c)||p_\phi(\boldsymbol{x}, c))]$ verifies that*

$$E_{\hat{p}(\boldsymbol{x})}[D_{KL}(\hat{p}(c|\boldsymbol{x})||p_\phi(c|\boldsymbol{x}))] \propto CLL(\phi|\mathcal{S}) \tag{5.10}$$

The proof is straightforward in the light of Proposition 5.3 and Proposition 5.4.

Thus, maximizing the conditional log likelihood, or equivalently minimizing the conditional entropy, is equivalent to minimizing the expected Kullback-Leibler divergence between the empirical and the modeled conditional distributions, $\hat{p}(c|\boldsymbol{x})$ and $p_\phi(c|\boldsymbol{x})$.

Following, in order to illustrate some of the presented theoretical results, we perform a set of experiments which show the benefits of CLL for guiding the structural learning of classifiers based on TAN structures when the parameters are learned in a generative way.

5.1.4 Experimental results with TAN structures

This section provides empirical evidence in order to illustrate the benefits of using conditional log likelihood instead of log likelihood in the structural learning process. We experimentally show that conditional log likelihood is significantly better than log likelihood for guiding the structural learning of TAN structures with classification purposes. We estimate the Spearman correlation coefficient between the classification error and both the likelihood and the conditional likelihood of TAN structures. This estimation is performed in different domains using Algorithm 11.

The experiments have been performed in both artificial and real-world [Asuncion and Newman (2007)] continuous domains. It must be noted that the experiments of this section have been performed with *kernel based Bayesian networks* (KBN) [Pérez et al. (2009)], a kind of BN which uses kernel density estimation [Silverman (1986); Wand and Jones (1995); Simonoff (1996)] in order to model the densities of the continuous predictors conditioned to each class value. The parameters of the classifiers based on KBN are learned with differential scaled plus normal rule option, as described in Chapter 4. Note that this approach can be understood as a generative learning of parameters. The reader may consult Chapter 4 for further details on KBN paradigm.

Algorithm 11: **Estimation of the Spearman correlation coefficient.**

Inputs

The number of different TAN structures (l), the data $\mathcal{S}$.

Outputs

Spearman correlation coefficients: $s(LL(\phi|\mathcal{S}); \epsilon_\phi)$ and $s(CLL(\phi|\mathcal{S}); \epsilon_\phi)$.

Procedure

Create l different TAN structures $\boldsymbol{s}$ in a deterministic way.

Estimate for each TAN structure $\boldsymbol{s}$: classification error ϵ_ϕ, log likelihood $LL(\phi|\mathcal{S})$, and conditional log likelihood $CLL(\phi|\mathcal{S})$.

Using all the estimated measures calculate the following Spearman correlation coefficients: $s(LL(\phi|\mathcal{S}); \epsilon_\phi)$ and $s(CLL(\phi|\mathcal{S}); \epsilon_\phi)$.

A. Artificial data sets

This section summarizes the results obtained in a set of artificial domains generated using the procedure of Figure 12. Intuitively, the procedure starts generating a graph. Then, a uniform distribution is sampled. The factorization of the joint probability function, given by the graph, is estimated by means of kernels using the sampled data. The result of the procedure is a factorization of $\rho(\boldsymbol{x}, c)$ based on KBN. Most experiments with artificial data are restricted to certain distribution assumptions, but real-world problems do not necessarily

follow a parametric distribution in general. Besides, the underlaying distribution usually is unknown in real-world domains. On the other hand, there is no way of generating all possible kinds of distributions as a consequence of the *No Free Lunch Theorem* [Bishop (2006)]. However, a broad range of distributions can be represented using KBNs. The generation of artificial domains based on KBNs ensure a wide range of density shapes. The conclusions obtained from the data sets sampled from KBNs can be considered general enough for our purposes.

Algorithm 12: **Procedure used to generate the artificial data sets.**

1 **Inputs**

2 The number of classes (r), the a priori distribution of C ($p(c)$), the number of predictor variables (n), the maximum number of predictor parents for each predictor (k), the number of kernels (m) which models each density $f(x_i|\boldsymbol{pa}_i, C = c)$, and the range of the variables $[0...max]$.

3 **Outputs**

4 A probability distribution $\rho(\boldsymbol{x}, c)$ based on the KBN paradigm.

5 **Procedure**

6 Generate randomly the graph $\boldsymbol{s}$ of the KBN structure taking into account k and n.

7 Determine randomly, $rand(1, 1+2^0, 1+2^1, ..., 1+2^{\lfloor log(m-1) \rfloor}, m)$, the number of instances to be generated per each class $C = c$.

8 Generate randomly the instances $\mathcal{S}$ determined for each class c, which will be used in order to model each kernel based density functions $f(x_i|x_{pa_i}, c)$, associated to $\boldsymbol{s}$.

9 Learn from $\mathcal{S}$ the KBN based classifier associated to $\boldsymbol{s}$ and return it.

Each cell of Table 5.1 represents a Spearman non-linear correlation coefficient. The Spearman correlation coefficients have been obtained as follows:

- Repeat 50 times:
  - Generate a distribution $\rho(\boldsymbol{x}, c)$ based on the KBN paradigm following the procedure described in Figure 12 with parameters $r = rand(2, 3, 4)$, $p(c)$ at random, $n = \{4, 8, 16, 32\}$, $k = rand(1, ..., n - 1)$, $m = 2048$, and $max = 10$.
  - Sample from $\rho(\boldsymbol{x}, c)$ a test set $\mathcal{S}_T$ of size 1000.
  - Sample from $\rho(\boldsymbol{x}, c)$ three training sets $\mathcal{S}$ of size $N = \{20, 400, 8000\}$.
  - Repeat for each training set $\mathcal{S}$ generated:
    Obtain the Spearman correlation coefficients of this domain following the procedure described in Figure 11 with $l = 50$. The estimation of step 8 is performed using a holdout scheme ($\mathcal{S}$ for training and $\mathcal{S}_T$ for testing).
- Compute the mean value of the Spearman correlation coefficients across the 50 domains generated.

- Perform a Wilcoxon paired test at $\alpha = 1\%$ significance level between the Spearman coefficients $s(LL(\phi|\mathcal{S}); \epsilon_\phi)$ and $s(CLL(\phi|\mathcal{S}); \epsilon_\phi)$ following the procedure described in Demšar (2006).

		Number of variables			
Score	N	4	8	16	32
LL	20	-58.1	-56.0	-52.3	-46.1
CLL	20	∘ -71.0	∘ -67.2	∘ -64.4	∘ -60.5
LL	400	-58.1	-56.0	-52.3	-46.1
CLL	400	∘ -71.0	∘ -67.2	∘ -64.4	∘ -60.5
LL	8000	-64.4	-60.0	-55.0	-55.1
CLL	8000	∘ -82.3	∘ -80.5	∘ -79.3	∘ -80.6

Table 5.1. Mean of the Spearman correlation coefficients between the error and both the LL and CLL scores across 50 domains. The artificial domains have been generated with the procedure of Fig 12 with parameters $r \in rand\{2, 3, 4\}$, $p(c)$ at random (ensuring a minimum of $p(c) = 0.1$ per each class), $n \in \{4, 8, 16, 32\}$, $k \in rand\{1, ..., n-1\}$, $m = 2048$, and $max = 10$.

Table 5.1 shows that CLL obtains statistically significant better non-linear correlation coefficients for a different number of variables and different training set sizes. We can conclude that, in the set of generated domains, CLL is better than LL for guiding the structural learning of TAN structures when the local factors are learned in a generative way.

B. UCI data sets

The main characteristics of the real-world continuous domains selected from the UCI repository [Asuncion and Newman (2007)] are summarized in Figure 5.2. In order to obtain robust conclusions, we have selected data sets with more than 5 variables for the experiment. Predictor variables with a variance less than 10^{-4} have been removed to avoid precision problems.

The Spearman correlation coefficients of Table 5.3 have been obtained, at each data set, using the procedure of Figure 11 with $l = 50$. The estimation of step 3 is performed using a stratified 10-fold cross-validation for data sets of less than 3000 samples and stratified 3-fold cross-validation in other cases. In order to check the significance of the differences between the Spearman coefficients $s(LL(\phi|\mathcal{S}); \epsilon_\phi)$ and $s(CLL(\phi|\mathcal{S}); \epsilon_\phi)$, a Wilcoxon paired test at $\alpha = 1\%$ significance level has been performed across all the included UCI data sets following the procedure described in Demšar (2006).

The results show that conditional log likelihood is clearly better to guide the structural learning of TAN structures for classification in the selected data sets: it obtains statistically significant results at $\alpha = 1\%$ across all the data sets. Only in two data sets, Liver and Yeast, log likelihood obtains slightly better non-linear correlation coefficients.

Label	r	n	N	Name of the data set
1 Balance	3	4	625	Balance Scale Weight & Distance
2 Block	5	10	5474	Page Block's Classification
3 Haberman	2	3	307	Haberman's Survival
4 Image	7	18	2310	Image Segmentation
5 Ionosphere	2	34	351	Johns Hopkins University Ionosphere
6 Iris	3	4	150	Iris Plant
7 Letter	26	16	20000	Letter Image Recognition
8 Liver	2	6	345	Bupa Liver Disorders
9 Pima	2	8	768	Pima Indians Diabetes
10 Satellite	6	36	6435	Landsat Satellite
11 Sonar	2	60	208	Sonar, Mines VS Rocks
12 Thyroid	3	5	215	Thyroid Disease Records
13 Vehicle	4	19	846	Vehicle Silhouetes
14 Vowel	11	10	990	Vowel Recognition
15 Waveform	3	21	5000	Waveform Data Generation
16 Wine	3	13	179	Wine Recognition
17 Yeast	10	9	1484	Yeast

Table 5.2. Main characteristics of the data sets selected from UCI [Asuncion and Newman (2007)]: the number of different values of the class variable (r), the number of predictor variables (n), and the number of instances (N).

