ELSEVIER

Artificial Intelligence 123 (2000) 157–184

Artificial Intelligence

www.elsevier.com/locate/artint

Feature Subset Selection by Bayesian network-based optimization

I. Inza *, P. Larrañaga, R. Etxeberria, B. Sierra

Department of Computer Science and Artificial Intelligence, University of the Basque Country, P.O. Box 649, E-20080 Donostia – San Sebastián, Basque Country, Spain

Received 12 December 1999

Abstract

A new method for Feature Subset Selection in machine learning, FSS-EBNA (Feature Subset Selection by Estimation of Bayesian Network Algorithm), is presented. FSS-EBNA is an evolutionary, population-based, randomized search algorithm, and it can be executed when domain knowledge is not available. A wrapper approach, over Naive-Bayes and ID3 learning algorithms, is used to evaluate the goodness of each visited solution. FSS-EBNA, based on the EDA (Estimation of Distribution Algorithm) paradigm, avoids the use of crossover and mutation operators to evolve the populations, in contrast to Genetic Algorithms. In absence of these operators, the evolution is guaranteed by the factorization of the probability distribution of the best solutions found in a generation of the search. This factorization is carried out by means of Bayesian networks. Promising results are achieved in a variety of tasks where domain knowledge is not available. The paper explains the main ideas of Feature Subset Selection, Estimation of Distribution Algorithm and Bayesian networks, presenting related work about each concept. A study about the 'overfitting' problem in the Feature Subset Selection process is carried out, obtaining a basis to define the stopping criteria of the new algorithm.

Keywords: Machine learning; Supervised learning; Feature Subset Selection; Wrapper; Predictive accuracy; Estimation of Distribution Algorithm; Estimation of Bayesian Network Algorithm; Bayesian network; Overfitting

1. Introduction

In supervised machine learning, the goal of a supervised learning algorithm is to induce a classifier that allows us to classify new examples $E* = \{e_{n+1}, \ldots, e_{n+m}\}$ that are only

* Corresponding author.
E-mail address: ccbincai@si.ehu.es (I. Inza).

0004-3702/00/$ – see front matter
PII: S0004-3702(00)00052-7

characterized by their d descriptive features. To generate this classifier we have a set of n samples $E = \{e_1, \ldots, e_n\}$, characterized by d descriptive features $X = \{X_1, \ldots, X_d\}$ and the class label $C = \{w_1, \ldots, w_n\}$ to which they belong. Machine learning can be seen as a 'data-driven' process where, putting less emphasis on prior hypotheses than is the case with classical statistics, a 'general rule' is induced for classifying new examples using a learning algorithm. Many representations with different biases have been used to develop this 'classification rule'. Here, the machine learning community has formulated the following question: *"Are all of these d descriptive features useful for learning the 'classification rule'?"* Trying to respond to this question the Feature Subset Selection (FSS) approach appears, which can be reformulated as follows: *given a set of candidate features, select the best subset under some learning algorithm*.

This dimensionality reduction made by an FSS process can carry out several advantages for a classification system in a specific task:

- a reduction in the cost of acquisition of the data,
- improvement of the comprensibility of the final classification model,
- a faster induction of the final classification model,
- an improvement in classification accuracy.

The attainment of higher classification accuracies is the usual objective of machine learning processes. It has been long proved that the classification accuracy of machine learning algorithms is not monotonic with respect to the addition of features. Irrelevant or redundant features, depending on the specific characteristics of the learning algorithm, may degrade the predictive accuracy of the classification model. In our work, the FSS objective will be the maximization of the performance of the classification algorithm. In addition, with the reduction in the number of features, it is more likely that the final classifier is less complex and more understandable by humans.

Once the objective is fixed, FSS can be viewed as a search problem, with each state in the search space specifying a subset of the possible features of the task. Exhaustive evaluation of possible feature subsets is usually unfeasible in practice because of the large amount of computational effort required. Many search techniques have been proposed to solve the FSS problem when there is no knowledge about the nature of the task, carrying out an intelligent search in the space of possible solutions. As randomized, evolutionary and population-based search algorithm, Genetic Algorithms (GAs) have long been used as the search engine in the FSS process. GAs need crossover and mutation operators to make the evolution possible. However, the optimal selection of crossover and mutation rates is an open problem in the field of GAs [33] and they are normally fixed by means of experimentation.