	Correlation	
Label	LL	∘CLL
Block	-0.51	-0.65
Image	-0.12	-0.76
Ionosphere	-0.60	-0.61
Letter	-0.84	-0.96
Liver	-0.48	-0.40
Pima	-0.18	-0.28
Satellite	-0.53	-0.83
Sonar	-0.24	-0.27
Vehicle	-0.55	-0.67
Vowel	-0.81	-0.93
Waveform	-0.78	-0.85
Wine	-0.19	-0.42
Yeast	-0.04	0.24

Table 5.3. Spearman non-linear correlation coefficients between the error and both the LL and CLL scores for 50 different TAN structures in the selected data sets.

5.2 Structural learning of classifiers based on TAN structures

The version of the well known classifier induction algorithm proposed by Friedman et al. (1997), which learns tree-augmented naive Bayes classifier structures in a discriminative way, was presented to the Machine Learning community by Pernkopf and Bilmes (2005). This section analyzes the promising discriminative structural learning algorithm, highlighting its theoretical properties. We demonstrate that, the discriminative algorithm does not necessarily find a TAN structure that maximizes the conditional likelihood. The experimentation has been performed with both artificial and real-world continuous data sets for the following paradigms based on BNs: Bayesian multinomial network (BMNs) plus discretization, conditional Gaussian networks (CGNs) and kernel based Bayesian networks (KBNs). When the parameters are learned in a generative way, the analyzed algorithm does not seem to outperform the algorithm of Friedman et al. (1997). For further details on BMN, CGN and KBN paradigms the reader may consult Section 2, Section 3 and Section 4, respectively.

5.2.1 Some intuitions for generative classifiers

This section provides a set of experiments which intuitively illustrates the relation between the uncertainty surrounding the class variable and the classification error. The explanations are based on the following random framework: on two continuous predictor random variables, X_1 and X_2, and a class random variable, C, with r states, $\{c^1, ..., c^r\}$. Using these random variables, two classification models could be proposed based on TAN structures, ϕ_{uni} and ϕ_{mul}. ϕ_{uni} corresponds to a classifier with a naive Bayes structure and ϕ_{mul} corresponds to a classifier with a complete structure. Note that in domains with two random variables, a complete TAN structure is equivalent to a complete graph. Both models, ϕ_{uni} and ϕ_{mul} are based on the factorizations $\rho_{uni}(x_1, x_2, c) = p(c)f(x_1|c)f(x_2|c)$ and $\rho_{uni}(x_1, x_2, c) = p(c)f(x_1|c)f(x_2|x_1, c)$, respectively. We assume that the factors are obtained from $\rho(x_1, x_2, c)$ by conditioning and marginalizing over the appropriate variables. For example, $p(c)$ is obtained by marginalizing over X_1 and X_2, and it represents the true distribution of the class variable. However, in the experimental subsection the parameters will be learned from data $\mathcal{S} = \{(x_1^{(1)}, x_2^{(1)}, c^{(1)}), ..., (x_1^{(N)}, x_2^{(N)}, c^{(N)})\}$ in a generative way and the factors will be estimators of the true $p(c)$, $f(x_1|c)$, $f(x_2|c)$ and $f(x_2|x_1, c)$.

As we have introduced in Section 5.1, intuitively, the error of a classifier ϕ should be directly related with the ϕ-entropy, $H_\phi(C|\boldsymbol{X})$, which measures the uncertainty of the class when the predictor random variables are known. Based on the models, ϕ_{uni} and ϕ_{mul}, two conditional entropies are defined $H_{uni}(C|X_1, X_2)$ and $H_{mul}(C|X_1, X_2)$. Besides, as we have noted in Section 5.1 both quantities are given by:

$$H_{uni}(C|X_1, X_2) = H(C) - I_{uni}((X_1, X_2); C)$$
$$H_{mul}(C|X_1, X_2) = H(C) - I_{mul}((X_1, X_2); C)$$

where $I_{uni}(X_1, X_2; C) = I(C; X_1) + I(C; X_2)$, due to the conditional independence $CI(X_1; X_2|C)$, and $I_{mul}(X_1, X_2; C) = I(C; X_1) + I(C; X_2) - I(X_1; X_2; C)$. Since the distribution $p(c)$ is constant, the entropy $H(C)$ is constant and thus the quantities $H_{uni}(C|X_1, X_2)$ and $H_{mul}(C|X_1, X_2)$ are indirectly related with $I_{uni}((X_1, X_2); C)$ and $I_{mul}((X_1, X_2); C)$, respectively. In the next subsection we will illustrate that the errors associated to ϕ_{uni} and ϕ_{mul} (ϵ_{uni} and ϵ_{mul}) are directly related to $H_{uni}(C|X_1, X_2)$ and $H_{mul}(C|X_1, X_2)$, respectively. This is equivalent to saying that the errors ϵ_{uni} and ϵ_{mul} are indirectly related with the quantities $I_{uni}(X_1, X_2; C)$ and $I_{mul}(X_1, X_2; C)$ respectively. Therefore, in order to minimize the error in a certain domain defined over (X_1, X_2, C), it should be more advisable to classify using ϕ_{mul} instead of ϕ_{uni} as the quantity $I(X_1; X_2; C)$ increases. In the following subsection we try to answer the following questions: given the *a posteriori* probability $p_{mul}(c|x_1, x_2)$, what kind of relation exists between the uncertainty that surrounds the class variable and the classification errors ϵ_{mul} and ϵ_{uni}? Intuitively, the error should grow with the increase in the uncertainty associated to $p_{\phi}(c|x_1, x_2)$. When is more advisable to use $p_{mul}(c|x_1, x_2)$ instead of $p_{uni}(c|x_1, x_2)$ for classification? It can be hoped that a relevant measure is the difference between the information theory based measures related to ϵ_{uni} and ϵ_{mul}.

5.2.1.1 Experimental analysis

This section tries to find monotonically increasing/decreasing relations between different measures based on information theory and the errors ϵ_{mul}, ϵ_{uni} and $\epsilon_{dif} = \epsilon_{uni} - \epsilon_{mul}$, when the predictors are continuous. For this purpose, we randomly generate a set of continuous bivariate artificial domains for computing a set of measures from information theory and the errors introduced. The experimental results presented in this section have been taken from Pérez et al. (2006a).

A. Random generator of domains

The random generator of domains (*RGD*) function employed in the experimentation generates, in a random way, a set of artificial bivariate domains (see Algorithm 13). Then, in each domain, a set of measures of information theory and the exposed errors (ϵ_{mul}, ϵ_{uni} and ϵ_{dif}) are computed. Each generated domain is specified by a KBN given by the pseudocode shown in Algorithm 13. The factor $f(x_1, x_2|c)$ is based on kernel densities and it depends on the number of kernels to be used m, on the coordinates of each of those kernels $\{k_1, ..., k_m\}$, being $k_i = (x_1^{(i)}, x_2^{(i)})$, and on a unique smooth parameter $h^2 = m^{-\frac{1}{6}}$.

Algorithm 13: **The pseudocode of the RGD function.**

1 **Inputs**
2 The maximum number of kernels (m_{max}), number of classes (r), number of domains to be generated (d), the range of the variables X_1 and X_2 ($ranX_1$ and $ranX_2$) and the a priori distribution of the class ($p(c)$).
3 **Outputs**
4 The errors ϵ_{mul} and ϵ_{uni} and a set of measures from information theory for each domain generated.
5 **Procedure**
6 **Repeat** d **times**:
7 Set a number of kernel components $m_c \in \{1, ..., m_{max}\}$ for each density $f(x_1, x_2|c)$.
8 Compute a set of measures from information theory and the errors ϵ_{mul}, ϵ_{uni} and ϵ_{dif} for the current domain.
9 **End repeat**
10 Plot ϵ_{mul}, ϵ_{uni} and ϵ_{dif} versus measures from information theory.

B. Experimental results

In order to study the relation between information theory based measures and introduced errors, we generate a set of artificial domains with the same a priori distribution $p(c)$. Therefore, the entropy $H(C)$ is constant across them. Any real-world domain, with n predictors, can be seen as a set of $\binom{n}{2}$ bivariate domains fulfilling the constraint that $H(C)$ is constant. In this sense, we hope that the conclusions of this study can be applied to certain real-world domains. Figure 5.1 has been obtained using the RGD function with parameters $m_{max} = 40$, $r = 4$, $d = 10000$, $ranX_1 = [0, 15]$, $ranX_2 = [0, 15]$ and $p(C = c) = 0.1c$ for $c \in \{1, 2, 3, 4\}$. The experiment has been performed with different parameter values obtaining similar conclusions.

It seems that $I((X_1, X_2); C)$ and $I(X_1; C) + I(X_2; C)$ are inversely proportional to ϵ_{mul} and ϵ_{uni} respectively (see Figures 5.1(a) and 5.1(b)). Besides, $I(X_1; X_2; C) = I(X_1; C) + I(X_2; C) - I((X_1, X_2); C)$ seems to be inversely proportional to ϵ_{dif} (see Figure 5.1(c)). Therefore, given a pair of continuous predictors, the lower $I(X_1; X_2; C)$ becomes, the more advisable it is to use a classifier which models $p_\phi(c|x_1, x_2)$ in a multivariate way.

5.2.2 Filter TAN structural learning algorithms

In this section we theoretically analyze the promising discriminative structural learning of TAN structures proposed by Pernkopf and Bilmes (2005). Besides, we empirically study the algorithm in both artificial and real-world data sets. The algorithm is compared with the classic and well known generative version proposed by Friedman et al. (1997).

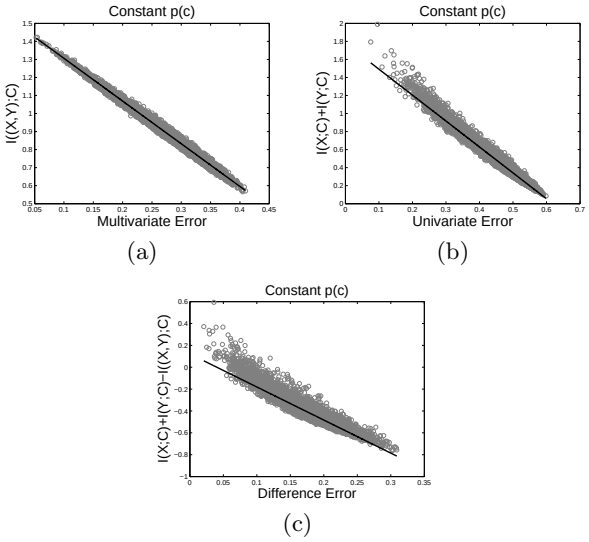


Fig. 5.1. This figure shows the errors ϵ_{mul}, ϵ_{uni} and ϵ_{dif} versus measures based on information theory.

5.2.2.1 Generative tree augmented naive Bayes

TAN structure was introduced by Friedman et al. (1997) in order to relax the strong conditional independence assumption between predictor variables made by a naive Bayes structure (see Section 2.5.2 for further details on TAN and naive Bayes structures).

In TAN structures each predictor may have at most one predictor variable as parent, plus the class variable. In Friedman et al. (1997) a generative algorithm is proposed for learning a classifier with TAN structure which maximizes the likelihood $L(\phi|\mathcal{S})$. We name this algorithm filter multinomial TAN and it has been introduced in detail in Section 2.5.3.2. In this section we call this algorithm *generative TAN* for the sake of clarity and simplicity. The structural learning procedure of generative TAN constructs a maximum weighted spanning tree, using the mutual information conditioned to the class $I(X_i; X_j|C)$, following the procedure proposed by Chow and Liu (1968). The algorithm builds a TAN model that maximizes the likelihood and it is asymptotically correct if the data have been generated from a TAN structure.

5.2.2.2 Discriminative tree augmented naive Bayes

This section analyzes the theoretical properties of *discriminative TAN* algorithm proposed by Pernkopf and Bilmes (2005) (see Algorithm 14). Discriminative TAN is equivalent to generative TAN introduced by Friedman et al. (1997) with two differences:

- Instead of using the conditional mutual information between two attributes given the class variables, it uses the 3-way interaction information (see Equation 1.59).
- Edges (X_i, X_j) with a negative estimation of the 3-way interaction information, $I(X_i; X_j; C)$, are forbidden. This difference provides the advantage with respect to its generative version of removing useless arcs between variables without the need of any threshold measure.

Algorithm 14: **Discriminative TAN algorithm.**

Inputs
  Training set $\mathcal{S}$.
Outputs
  TAN structure.
Procedure
  $\forall i, j / i < j$ compute $I(X_i; X_j; C)$.
  Construct an undirected complete graph with the predictors as nodes $\{X_1, ..., X_n\}$. Label each edge $\overline{X_i X_j}$ with $I(X_i; X_j; C)$.
  Construct the maximum spanning tree from the undirected complete one using the Chow and Liu procedure [Chow and Liu (1968)].
  Delete the edges $\overline{X_i X_j}$ with $I(X_i; X_j; C) < 0$.
  Direct the forest structure obtained choosing at random a root node X_i at each connected component. Add arcs from the class variable C to each predictor $\{X_1, ..., X_n\}$.

5.2.3 Analyzing discriminative TAN

This section analyzes the theoretical properties of discriminative TAN algorithm taking into account the information theory measures based on the model ϕ: ϕ-conditional entropy, $H_\phi(C|\boldsymbol{X})$ (see Equation 5.2), and ϕ-mutual information, $I_\phi(\boldsymbol{X}; C)$ (see Equation 5.3).

Theorem 9. *Let C be a discrete random variable and $\boldsymbol{X} = (X_1, ..., X_n)$ a multivariate discrete random variable, and let $\mathcal{S}$ be a collection of N instances, $\mathcal{S} = \{(\boldsymbol{x}^{(1)}, c^{(1)}), ..., (\boldsymbol{x}^{(N)}, c^{(N)})\}$. Given a joint probability function $p(\boldsymbol{x}, c)$ learned from data $\mathcal{S}$ and the set of marginal distributions obtained by its marginalization, the* discriminative TAN *algorithm (see Figure 14) builds a TAN structure $\boldsymbol{s}$ that does not necessarily maximize $CLL(\phi|\mathcal{S})$.*

Proof. Maximizing conditional likelihood is equivalent to maximizing the conditional log likelihood:

$$CLL(\phi|\mathcal{S}) = \sum_{j=1}^{N} \log p_\phi(c^{(j)}|\boldsymbol{x}^{(j)})$$

where $p_\phi(c|\boldsymbol{x})$ is given by the joint distribution $p_\phi(\boldsymbol{x}, c)$ modeled by the classifier ϕ, which has been learned from $\mathcal{S}$.

Using the empirical distribution, $\hat{p}(\boldsymbol{x}, c)$, we can reformulate the conditional log likelihood in terms of ϕ-conditional entropy defined in Equation 5.1:

$$\begin{aligned} CLL(\phi|\mathcal{S}) &= -N(H_\phi(C|\boldsymbol{X})) \\ &\propto -H_\phi(C) + I_\phi(\boldsymbol{X}; C) \end{aligned} \tag{5.11}$$

Given $p(\boldsymbol{x}, c)$, $H_\phi(C)$ is constant and thus, minimizing $H_\phi(C|\boldsymbol{X})$ is equivalent to maximizing $I_\phi(\boldsymbol{X}; C)$ (Equation 5.11). Using the chain rule for the mutual information [Cover and Thomas (1991)] we can reformulate $I_\phi(\boldsymbol{X}; C)$ as follows:

$$\begin{aligned} I_\phi(\boldsymbol{X}; C) &= \sum_{i=1}^{n} I(X_i; C|X_1, ..., X_{i-1}) \\ &= \sum_{i=1}^{n} I(X_i; C) \\ &\quad + \sum_{i=1}^{n} I(X_i; \boldsymbol{Pa}_i; C) \\ &\quad - \sum_{i=1}^{n} I(X_i; \neg\boldsymbol{Pa}_i|\boldsymbol{Pa}_i) \end{aligned}$$

where $\neg\boldsymbol{Pa}_i = \{X_1, ..., X_{i-1}\}\backslash\boldsymbol{Pa}_i$ and $I(X_j; \boldsymbol{Pa}_i; C) = I(X_j; \boldsymbol{Pa}_i|C) - I(X_j; \boldsymbol{Pa}_i)$. Considering the constraints of the TAN structures, $I_\phi(\boldsymbol{X}; C)$ can be rewritten as:

$$\begin{aligned} I_\phi(\boldsymbol{X}; C) &= \sum_{i=1}^{n} I(X_i; C) \\ &\quad + \sum_{i=1}^{n} I(X_i; Pa_i; C) \\ &\quad - \sum_{i=1}^{n} I(X_i; \neg\boldsymbol{Pa}_i|Pa_i) \end{aligned} \tag{5.12}$$

The first term is constant in TAN structures. The second term is maximized by the algorithm discriminative TAN (see Figure 14). However, the second term is not considered by discriminative TAN and it can be greater than zero. We call the quantity $\sum_{i=1}^{n} I(X_i; \neg\boldsymbol{Pa}_i|Pa_i)$ the residual (see Equation 5.12). This term becomes zero when the following set of conditional independencies is true $\{\forall i \in \{1, ..., n\} : \neg\boldsymbol{Pa_i} \perp X_i|Pa_i \wedge Pa_i \in \{X_1, ..., X_{i-1}\} \wedge |Pa_i| \leq 1\}$. These constraints are not true in general for TAN models. Moreover, the residual is not constant for all TAN structures.

Therefore, discriminative TAN algorithm does not necessarily maximize the conditional log likelihood. ■.

Besides, as a consequence of Proposition 5.4 and Corollary 5.2, discriminative TAN procedure does not necessarily minimize $E_{\hat{p}(\boldsymbol{x})}(D_{KL}(\hat{p}(c|\boldsymbol{x})||p_{\phi}(c|\boldsymbol{x}))$.

Discriminative TAN algorithm needs to compute $\mathcal{O}(n^2)$ 3-way interaction information statistics. If the complexity of the estimator used to compute $I(X_i; X_j; C)$ is $\mathcal{O}(I)$, the time complexity of discriminative TAN results in $\mathcal{O}(n^2 I)$. Usually, the computational cost to estimate $I(X_i; X_j; C)$ is similar to $I(X_i; X_j; C)$. Thus, usually, discriminative TAN and generative TAN [Friedman et al. (1997); Pérez et al. (2006b)] have the same the computational complexity.

5.2.3.1 Analyzing the residual term

We call the quantity $\sum_{i=1}^{n} I(X_i; \neg \boldsymbol{Pa}_i | Pa_i)$ the residual (see Equation 5.12). This term becomes zero when the following set of conditional independencies is true $\{\forall i \in \{1, ..., n\} : \neg \boldsymbol{Pa_i} \perp X_i | Pa_i \wedge Pa_i \in \{X_1, ..., X_{i-1}\} \wedge |Pa_i| \leq 1\}$. These constraints are not true in general for TAN models.

5.2.4 Experimental results

This section includes the experimentation in both artificial and real-world continuous domains. The experimentation has been performed for algorithms which induct TAN structures in both discriminative (see Figure 14) and generative [Friedman et al. (1997)] ways. It must be noted that parametric learning is performed in a generative way, maximizing the likelihood [Geiger and Heckerman (1994); Friedman et al. (1997)]. The experimentation has been performed for the following algorithms:

- Generative flexible TAN (GF) [Pérez et al. (2009)].
- Discriminative flexible TAN (DF).
- Generative Gaussian TAN (GG) [Pérez et al. (2006b)].
- Discriminative Gaussian TAN (DG).
- Generative multinomial TAN (GM) [Friedman et al. (1997)].
- Discriminative multinomial TAN (DM) [Pernkopf and Bilmes (2005)].

The term *flexible* means that inducted classifier is based on KBN [Pérez et al. (2009)], *Gaussian* is based on conditional Gaussian networks [Pérez et al. (2006b)], and *multinomial* is based on Bayesian multinomial networks (see Chapters 2, 3 and 4 for further details on BMN, CGN and KBN paradigms, respectively). The mutual information quantities $I(X_i; X_j)$ and $I(X_i; X_j|C)$ [Cover and Thomas (1991)] used by the algorithms have been estimated using the KDE [Pérez et al. (2009)], Gaussian [Pérez et al. (2006b); Geiger and Heckerman (1994); Cover and Thomas (1991)] and multinomial [Friedman et al. (1997); Cover and Thomas (1991)] approaches, respectively. In order to work

with TAN models based on BMN, we have performed independent discretizations for each pair training-test, learning the discretization policy using only the training set. We have decided to use the supervised entropy discretization algorithm [Fayyad and Irani (1993)] because it obtains competitive performances in supervised classification tasks. When the entropy algorithm only creates one interval, equal frequency discretization with 3 intervals has been used.