In this work, a new search engine, Estimation of Bayesian Network Algorithm (EBNA) [29], inspired in the Estimation of Distribution Algorithm paradigm (EDA), will be used for FSS, resulting in the new FSS-EBNA algorithm. FSS-EBNA shares the basic characteristics with GAs, with the attractive property of avoiding crossover and mutation operators. In the new FSS algorithm the evolution is based on the probabilistic modeling by Bayesian networks of promising solutions of each generation to guide further exploration of the space of features.

The work is organized as follows: the next section introduces the FSS concept and its components. Section 3 introduces the EDA paradigm, Bayesian networks and the EBNA

search algorithm. Section 4 presents the details of the new algorithm for Feature Subset Selection, FSS-EBNA. Section 5 presents the datafiles and learning algorithms used to test the new approach and the corresponding results appear in the sixth section. We conclude with a summary and future work.

2. Feature Subset Selection as a search problem

Even if our work is located in machine learning, the FSS literature includes plenty of works in other fields such as pattern recognition (Jain and Chandrasekaran [39], Stearns [83], Kittler [43], Ferri et al. [31]), statistics (Narendra and Fukunaga [69], Boyce et al. [13], Miller [62]), data mining (Chen et al. [20], Provost and Kolluri [75]) or text learning (Mladenić [63], Yang and Pedersen [89]). In this way, different communities have exchanged and shared ideas among them to deal with the FSS problem.

As reported by Aha and Bankert [2], the objective of Feature Subset Selection in machine learning is to *reduce the number of features used to characterize a dataset so as to improve a learning algorithm's performance on a given task*. Our objective will be the maximization of the classification accuracy in a specific task for a certain learning algorithm; as a co-lateral effect we will have the reduction in the number of features to induce the final classification model. The feature selection task can be exposed as a search problem, each state in the search space identifying a subset of possible features. A partial ordering on this space, with each child having exactly one more feature than its parents, can be stated.

Fig. 1 expresses the search algorithm nature of the FSS process. Blum and Langley [10] argue that the structure of this space suggest that any feature selection method must take a stance on the next four basic issues that determine the nature of the search process: a starting point in the search space, an organization of the search, an evaluation strategy of the feature subsets and a criterion for halting the search.

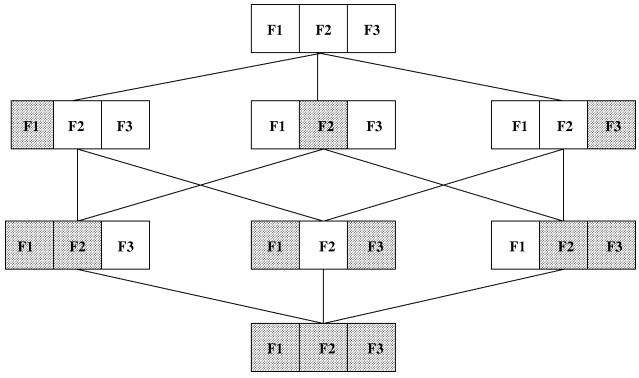

Fig. 1. In this 3-feature (F1, F2, F3) problem, each individual in the space is related with a feature subset, possible solution for the FSS problem. In each individual, when a feature's rectangle is filled, it indicates that it is included in the feature subset.

2.1. Starting point in the space

The starting point in the space determines the direction of the search. One might start with no features and successively add them, or one might start with all the features and successively remove them. One might also select an initial state somewhere in the middle of the search space.

2.2. Organization of the search

The organization of the search determines the strategy of the search in a space of size 2^d, where d is the number of features in the problem. Roughly speaking, the search strategies can be *optimal* or *heuristic*. Two classic optimal search algorithms which exhaustively evaluate all possible subsets are depth-first and breadth-first (Liu and Motoda [58]). Otherwise, Branch & Bound search (Narendra and Fukunaga [69]) guarantees the detection of the optimal subset for monotonic evaluation functions without the systematic examination of all subsets.