A. Artificial data sets

This section shows the results obtained in a set of continuous artificial domains generated using the procedure of Figure 12. Table 5.4 summarizes the obtained results. Each entry of the table represents a mean estimated classification error and a Wilcoxon paired test between the estimated errors across all data sets following the procedure described in Demšar (2006). Each mean estimated error has been obtained using the following procedure:

- Repeat 50 times:
  - Generate a KBN following the procedure described in Figure 12 with parameters $r = rand(2, 3, 4)$, $p(c)$ at random, $n \in \{4, 8, 16, 32\}$, $k \in rand\{1, ..., n-1\}$ and $m = 1024$. The value of m has been increased to 2048 for $n = 32$ in order to obtain similar errors.
  - Generate a test set $\mathcal{S}_T$ of size 1000, sampling the generated KBN.
  - Sample three training sets $\mathcal{S}_N$ of size $N = \{20, 400, 8000\}$ from the generated KBN.
  - Repeat for each training set $\mathcal{S}_N$ generated: Obtain the errors of this domain using a holdout scheme ($\mathcal{S}_N$ for training and $\mathcal{S}_T$ for testing).
- Compute the mean error value across the 50 domains.
- Compute a Wilcoxon paired test at $\alpha = 1\%$ significance level with each paradigm using the estimated errors obtained by the generative and discriminative models, across the 50 artificial domains following the procedure described in Demšar (2006).

A set of statistically significant difference of errors is shown in Table 5.4. Most of them indicate that the generative approach is better. In spite of being statistically significant, they represent small differences (less than 1%) (see Figure 5.2).

B. UCI data sets

The results provided in this section have been obtained in a set of continuous real-world domains taken from the UCI repository [Asuncion and Newman (2007)] (see Figure 5.2). Predictor variables with a variance less than 10^{-4} have been removed. The classification error has been estimated using a stratified 10-fold cross-validation. The results have been summarized in Table 5.5.

There is no statistically significative difference between the discriminative and generative versions at $\alpha = 1\%$ significance level taking into account all

		Number of variables			
Training size	Classifiers	4	8	16	32
20	GF	∘ 38.3	∘ 36.3	36.9	32.2
	DF	39.3	36.9	36.7	32.4
	GG	41.5	40.0	40.7	37.4
	DG	41.2	40.1	40.8	37.2
	GM	42.7	41.4	40.0	34.1
	DM	∘ 42.3	∘ 40.5	39.5	33.8
400	GF	∘ 26.9	21.5	15.7	20.0
	DF	27.5	21.4	15.8	20.1
	GG	29.5	24.0	17.9	23.0
	DG	29.7	24.3	17.8	23.0
	GM	30.5	25.3	18.6	23.3
	DM	30.5	25.3	18.7	23.6
8000	GF	23.3	∘ 16.2	13.1	17.1
	DF	23.4	16.6	13.1	17.2
	GG	29.3	22.1	17.3	19.9
	DG	29.3	22.3	17.0	20.0
	GM	22.5	16.6	∘ 12.6	17.5
	DM	22.7	16.8	13.0	17.6

Table 5.4. Mean errors of the classifiers in generated artificial data sets. The circles ∘ represent statistically significant differences between the generative and discriminative versions of the algorithm.

	Error					
Label	GF	DF	GG	DG	GM	DM
Balance	12.3	11.7	11.4	11.4	29.0	28.6
Block	5.6	5.6	7.8	8.4	4.5	48.4
Haberman	26.2	25.5	24.2	24.9	30.4	31.0
Image	14.5	16.4	12.9	14.3	6.1	12.2
Ionosphere	8.3	7.7	7.7	8.8	8.5	9.9
Iris	4.0	4.0	2.7	3.3	6.7	6.0
Letter	19.0	23.7	28.5	28.3	15.4	14.9
Liver	38.5	38.7	41.7	43.4	35.4	36.8
Pima	28.2	26.0	24.5	25.8	24.7	25.4
Satellite	13.3	15.3	16.8	16.7	12.2	17.8
Sonar	26.0	26.0	35.1	29.4	21.2	20.7
Thyroid	1.6	3.7	4.2	3.3	6.5	6.5
Vehicle	37.1	37.8	23.9	37.9	29.8	31.0
Vowel	10.7	11.2	22.8	26.8	28.2	27.2
Waveform	19.5	19.7	17.8	19.5	17.9	19.6
Wine	2.3	0.6	0.6	1.1	2.2	1.7
Yeast	40.6	41.1	43.3	44.1	43.0	42.9

Table 5.5. Errors of the different classifiers in selected UCI data sets.

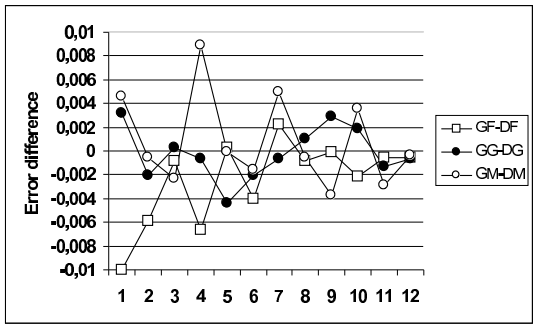


Fig. 5.2. This graph shows the error of the generative approach minus the error of the discriminative approach ($\epsilon_{gener.} - \epsilon_{discrim.}$) for the artificial data sets. Each value in the axis X corresponds to a configuration of the number of variables and the training size: 1-(4 variables, 20 cases), 2-(4 variables, 400 cases),...,11-(32 variables, 400 cases), 12-(32 variables, 8000 cases).

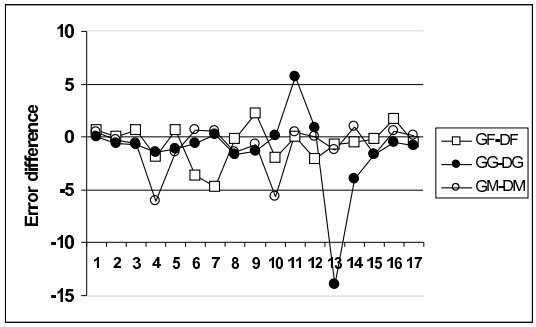


Fig. 5.3. This figure shows the error of the generative approach minus the error of discriminative approach in the selected real-world data sets. The values in the axis X correspond to the indexes of the data sets (see Table 5.2)

the data sets for each kind of BN. The differences between the errors of the generative and discriminative approaches at each data set are shown in Figure 5.3. In most of the data sets the differences are smaller than 3%.

Taking into account the results presented in this section for both artificial and real-world data sets and the results of Section 5.1.4, it seems that the influence of the residual term should be taken into account by the discriminative TAN algorithm.

5.3 Conclusions

This chapter intuitively shows the relations among the classification error ϵ_ϕ, the conditional entropy $H_\phi(C|\boldsymbol{X})$, the mutual information $I_\phi(C;\boldsymbol{X})$, and the

conditional log likelihood $CLL(\phi|\mathcal{S})$. The chapter provides a set of illustrative experimental results which support the presented intuitions. Besides, we present a theoretical result which relates the conditional likelihood $CLL(\phi|\mathcal{S})$ with the conditional entropy $H_\phi(C|\boldsymbol{X})$. We theoretically analyze the promising discriminative structural learning of tree-augmented naive Bayes classifiers proposed by Pernkopf and Bilmes (2005), proving that it does not necessarily maximize the conditional likelihood of the obtained model.

The main future line of work consists of proposing new algorithms for the induction of classifiers based on Bayesian networks guided by conditional likelihood.

Part III

Conclusions and future work

6

Conclusions

This chapter summarizes the main contributions and conclusions provided in this dissertation. Moreover, this chapter addresses a set of general directions for further research. More specific conclusions and future work lines have been exposed in each corresponding chapter.

6.1 Summary of the dissertation

The summary of this document can be divided into two parts: background concepts related to supervised classification and Bayesian networks, and methodological contributions.

6.1.1 Background concepts

The literature review presented in the first part of this dissertation tries to cover the most relevant aspects of supervised classification based on Bayesian networks. This part is divided into Chapter 1 and Chapter 2.

Chapter 1 formally presents a set of relevant concepts and procedures for supervised classification. It covers the following issues: probability theory, definition of supervised classification and some related problems, decision theory, generative, conditional and discriminative learning of classifiers, estimation theory, error estimation, comparison of classifier induction algorithms, bias plus variance decomposition of the error [Kohavi and Wolpert (1996)], different performance measures based on confusion matrix and ROC curves, information theory, curse of dimensionality and feature subset selection, discretization, and non-parametric density estimation.

In Chapter 2 we focus our attention on the Bayesian multinomial network paradigm for supervised classification. For this purpose, we have introduced some concepts of graph theory focusing on directed acyclic graphs, probabilistic graphical models, Bayesian networks for supervised classification, *aug-*

mented naïve Bayes family of structures and a set of classifier induction algorithms for each kind of augmented naïve Bayes structure. Besides, we have explained the main drawbacks of Bayesian multinomial network paradigm in order to deal with continuous features and we have presented the most popular approaches to avoid them.

6.1.2 Methodological contributions

The main contributions provided in this document have a methodological nature. This dissertation is concerned with supervised classification in domains with continuous random variables, Bayesian networks and information theory. The methodological contributions can be divided into three parts: conditional Gaussian networks, kernel based Bayesian networks and information theory.