When monotonicity cannot be satisfied, in a search space with a 2^d cardinality, depending in the evaluation function used, an exhaustive search can be impractical. Can we make some smart choices based on the information available about the search space, but without looking it on the whole? Here appears the *heuristic* search concept. They find near optimal solution, if not optimal. Among heuristic algorithms, there are *deterministic heuristic* and *non-deterministic heuristic* algorithms. Classic *deterministic heuristic* FSS algorithms are sequential forward and backward selections (SFS and SBS, Kittler [43]), floating selection methods (SFFS and SFBS, Pudil et al. [76]) or best-first search (Kohavi and John [47]). They are deterministic in the sense that all the runs always obtain the same solution. Vafaie and De Jong [86] results suggest that classic greedy hill-climbing approaches, tend to get trapped on local peaks caused by interdependencies among features. In this sense the work of Pudil et al. [76] is an interesting idea in an attempt to avoid this phenomenon. *Non-deterministic heuristic* search appears in a motivation to avoid getting stuck in local maximum. Randomness is used to escape from local maximum and this implies that one should not expect the same solution from different runs. Up until now, the next non-deterministic search engines have been used in FSS: Genetic Algorithms [30,51,81,86,88], Simulated Annealing [27], Las Vegas Algorithm [57,82] (see Liu and Motoda [58] or Jain and Zongker [40] for other kinds of classifications of FSS search algorithms).

2.3. Evaluation function

The evaluation function measures the effectiveness of a particular subset of features after the search algorithm has chosen it for examination. Being the objective of the search its maximization, the search algorithm utilizes the value returned by the evaluation function to help guide the search. Many measures carry out this objective regarding only the characteristics of the data, capturing the relevance of each feature or set of features to define the target concept. As reported by John et al. [41], when the goal of FSS is the maximization of the accuracy, the features selected should depend not only on

the features and the target concept to be learned, but also on the learning algorithm. Kohavi and John [47] report domains in which a feature, although being in the target concept to be learned, does not appear in the optimal feature subset that maximizes the predictive accuracy for the specific learning algorithm used. This occurs due to the intrinsic characteristics and limitations of the classifier used: feature relevance and accuracy optimality are not always coupled in FSS. The idea of using the error reported by a classifier as the feature subset evaluation criterion appears in many previous works done, for example, by Stearns [83] in 1976 or Siedelecky and Skalansky [81] in 1988. Doak [27] in 1992 used the classification error rate to guide non-large searches. In John et al.'s [41] the *wrapper* concept definitively appears. This implies that the FSS algorithm conducts a search for a good subset of features using the induction algorithm itself as a part of the evaluation function, the same algorithm that will be used to induce the final classification model. Once the classification algorithm is fixed, the idea is to train it with the feature subset encountered by the search algorithm, estimating the error percentage, and assigning it as the value of the evaluation function of the feature subset. In this way, representational biases of the induction algorithm used to construct the final classifier are included in the FSS process. Wrapper strategy needs a high computational cost, but technical computer advances in the last decade have made the use of this wrapper approach possible, calculating an amount of accuracy estimations (training and testing on significant amount of data) not envisioned in the 1980s.

Before applying the wrapper approach, an enumeration of the available computer resources is critical. Two different factors become an FSS problem 'large' (Liu and Setiono [59]): the number of features and the number of instances. One must bear in mind the time needed for the learning algorithm used in the wrapper scheme as a training phase is required for every possible solution visited by the FSS search engine. Many approaches have been proposed in literature to alleviate the loading of the training phase, such as Caruana and Freitag [17] (avoiding the evaluation of many subsets taking advantage of the intrinsic properties of the used learning algorithm) or Moore and Lee [64] (reducing the burden of the cross-validation technique for model selection).

When the learning algorithm is not used in the evaluation function, the goodness of a feature subset can be assessed regarding only the intrinsic properties of the data. The learning algorithm only appears in the final part of the FSS process to construct the final classifier using the set of selected features. The statistics literature proposes many measures for evaluating the goodness of a candidate feature subset (see Ben-Bassat [9] for a review of these classic measures). These statistical measures try to detect the feature subsets with higher discriminatory information with respect to the class (Kittler [43]) regarding the probability distribution of data. These measures are usually monotonic and increase with the addition of features that afterwards can hurt the accuracy classification of the final classifier. In pattern recognition FSS works, in order to recognize the forms of the task, it is so common to fix a positive integer number d and select the best feature subset of d cardinality found during the search. When this d parameter is not fixed a examination of the slope of the curve—value of the proposed statistical measure versus cardinality of the selected feature subset—of the best feature subsets is required to select the cardinality of the final feature subset. In text-learning, its predictive accuracy will be assessed running the classifier only with the selected features (Doak [27]). This type of FSS