The main contributions concerning conditional Gaussian networks can be summarized as follows:

- We have collected a set of theoretical results taken from [Lauritzen (1996)] regarding joint, marginal and conditional forms of *Gaussian density function* and *mixed Gaussian distribution*. We have provided alternative proofs to some of these theoretical results.
- We have proposed the formulation of a set of measures of information theory under some assumptions related to mixed Gaussian distribution.
- The *conditional Gaussian network* paradigm is formally defined [Bottcher (2004)] analyzing the properties of the local factors. Besides, we have included a discussion about alternative definitions of conditional Gaussian networks.
- A set of classifiers for each subfamily of augmented naïve Bayes structures is presented: *ranking selective Gaussian naïve Bayes*, *wrapper selective Gaussian naïve Bayes*, *Gaussian tree-augmented naïve Bayes*, *wrapper selective Gaussian tree-augmented naïve Bayes*, *filter Gaussian k-dependent augmented naïve Bayes*, *wrapper selective Gaussian k-dependent augmented naïve Bayes*, *wrapper forward Gaussian joint augmented naïve Bayes*, *wrapper backward Gaussian joint augmented naïve Bayes* and *wrapper condensed Gaussian joint augmented naïve Bayes*.
- We have included an experimental study in continuous real-world domains of the presented classifiers based on conditional Gaussian networks. The errors of the classifiers are estimated and, then, they are compared following a multiple comparison procedure [García and Herrera (2008)]. In addition, the behavior of the classifiers and the sources of the error are analyzed using the bias plus variance decomposition of the error [Kohavi and Wolpert (1996)].

For further details on the methodological contributions related to conditional Gaussian networks the reader should consult Chapter 3.

The main contributions concerning kernel based Bayesian networks are summarized in the following list:

- In order to learn the bandwidth matrix with classification purposes, we have proposed the *normal rule plus differential scaled* heuristic.
- *Gaussian kernel probability density function* and *mixed Gaussian kernel distribution* have been introduced. Besides, we have provided a set of theoretical results regarding joint, marginal and conditional forms of Gaussian kernel probability density function and mixed Gaussian kernel distribution.
- We have introduced a set of *non-parametric estimators for quantities of information theory* which can be thought of as being based on mixed Gaussian kernel distributions.
- *Kernel based Bayesian network* paradigm has been formally defined based on the results provided for mixed Gaussian kernel distributions.
- A set of classifiers based on augmented naïve Bayes structures has been presented for kernel based Bayesian networks: *flexible naïve Bayes*, *flexible tree-augmented naïve Bayes*, *flexible k-dependent augmented naïve Bayes* and *flexible complete graph classifier*. These classifiers have been named *flexible classifiers* [Pérez et al. (2009)].
- We have analyzed the storage and computational complexity of the proposed classifier induction algorithms.
- We have studied the asymptotic properties of the results based on mixed Gaussian kernel distribution: the definition of kernel based Bayesian network, the flexible classifiers and the provided non-parametric estimators of the quantities of information theory.
- Flexible classifiers have been empirically studied using artificial and real-world data sets. We have analyzed the behavior of the flexible classifiers and their sensitivity to changes in the smoothing degree using artificial domains. We have estimated the classification errors of a set of benchmarks and the proposed flexible classifiers in 21 continuous data sets from UCI repository [Asuncion and Newman (2007)]. Based on the estimated errors, we have performed different comparisons among the classifiers included in the study. Besides, using the real-world data sets, we have studied the effect of the smoothing degree in the performance of flexible classifiers and the bias plus variance decomposition of the expected error [Kohavi and Wolpert (1996)].

For further details on the methodological contributions related to kernel based Bayesian networks the reader may consult Chapter 4.

The main contributions concerning the relation between information theory and classification error are summarized in the following list:

- We have intuitively explained the connections between a set of measures of information theory, conditional likelihood and classification error.
- We have provided a set of theoretical results which support some of the previously introduced intuitions.

- The relation between the error and information theory have been empirically studied for tree-augmented naïve Bayes structures, using artificial and real-world domains.
- *Discriminative tree-augmented naïve Bayes* classifier induction algorithm [Pernkopf and Bilmes (2005)] has been theoretically studied and we show that it does not necessarily maximize the conditional likelihood. A modification of the algorithm which takes into account the influence of the residual term is proposed. The original algorithm has been compared to its generative version for Bayesian multinomial network, conditional Gaussian network and kernel based Bayesian network paradigms. The experimentation has been performed in artificial and real-world domains.

For further details on the methodological contributions related to information theory and classification error the reader may consult Chapter 5.

6.2 Publications

The research work performed during the thesis period has produced the following publications in international JCR journals:

- **A. Pérez** and P. Larrañaga and I. Inza (2009) Bayesian classifiers based on kernel density estimation: Flexible classifiers. *International Journal of Approximate Reasoning*, 50(2):341–362.
- J. D. Rodríguez, **A. Pérez** and J. A. Lozano (2009) Sensitivity analysis of k-fold cross-validation in prediction error estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. In press.
- J. A. Fernandes, X. Irigoien, N. Goikoetxea, J. A. Lozano, I. Inza, **A. Pérez** and A. Bode (2009) Fish recruitment prediction, using robust supervised classification methods. *Ecological Modelling*. In press.
- M. Guid, **A. Pérez** and I. Bratko (2008) How trustworthy is Crafty's analysis of world chess champions?. *International Computer Games Association Journal*, 31(3):131-144.
- **A. Pérez** and P. Larrañaga and I. Inza (2006). Supervised classification with conditional Gaussian networks: Increasing the structure complexity from naïve Bayes. *International Journal of Approximate Reasoning*, 43; 1–25.
- P. Larrañaga, B. Calvo, R. Santana, C. Bielza, J. Galdiano, I. Inza, J. A. Lozano, R. Armañanzas, G. Santafé, **A. Pérez** and V. Robles (2006) Machine learning in Bioinformatics. *Briefings in Bioinformatics*, 7:86-112.

In addition, the author has produced the following publications in conferences and workshops:

- **A. Pérez** and P. Larrañaga and I. Inza (2006) Information theory and classification error in probabilistic classifiers. *Proceedings of the Ninth International Conference on Discovery Science. Lecture Notes in Artificial Intelligence*, 4265:347–351.
- M. Guid, **A. Pérez** and I. Bratko (2007) How trustworthy is Crafty's analysis of world chess champions? *Workshop on Computer Games*, 15–26.
- **A. Pérez**, P. Larrañaga e I. Inza (2005) Modelos gráficos probabilísticos para la clasificación supervisada empleando la estimación basada en kernels Gaussianos esféricos. *Primer Congreso Español de Informática. Actas del Tercer Taller de Minería de Datos y Aprendizaje*, 3:125–134.

6.3 Future work

The topics of the future work can be divided into four groups: Bayesian multinomial networks, conditional Gaussian networks, kernel based Bayesian networks and, information theory and classification error.

Regarding Bayesian multinomial networks, we are working on the design of new classifier induction algorithms which control the bias and the variance of the learned classifiers for augmented naïve Bayes structures. The idea consists of controlling the available number of cases to compute reliable parameters of the classifiers. This heuristic should directly control the variance component of the classification error. Moreover, due to the bias and variance trade-off, it controls the bias component. An example of this family of algorithms is the wrapper selective dynamic multinomial kAN (see Section 2.7).

We suggest three future work lines related to the conditional Gaussian network paradigm:

- We will provide a more flexible paradigm to deal with mixed domains under Gaussian assumption by breaking with the structural constraints imposed on the paradigm.
- Novel estimators for the quantities based on information theory under Gaussian assumptions will be proposed.
- We will empirically evaluate alternative estimations to maximum likelihood for learning the parameters of the model. Two are the main alternatives: a generative learning of parameters more stable to outliers than maximum likelihood, and a discriminative learning of the parameters based on conditional log likelihood and classification error.

We suggest four future work lines regarding kernel based Bayesian networks:

- The experimental part presented in this document will be extended to mixed domains.
- We will relax the strong computational requirements of the paradigm for classifying new instances. We consider approximating the kernel-based estimation by means of a semi-parametric approach. This reduction in the classification time will allow us to design and implement other search techniques in the space of possible structures.
- We will apply the novel paradigm to inference problems in mixed domains.
- The structural constraint imposed on the paradigm will be broken and we will propose a novel definition based on mixed variable Gaussian kernel distribution.

Concerning information theory and supervised classification, we will propose novel discriminative classifier induction algorithms for Bayesian networks guided by conditional likelihood.

References

Aladjem, M., 2002. Projection pursuit fitting Gaussian mixture models. In: Proceedings of Joint IAPR. Vol. 2396 of Lecture Notes in Computer Science. pp. 396–404.

Aladjem, M., 2005. Projection pursuit mixture density estimation. IEEE Transactions on Signal Processing 53 (11), 4376–4383.

Ali, S. M., Silvey, S. D., 1966. A general class of coefficient of divergence of one distribution from another. Journal of the Royal Statistical Society (286), 131–142.

Anderson, F. W., 1958. An Introduction to Multivariate Statistical Analysis. John Wiley and Sons.

Anscombe, F. J., 1967. Topics in the investigation of linear relations fitted by the method of least squares. Journal of the Royal Statistical Society Series B 29, 1–52.

Asuncion, A., Newman, A., 2007. UCI machine learning repository. Tech. rep., Irvine, CA: University of California, School of Information and Computer Science [http://www.ics.uci.edu/ mlearn/MLRepository.html].

Bell, D. A., Wang, H., 2000. A formalism for relevance and its application in feature subset selection. Machine Learning 41, 175–195.

Ben-Bassat, M., 1982. Pattern recognition and reduction of dimensionality. In: Krishnaiah, P. R., Kanal, L. N. (Eds.), Handbook of Statistics. Vol. 2. North-Holland, pp. 773–791, Amsterdam.

Bengio, Y., Grandvalet, Y., 2004. No unbiased estimator of the variance of k-fold cross-validation. The Journal of Machine Learning Research 5, 1089–1105.

Bengio, Y., Grandvalet, Y., 2005. Bias in estimating the variance of k-fold cross-validation. Statistical Modeling and Analysis for Complex Data Problems 1, 75–95.

Bilmes, J., 2000. Dynamic Bayesian multinets. In: Proceedings of the 16th International Conference of Uncertainty in Artificial Intelligence. pp. 38–45.

Bilmes, J. A., 1997. A gentle tutorial on the EM algorithm and its application to parameter estimation for Gaussian mixture models. Tech. Rep. ICSI-TR-97-021, University of Berkeley.

Bishop, C. M., 2006. Pattern Recognition and Machine Learning. Information Science and Statistics. Springer.

Blanco, R., 2005. Learning Bayesian Networks from Data with Factorisation and Classification Purposes. Applications in Biomedicine. Ph.D. thesis, University of the Basque Country.

Blanco, R., Larrañaga, P., Inza, I., Sierra, B., 2003. Gene selection for cancer classification using wrapper approaches. International Journal of Pattern Recognition and Artificial Intelligence 18 (8), 1373–1390.

Bottcher, S. G., 2004. Learning Bayesian Networks with Mixed Variables. Ph.D. thesis, Aalborg University.

Bouckaert, R. R., 1995. Bayesian Belief Networks: From Construction to Inference. Ph.D. thesis, University of Utrecht.

Bouckaert, R. R., 2004. Naive Bayes classifiers that perform well with continuous variables. In: Proceedings of the Seventeenth Australian Conference on Artificial Intelligence. pp. 1089–1094.

Braga-Neto, U. M., 2005. Small-sample error estimation: mythology versus mathematics. In: Proceedings of the International Society for Optical Engineering. Vol. 5916. pp. 304–314.

Braga-Neto, U. M., Dougherty, E., 2004. Bolstered error estimation. Pattern Recognition 37, 1267–1281.

Burman, P., 1989. A comparative study of ordinary cross-validation, v-fold cross-validation and the repeated learning-testing methods. Biometrika 76, 503–514.

Burman, P., Chow, E., Nolan, D., 1994. A cross-validation method for dependent data. Biometrika 81, 351–358.

Casella, G., Berger, R. L., 1990. Statistical Inference. Wadsworth and Brooks.

Castillo, E., Gutierrez, J. M., Hadi, A. S., 1997. Expert Systems and Probabilistic Network Models. Springer-Verlag.

Cerquides, J., de Mántaras, R. L., 2003. The indifferent naïve Bayes classifier. Tech. Rep. TR-2003-01, Institut d'Investigació de Inteligéncia Artificial. CSIC.

Cerquides, J., de Mántaras, R. L., 2005. TAN classifiers based on decomposable distributions. Machine Learning 59 (3), 323–354.

Cestnik, B., Kononenko, I., Bratko, I., 1987. ASSISTANT-86: A knowledge elicitation tool for sophisticated users. In: Bratko, I., Lavrac, N. (Eds.), Progress in Machine Learning. Sigma Press, pp. 31–45.

Cheng, J., Greiner, R., 1999. Comparing Bayesian network classifiers. In: Proceedings of the 15th Conference on Uncertainty in Artificial Intelligence. pp. 101–107.

Chernick, M., 1999. Bootstrap Methods: A Practitioner Guide. John Wiley and Sons, New York.

Chickering, D. M., 2002. Learning equivalence classes of Bayesian-network structures. Journal of Machine Learning Research 2, 445–498.

Chow, C., Liu, C., 1968. Approximating discrete probability distributions with dependence trees. IEEE Transactions on Information Theory 14, 462–467.

Cochran, W. G., 1968. Commentary on estimation of error rates in discriminant analysis. Technometrics 10, 204–205.

Cooper, G. F., Herskovits, E. A., 1992. A Bayesian method for the induction of probabilistic networks from data. Machine Learning 9, 309–347.

Cormen, T. H., Charles, L. E., Ronald, R. L., Clifford, S., 2003. Introduction to Algorithms. MIT Press.

Cover, T. M., Thomas, J. A., 1991. Elements of Information Theory. John Wiley and Sons.

Cover, T. T., Hart, P. E., 1967. Nearest neighbour pattern classification. IEEE Transactions on Information Theory 13, 21–27.

Cowell, R. G., 2001. On searching for optimal classifiers among Bayesian networks. In: Proceedings of the 8th International Conference on Artificial Intelligence and Statistics. pp. 175–180.

Cowell, R. G., Ramoni, M., Sebastiani, P., 1999. An experimental evaluation of the predictive accuracy of Bayesian networks. In: Book of Abstracts of the Second European Conference on Highly Structured Stochastic Systems. University of Pavia, pp. 75–76.

Dash, D., Cooper, G. F., 2002. Exact model averaging with naïve Bayesian classifiers. In: Proceedings of the Nineteenth International Conference on Machine Learning. pp. 91–98.

Dash, D., Cooper, G. F., 2003. Model averaging with discrete Bayesian network classifiers. In: Proceedings of the 9th Artificial Intelligence and Statistics.

Dash, D., Cooper, G. F., 2004. Model averaging for prediction with discrete Bayesian networks. Journal of Machine Learning Research 5, 1177–1203.

de Boor, C., 2001. A Practical Guideline to Splines, revised Edition. Vol. 27 of Applied Mathematical Sciences. Springer-Verlag, New York.

Delaigle, A., Gijbels, I., 2002. Comparison of data-driven bandwidth selection procedures in deconvolution kernel density estimation. Computational Statistics and Data Analysis 39, 1–20.

Demšar, J., 2006. Statistical comparisons of classifiers over multiple data sets. Journal of Machine Learning Research 7, 1–30.

Devroye, L., Gyorfi, L., Lugosi, G., 1996. A Probabilistic Theory for Pattern Recognition. Springer.

Devroye, L. P., 1983. The equivalence in L1 of weak, strong and complete convergence of kernel density estimates. Annals of Statistics 11, 896–904.

Devroye, L. P., Wagner, T. J., 1979. Distribution-free perfomance bounds with the resubstitution error estimate. IEEE Transactions on Information Theory 25 (2), 208–210.

Diamantidis, N. A., Karlis, D., Giakoumakis, E. A., 2000. Unsupervised stratification of cross-validation for accuracy estimation. Artificial Intelligence 116, 1–16.

Dietterich, T. G., 1999. Approximate statistical tests for comparing supervised classification learning algorithms. Neural Computation 10 (7), 1895–1924.

Doak, J., 1992. An evaluation of feature selection methods and their application to computer security. Tech. rep., University of California at Davis.

Domingos, P., 2000. A unified bias-variance decomposition and its applications. In: Proceedings of the 17th International Conference on Machine Learning. Morgan Kaufman, pp. 231–238.

Domingos, P., Pazzani, M., 1997. On the optimality of the simple Bayesian classifier under zero-one loss. Machine Learning 29, 103–130.

Dougherty, J., Kohavi, R., Sahami, M., 1995. Supervised and unsupervised discretization of continuous features. In: Proceedings of the 12th International Conference on Machine Learning. pp. 194–202.

Droge, B., 1996. Some comments on cross-validation. In: Härdle, W., Schimek, M., Hardle, W. (Eds.), Statistical Theory and Computational Aspects of Smoothing. pp. 178–199.

Duda, R., Hart, P., 1973. Pattern Classification and Scene Analysis. John Wiley and Sons.

Duda, R., Hart, P., Stork, D., 2000. Pattern Classification. John Wiley and Sons.

Dudewicz, E. J., Mishra, S. N., 1988. Modern Mathematical Statistics. John Wiley and Sons.

Efron, B., Gong, G., 1983. A leisurely look at the bootstrap, the jacknife and cross-validation. The American Statistician 37, 36–48.

Efron, B., Tibishirani, R. J., 1993. An Introduction to the Bootstrap. Monographs on Statistics and Applied Probability. Chapman and Hall.

Efron, B., Tibshirani, R. J., 1995. Cross-validation and the bootstrap: Estimating the error rate of a prediction rule. Tech. rep., Dept. Statistics, Stanford University.

Egmont-Peterson, M., 2004. Feature selection by Markov chain Monte Carlo sampling: A Bayesian approach. In: Proceedings of the Joint IAPR Workshops SSPR 2004 and SPR 2004. pp. 1034–1042.

Fayyad, U., Irani, K., 1993. Multi-interval discretization of continuous-valued attributes for classification learning. In: Proceedings of the 13th International Conference on Artificial Intelligence. pp. 1022–1027.

Figueiredo, M. A. T., Leitao, J. M. N., Jain, A. K., 1999. On fitting mixture models. In: Energy Minimization Methods in Computer Vision and Pattern Recognition. Vol. 1654 of Lecture Notes in Computer Science. pp. 732–749.

Fisher, R. A., 1936. The use of multiple measurements. Annals of Eugenics 7, 179–188.

Friedman, J. H., 1997. On bias, variance, 0/1 - loss, and the curse-of-dimensionality. Data Mining and Knowledge Discovery 1, 55–77.

Friedman, M., 1937. The use of ranks to avoid the assumption of normality implicit in the analysis of variance. Journal of the American Statistical Association 32, 675–701.

Friedman, N., Geiger, D., Goldszmidt, M., 1997. Bayesian network classifiers. Machine Learning 29, 131–163.

Friedman, N., Goldsmidt, M., Wyner, A., 1999. Data analysis with Bayesian networks: A bootstrap approach. In: Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence. pp. 196–205.

Frydemberg, M., 1990. Marginalization and collapsibility in graphical interaction models. The Annals of Statistics 18 (2), 790–805.

Fukunaga, K., 1972. Statistical Pattern Recognition. Academic Press, Inc.

Fukunaga, K., Hummels, D. M., 1989. Leave-one-out procedures for nonparametric error estimates. IEEE Transactions on Pattern Analysis and Machine Intelligence 11 (4), 421–423.

Gammerman, A., Thatcher, A. R., 1991. Bayesian diagnostic probabilities without assuming independence of symptoms. Methods of Information in Medicine 30, 15–22.

García, S., Herrera, F., 2008. An extension on statistical comparisons of classifiers over multiple data sets for all pairwise comparisons. Journal of Machine Learning Research 9, 2677–2694.

Geiger, D., Heckerman, D., 1994. Learning Gaussian networks. Tech. rep., Microsoft Research, Advanced Technology Division.

German, S., Bienenstock, E., Doursat, R., 1992. Neural networks and the bias-variance dilemma. Neural Computation 4, 1–58.

Goldberg, D. E., 1989. Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley.

Golub, T., Slonim, D., Tamayo, P., Huard, C., Gaasenbeek, M., Mesirov, J., Coller, H., Loh, M., Downing, J., Caliguri, M., Bloomfield, C., Lander, E., 1999. Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. Science 286, 531–537.

Grossman, D., Domingos, P., 2004. Learning Bayesian network classifiers by maximizing conditional likelihood. In: Proceeding of the 21th International Conference on Machine Learning.

Gurwicz, Y., Lerner, B., 2004. Rapid spline-based kernel density estimation for Bayesian networks. In: Proceedings of the 17th International Conference on Pattern Recognition. Vol. 3. pp. 700–703.