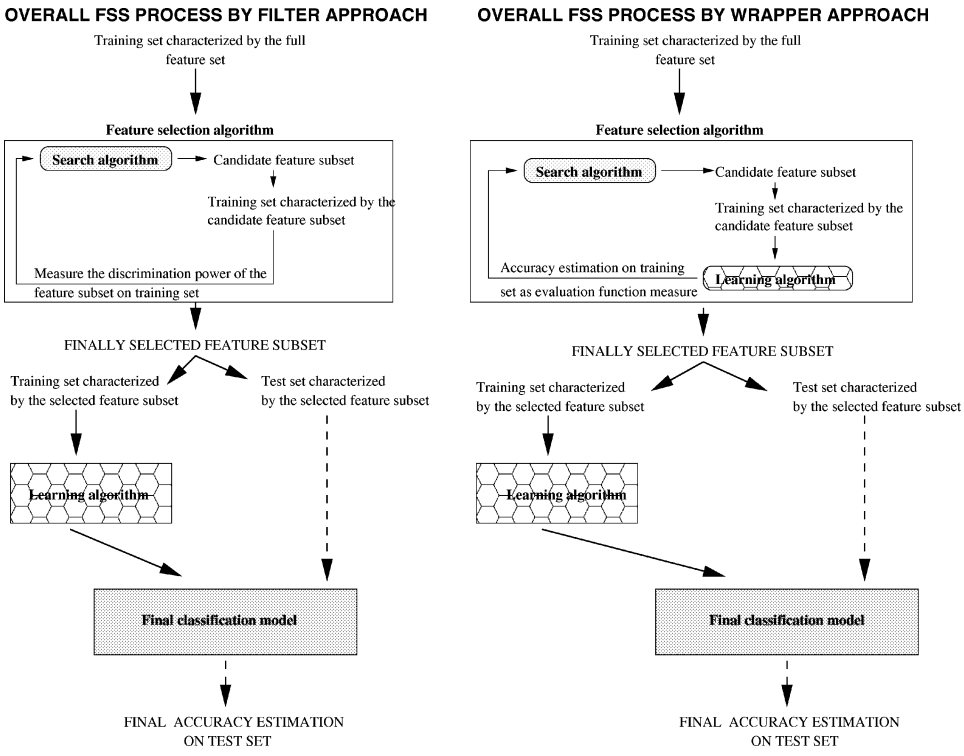

Fig. 2. Summarization of the whole FSS process for filter and wrapper approaches.

approach, which ignores the induction algorithm to assess the merits of a feature subset, is known as *filter* approach. Mainly inspired on these statistical measures, in the 1990s, more complex filter measures which do not use the final induction algorithm in the evaluation function generate new FSS algorithms, such as FOCUS (Almuallin and Dietterich [4]), RELIEF (Kira and Rendell [42]), Cardie's algorithm [16], Koller and Sahami's work with probabilistic concepts [50] or the named 'Incremental Feature Selection' (Liu and Setiono [59]). Nowadays, the filter approach is receiving considerable attention from the 'data mining' community to deal with huge databases when the wrapper approach is unfeasible (Liu and Motoda [58]). Fig. 2 locates the role of filter and wrapper approaches within the overall FSS process.

When the size of the problem allows the application of the wrapper approach, works in the 1990s have noted its superiority, in terms of predictive accuracy over the filter approach. Doak [27] in the early 1990s, also empirically showed this superiority of the wrapper model, but due to computational availability limitations, he could only apply Sequential Feature Selection with the wrapper model, discarding the use of computationally more expensive global search engines (Best-First, Genetic Algorithms, etc.) in his comparative work between FSS algorithms.

Blum and Langley [10] also present another type of FSS, known as *embedded*. This concept covers the feature selection process made by the induction algorithm itself. For

example, both partitioning and separate-and-conquer methods implicitly select features for inclusion in a branch or rule in preference to other features that appear less relevant, and in the final model some of the initial features might not appear. On the other hand, some induction algorithms (i.e., Naive-Bayes [19] or IB1 [1]) include all the presented features in the model when no FSS is executed. This FSS approach is done within the learning algorithm preferring some features instead of others and possibly not including all the available features in the final classification model induced by the learning algorithm. However, filter and wrapper approaches are located one abstraction level above the embedded approach, performing a feature selection process for the final classifier apart from the embedded selection done by the learning algorithm itself.

2.4. Criterion for halting the search

An intuitive approach for stopping the search will be the non-improvement of the evaluation function value of alternative subsets. Another classic criterion will be to fix an amount of possible solutions to be visited along the search.