Gurwicz, Y., Lerner, B., 2005. Bayesian network classification using spline-aproximated kernel density estimation. Pattern Recognition Letters 26, 1761–1771.

Guyon, I., Elisseeff, A., 2003. An introduction to variable and feature selection. Journal of Machine Learning Research 3, 1157–1182.

Guyon, I., Weston, J., Barnhill, S., Vapnik, V., 2002. Gene selection for cancer classification using support vector machines. Machine Learning 46, 389–422.

Hall, M. A., 1999. Correlation-based Feature Selection for Machine Learning. Ph.D. thesis, The University of Waikato.

Hall, M. A., Smith, L. A., 1997. Feature subset selection: A correlation based filter approach. In: Proceeding of the Fourth International Conference on Neural Information Processing and Intelligent Information Systems. pp. 855–858.

Hamerly, G., Elkan, C. H., 2001. Bayesian approaches to failure prediction for disk drives. In: Proceedings of the Eighteenth International Conference on Machine Learning. pp. 202–209.

Hand, D. J., Yu, K., 2001. Idiot's Bayes - not so stupid after all? International Statistical Review 69 (3), 385–398.

Heckerman, D. E., Geiger, D., Chickering, D., 1995. Learning Bayesian networks: The combination of knowledge and statistical data. Machine Learning 20, 197–243.

Hills, M., 1966. Allocation rules and their error rates. Journal of the Royal Statistical Society Series B 28, 1–31.

Holm, S., 1979. A simple sequentially rejective multiple hypothesis test procedure. Scandinavian Journal of Statistics 6, 65–70.

Horst, P., 1941. Prediction of Personal Adjustment. Vol. Bulletin 48. Social Science Research Council.

Iacobucci, D., 2005. On p-values. Journal of Consumer Research 32 (1), 6–11.

Iman, R. L., Davenport, J. M., 1980. Approximations of the critical region of the Friedman statistic. Communications in Statistics, 571–595.

Inza, I., Larrañaga, P., Blanco, R., Cerrolaza, A. J., 2004. Filter versus wrapper gene selection approaches in DNA microarray domains. Artificial Intelligence in Medicine 31 (2), 91–103.

Inza, I., Larrañaga, P., Etxeberria, R., Sierra, B., 2000. Feature subset selection by Bayesian network-based optimization. Artificial Intelligence 123, 157–184.

Inza, I., Larrañaga, P., Sierra, B., 2001a. Feature subset selection by Bayesian networks: A comparison with genetic and sequential algorithms. International Journal of Approximate Reasoning 27 (2), 143–164.

Inza, I., Merino, M., Larrañaga, P., Quiroga, J., Sierra, B., Girola, M., 2001b. Feature subset selection by genetic algorithms and estimation of distribution algorithms. A case study in the survival of cirrhotic patients treated with TIPS. Artificial Intelligence in Medicine 23 (2), 187–205.

Inza, I., Sierra, B., Blanco, R., Larrañaga, P., 2002. Gene selection by sequential search wrapper approaches in microarray cancer class prediction. Journal of Intelligent and Fuzzy Systems 12 (1), 25–33.

Jakulin, A., 2002. Attribute Interactions in Machine Learning. Ph.D. thesis, University of Ljubljana.

Jakulin, A., Bratko, I., 2004. Quantifying and visualizing attribute interactions: An approach based on entropy. Tech. rep., University of Ljubljana.

James, G. M., 2003. Variance and bias for general loss functions. Machine Learning 51, 115–135.

Jebara, T., 2004. Machine Learning: Discriminative and Generative. Kluwer Academic Publishers.

Jensen, F. V., 1996. An Introduction to Bayesian Networks. UCL Press.
Jensen, F. V., Lauritzen, S. L., 2000. Probabilistic networks. In: Handbook of Defeasible Reasoning and Uncertainty Management Systems. Vol. 5. Kluwer, pp. 289–320.
Jensen, F. V., Nielsen, T. D., 2007. Bayesian Networks and Decision Graphs, second edition Edition. Springer-Verlag.
John, G. H., Kohavi, R., Pfleger, K., 1994. Irrelevant features and the subset selection problem. In: Proceedings of the 13th International Conference on Machine Learning. pp. 121–129.
John, G. H., Langley, P., 1995. Estimating continuous distributions in Bayesian classifiers. In: Proceedings of the 11th Conference on Uncertainty in Artificial Intelligence. pp. 338–345.
Jolliffe, I. T., 1986. Principal Component Analysis. Springer-Verlag.
Kanji, G. K., 2006. 100 statistical tests, 3rd Edition. SAGE Publications Ltd.
Keogh, E. J., Pazzani, M., 1999. Learning augmented Bayesian classifiers: A comparison of distribution-based and non distribution-based approaches. In: Proceedings of the 7th International Workshop on Artificial Intelligence and Statistics. pp. 225–230.
Kiivery, H., Speed, T. P., Carlin, J. B., 1984. Recursive causal models. Journal of the Australian Mathematical Society A 36, 30–52.
Kinja, K., Rendell, L. A., 1992. The feature selection problem: traditional methods and a new algorithm. In: Proceedings of the Ninth National Conference on Artificial Intelligence. pp. 129–134.
Kohavi, R., 1995a. A study of cross-validation and bootstrap for accuracy estimation and model selection. In: Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence. pp. 1137–1145.
Kohavi, R., 1995b. Wrappers for Performance Enhancement and Oblivious Decision Graphs. Ph.D. thesis, Stanford University, Computer Science Department.
Kohavi, R., Becker, B., Sommerfield, D., 1997. Improving simple Bayes. Tech. rep., Data Mining and Visualization Group, Silicon Graphics.
Kohavi, R., John, G., 1997. Wrappers for feature subset selection. Artificial Intelligence 97 (1-2), 273–324.
Kohavi, R., Wolpert, D., 1996. Bias plus variance decomposition for zero-one loss functions. In: Saitta, L. (Ed.), International Conference on Machine Learning. Morgan Kaufmann, pp. 275–283.
Kononenko, I., 1990. Comparison of inductive and naïve Bayesian learning approaches to automatic knowledge acquisition. In: Wielinga, B., Boose, J., Gaines, B., Shereiber, G., van Someren, M. (Eds.), Current Trends in Knowledge Acquisition.
Kononenko, I., 1991. Semi-naive Bayesian classifiers. In: Proceedings of the 6th European Working Session on Learning. pp. 206–219.
Korb, K. B., Nicholson, A. E., 2004. Bayesian Artificial Intelligence. Chapman & Hall / CRC.

Kudo, M., 2000. Comparison of algorithms that select features for pattern classifiers. Machine Learning 33 (1), 25–41.

Kullback, S., 1959. Information Theory and Statistics. Dover Publications, Inc.

Kullback, S., Leibler, R. A., 1951. On information and sufficiency. Annals of Mathematical Statistics 22 (1), 79–86.

Lachenbruch, P., Mickey, M., 1968. Estimation of error rates in discriminant analysis. Technometrics 10, 1–11.

Lam, W., Bacchus, F., 1994. Learning Bayesian belief networks. An approach based on the MDL principle. Computational Intelligence (10), 269–293.

Langley, P., Iba, W., Thompson, K., 1992. An analysis of Bayesian classifiers. In: Proceedings of the 10th National Conference on Artificial Intelligence. pp. 223–228.

Langley, P., Sage, S., 1994. Induction of selective Bayesian classifiers. In: Proceedings of the 10th Conference on Uncertainty in Artificial Intelligence. pp. 399–406.

Larrañaga, P., 2003. Classificación Supervisada via Modelos Gráficos Probabilísticos. Research work for the full professor position. In Spanish.

Larrañaga, P., Lozano, J. A., 2002. Estimation of Distribution Algorithms. A New Tool for Evolutionary Computation. Kluwer Academic Publishers.

Larson, S. C., 1931. The shrinkage of the coefficient of multiple correlation. Journal of Educational Psychology 22, 45–55.

Lauritzen, S. L., 1992. Propagation of probabilities, means and variances in mixed graphical association models. Journal of the American Statistical Association 87 (420), 1098–1108.

Lauritzen, S. L., 1996. Graphical Models. Oxford University Press.

Lauritzen, S. L., David, A. P., Larsen, B. N., Leimer, H. G., 1990. Independence properties of directed Markov fields. Networks 20.

Lauritzen, S. L., Jensen, F., 2001. Stable local computations with conditional gaussian distributions. Statistics and Computing 11.

Lauritzen, S. L., Wermuth, N., 1984. Mixed interaction models. Technical report r 84-8, Institute for Electronic Systems, Aalborg University.

Lauritzen, S. L., Wermuth, N., 1989. Graphical models for associations between variables, some of wich are qualitative and some quantitative. Annals of Statistics 17.

Lerner, B., 2004. Bayesian fluorescence in situ hybridisation signal classification. Artificial Intelligence in Medicine 30 (3), 301–316.

Lerner, B., Lawrence, N. D., 2001. A comparison of state-of-the-art classification techniques with application to cytogenetics. Neural Computing and Applications 10 (1), 39–47.

Liu, H., Motoda, H., 1998. Feature Selection for Knowledge Discovery and Data Mining. Kluwer Academic Publishers.

Liu, H., Motoda, H. (Eds.), 2008. Computational Methods of Feature Selection. Chapman and Hall, CRC Press.

Lozano, J. A., Larrañaga, P., Inza, I., Bengoetxea, E., 2006. Towards a New Evolutionary Computation. Advances in Estimation of Distribution Algorithms. Springer.

Mani, S., Pazzani, M., West, J., 1997. Knowledge discovery from a breast cancer database. In: Lecture Notes in Artificial Intelligence, 1211. pp. 130–133.

McCallum, A., Nigam, K., 1998. A comparison on event models for naïve Bayes text classification. In: 15th American Association for Artificial Intelligence (AAAI) Workshop on Learning for Text Categorization. pp. 41–48.

McLachlan, G. J., Peel, D., 2000. Finite Mixture Models. Probability and Mathematical Statistics. John Wiley and Sons.

Minsky, M., 1961. Steps toward artificial intelligence. Transactions on Institute of Radio Engineers 49, 8–30.

Miyahara, K., Pazzani, M., 2000. Collaborative filtering with the simple Bayesian classifier. In: PRICAI. pp. 679–689.

Moon, Y., Rajagopalan, B., Lall, U., 1995. Estimation of mutual information using kernel density estimators. Physical Review 52 (3), 2318–2321.

Moral, S., Rumí, R., Salmerón, A., 2002. Estimating mixtures of truncated exponential from data. In: First European Workshop on Probabilistic Graphical Models. pp. 156–167.

Mosteller, F., Tukey, J. W., 1968. Data analysis, including statistics. In: Lindzey, G., Aronson, E. (Eds.), Handbook of Social Psycholgy. Vol. 2. Addison-Wesley, Ch. 10, pp. 80–203.

Movellan, J. R., Wachtler, T., Albright, T. D., Sejnowski, T., 2002. Naïve Bayesian coding of color in primary visual cortex. In: Advances on Neural Information Processing Systems 14. pp. 20–40.

Nadeau, C., Bengio, Y., 2003. Inference for the generalization error. Machine Learning 52, 239–281.

Nemenyi, P. B., 1963. Distribution-free Multiple Comparisons. Ph.D. thesis, Princeton University.

Nilsson, R., Peña, J. M., Bjorkegren, J., Tegnér, J., 2007. Consistent feature selection for pattern recognition in polynomial time. Journal of Machine Learning Research 8, 589–612.

Ohmann, C., Moustakis, V., Yang, Q., Lang, K., 1996. Evaluation of automatic knowledge acquisition techniques in the diagnosis of acute abdominal pain. Artificial Intelligence in Medicine 8, 23–36.

Ohmann, C., Yang, Q., Kunneke, M., Stolzing, H., Thon, K., Lorenz, W., 1988. Bayes theorem and conditional dependence of symptoms: Different models applied to data of upper gastrointestinal bleeding. Methods of Information in Medicine 27, 73–83.

Pardo, L., 1997. Teoría de la Información Estadística. Análisis de Datos Categorizables. Hespérides.

Pardo, L., Morales, D., Salicrú, M., Menéndez, M. L., 1993. The ϕ-divergence statistic in bivariate multinomial populations including stratification. Metrika 40, 223–235.

Parzen, E., 1962. On estimation of a probability density function and mode. Annals of Mathematical Statistics 33 (3), 1065–1076.

Pazzani, M., 1997. Searching for dependencies in Bayesian classifiers. In: Learning from Data: Artificial Intelligence and Statistics V. pp. 239–248.

Pazzani, M., Murumatsu, J., Billsus, D., 1996. Syskill and Webert: identifying interesting web sites. In: Proceedings of the 13th National Conference on Artificial Intelligence. pp. 54–61.

Pearl, J., 1988. Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference. Morgan Kaufmann Publishers.

Peña, J. M., 2001. On Unsupervised Learning of Bayesian Networks and Conditional Gaussian Networks. Ph.D. thesis, Department of Computer Science and Artificial Intelligence. University of the Basque Country.

Peña, J. M., Lozano, J. A., Larrañaga, P., 2002. Learning recursive Bayesian multinets for data clustering by means of constructive induction. Machine Learning 47, 63–89.

Pérez, A., Larrañaga, P., Inza, I., 2006a. Information theory and classification error in probabilistic classifiers. In: Proceedings of the Ninth International Conference on Discovery Science. Lecture Notes in Artificial Intelligence. Vol. 4265. pp. 347–351.

Pérez, A., Larrañaga, P., Inza, I., 2006b. Supervised classification with conditional Gaussian networks: Increasing the structure complexity from naïve Bayes. International Journal of Approximate Reasoning 43, 1–25.

Pérez, A., Larrañaga, P., Inza, I., 2009. Bayesian classifiers based on kernel density estimation: Flexible classifiers. International Journal of Approximate Reasoning 50 (2), 341–362.

Pernkopf, F., 2005. Bayesian network classifier versus k-NN classifier. Pattern Recognition 38 (1), 1–10.

Pernkopf, F., Bilmes, J., 2004. Discriminative versus generative parameter and structure learning of Bayesian network classifiers. Tech. rep., Institute of Signal Processing and Speech Communication, Graz University of Technology.

Pernkopf, F., Bilmes, J., 2005. Discriminative versus generative parameter and structure learning of Bayesian network classifiers. In: Proceedings of the 22nd International Conference on Machine Learning.

Quinlan, J. R., 1986. Induction of decision trees. Machine Learning 1, 81–106.

Quinlan, J. R., 1993. C4.5: Programs for Machine Learning. Morgan Kaufmann.

Raudys, S., 1991. On the effectiveness of Parzen window classifier. Informatica 2 (3), 434–453.

Rissanen, J., 1978. Modelling by shortest data description. Automatica (14), 465–471.

Rodríguez, J., Pérez, A., Lozano, J., 2009. Sensitivity analysis of k-fold cross-validation in prediction error estimation. IEEE Transactions on Pattern Analysis and Machine IntelligenceIn press.

Romero, V., Rumí, R., Salmerón, A., 2006. Learning hybrid Bayesian networks using mixture of truncate exponentials. International Journal of Approximate Reasoning 42, 54–68.

Rosenblatt, F., 1956. Remarks on some nonparametric estimates of a density function. Annals of Mathematical Statistics 27, 832–837.

Rosenblatt, F., 1959. Principles of Neurodynamics. Spartan Books.

Saeys, Y., Inza, I., Larrañaga, P., 2007. A review of feature selection techniques in bioinformatics. Bioinformatics 23, 2507–2517.

Sahami, M., 1996. Learning limited dependence Bayesian classifiers. In: Proceedings of the Second International Conference on Knowledge Discovery and Data Mining. pp. 335–338.

Santafé, G., 2007. Advances on Supervised and Unsupervised Learning of Bayesian Network Models. Application to Population Genetics. Ph.D. thesis, University of the Basque Country.

Santafé, G., Lozano, J., Larrañaga, P., 2005. Discriminative learning of Bayesian network classifiers via the TM algorithm. In: Proceedings of the Eighth European Conference on Symbolic and Quantitative Approaches to Reasoning with Uncertainty. pp. 148–160.

Scott, D. W., 1992. Multivariate Density Estimation: Theory, Practice and Visualization. John Wiley and Sons.

Scott, D. W., Szewczyk, W. F., 2001. From kernels to mixtures. Technometrics 43 (3), 323–335.

Shachter, R. D., Kenley, C. R., 1989. Gaussian influence diagrams. Management Science 535, 527–550.

Shaffer, J. P., 1986. Modified sequentially rejective multiple test procedures. Journal of the American Statistical Association 81 (395), 826–831.

Shaffer, J. P., 1995. Multiple hypothesis testing. Annual Review of Psychology 46, 561–584.

Shanon, C. E., 1948. A mathematical theory of communication. The Bell System Technical Journal 27 (3), 379–423 and 623–656.

Sheskin, D., 2003. Handbook of Parametric and Nonparametric Statistical Procedures. Chapman and Hall/CRC.

Silverman, B. W., 1986. Density Estimation for Statistics and Data Analysis. Chapman and Hall: London.

Sima, C., Attor, S., Braga-Neto, U., Lowey, J., Suh, E., Dougherty, E. R., 2005a. Impact of error estimation on feature selection. Pattern Recognition 38, 2472–2482.

Sima, C., Braga-Neto, U., Dougherty, E. R., 2005b. Superior feature-set ranking for small samples using bolstered error estimation. Bioinformatics 21 (7), 1046–1054.

Simonoff, J. S., 1996. Smoothing Methods in Statistics. Springer.

Stone, M., 1974. Cross-validatory choice and assessment of statistical predictions. Journal of the Royal Statistical Society Series B 36 (1), 111–147.

Titterington, D. M., Smith, A. F. M., Makov, U. E., 1985. Statistical Analysis of Finite Mixture Fistributions. John Wiley and Sons.

Todd, B. S., Stamper, R., 1994. The relative accuracy of a variety of medical diagnostic programs. Methods of Information in Medicine 33, 402–416.

van der Putten, P., van Someren, M., 2004. A bias-variance analysis of a real world learning problem: The CoIL challenge 2000. Machine Learning 57, 177–195.

Vapnik, V., 1998. Statistical Learning Theory. John Wiley and Sons.

Vinciotti, V., T. A. K. P., Liu, X., 2006. The robust selection of predictive genes via a simple classifier. Applied Bioinformatics 5 (1), 1–11.

Wand, M. P., Jones, M. C., 1995. Kernel Smoothing. Monographs on Statistics and Applied Probability. Chapman and Hall.

Wang, H., 1996. Towards a Unified Framework of Relevance. Ph.D. thesis, Faculty of Informatics, University of Ulster.

Wang, H., Bell, D., Murtagh, F., 1999. Axiomatic approach to feature subset selection based on relevance. IEEE Transactions on Pattern Analysis and Machine Intelligence 21 (3), 271–277.

Wermuth, N., 1976. Analogies between multiplicative models in contingency tables and covariance selection. Biometriks 32, 95–108.

Whittaker, J., 1990. Graphical Models in Applied Multivariate Statistics. John Wiley and Sons.

Wilcoxon, F., 1945. Individual comparisons by ranking methods. Biometrics 1, 80–83.

Witten, I. H., Frank, E., 2005. Data Mining: Practical Machine Learning Tools and Techniques, 2nd Edition. Morgan Kaufmann.

Yang, Y., 2003. Discretization for naïve-bayes learning. Ph.D. thesis, School of Computer Science and Software Engineering of Monash University.

Yang, Y., Webb, G. I., 2003. Discretization for naive-Bayes learning: Managing discretization bias and variance. Technical report 2003-131, School of Computer Science and Software Engineering, Monash University.

Yu, L., Liu, H., 2004. Efficient feature selection via analysis of relevance and redundancy. Machine Learning Research 5, 1205–1224.

Zheng, Z., Webb, G., Ting, K. M., 1999. Lazy Bayesian rules: A lazy semi-naïve Bayesian learning technique competitive to boosting decision trees. In: Proceedings of the 16th International Conference on Machine Learning. pp. 493–502.

Zhou, A., Cai, Z., Wei, L., 2003. M-kernel merging: Towards density estimation over data streams. In: Proceedings of the Database Systems for Advanced Applications. pp. 285–292.