3. EDA paradigm, Bayesian networks and EBNA approach

In this section, EDA paradigm and Bayesian networks will be explained. Bearing in mind these two concepts, EBNA, the search engine used in our FSS algorithm will be presented. EDA paradigm is the general formula of the EBNA algorithm and Bayesian networks can be seen as the most important basis of EBNA.

3.1. EDA paradigm

Genetic Algorithms (GAs, see Holland [37]) are one of the best known techniques for solving optimization problems. Their use has reported promising results in many areas but there are still some problems where GAs fail. These problems, known as deceptive problems, have attracted the attention of many researchers and as a consequence there has been growing interest in adapting the GAs in order to overcome their weaknesses.

The GA is a population-based search method. First, a set of individuals (or candidate solutions to our optimization problem) is generated (a population), then promising individuals are selected, and finally new individuals which will form the new population are generated using crossover and mutation operators.

An interesting adaptation of this is the Estimation of Distribution Algorithm (EDA) [65] (see Fig. 3). In EDA, there are neither crossover nor mutation operators, the new population is sampled from a probability distribution which is estimated from the selected individuals.

In this way, a randomized, evolutionary, population-based search can be performed using probabilistic information to guide the search. It is shown that although EDA approach process solutions in a different way to GAs, it has been empirically proven that the results of both approaches can be very similar (Pelikan et al. [74]). In this way, both approaches do the same except that EDA replaces genetic crossover and mutation operators by means of the following two steps:

EDA
$D_0 \leftarrow$ Generate N individuals (the initial population) randomly.
Repeat for $l = 1, 2, \ldots$ until a stop criterion is met.
$D^s_{l-1} \leftarrow$ Select $S \leqslant N$ individuals from D_{l-1} according to a selection method.
$p_l(\boldsymbol{x}) = p(\boldsymbol{x}|D^s_{l-1}) \leftarrow$ Estimate the joint probability distribution of an individual being among the selected individuals.
$D_l \leftarrow$ Sample N individuals (the new population) from $p_l(\boldsymbol{x})$.

Fig. 3. Main scheme of the EDA approach.

(1) a probabilistic model of selected promising solutions is induced,
(2) new solutions are generated according to the induced model.

The main problem of EDA resides on how the probability distribution $p_l(\boldsymbol{x})$ is estimated. Obviously, the computation of 2^n probabilities (for a domain with n binary variables) is impractical. This has led to several approximations where the probability distribution is assumed to factorize according to a probability model (see Larrañaga et al. [55] or Pelikan et al. [74] for a review).

The simplest way to estimate the distribution of good solutions assumes the independence between the features [1] of the domain. New candidate solutions are sampled by only regarding the proportions of the values [2] of all features independently to the remaining solutions. Population-Based Incremental Learning (PBIL, Baluja [7]), Compact Genetic Algorithm (cGA, Harik et al. [34]), Univariate Marginal Distribution Algorithm (UMDA, Mühlenbein [66]) and Bit-Based Simulated Crossover (BSC, Syswerda [84]) are four algorithms of this type. They have worked well under artificial tasks with no significant interactions among features and so, the need for covering higher order interactions among the variables is seen for more complex or real tasks.

Efforts covering pairwise interactions among the features of the problem have generated algorithms such as population-based MIMIC algorithm using simple chain distributions (De Bonet et al. [25]), the so-called dependency trees (Baluja and Davies [8]) and Bivariate Marginal Distribution Algorithm (BMDA, Pelikan and Mühlenbein [72]). Pelikan and Mühlenbein [72] have demonstrated that only covering pairwise dependencies is not enough with problems which have higher order interactions.

In this way, the Factorized Distribution Algorithm (FDA, Mühlenbein et al. [67]) covers higher order interactions. This is done using a previously fixed factorization of the joint probability distribution. However, FDA needs prior information about the decomposition and factorization of the problem which is often not available.

Without the need of this extra information about the decomposition and factorization of the problem, Bayesian networks are graphical representations which cover higher order interactions. EBNA (Etxeberria and Larrañaga [29]) and BOA (Pelikan et al. [73]) are algorithms which use Bayesian networks for estimating the joint distribution of promising solutions. In this way multivariate interactions among problem variables can be covered.

[1] In the Evolutionary Computation community, the term 'variable' is normally used instead of 'feature'. We use both terms indistinctly.

[2] In the FSS problem there are two values for each candidate solution: '0' denoting the absence of the feature and '1' denoting its presence.

Based on the EBNA work of Etxeberria and Larrañaga [29], we propose the use of Bayesian networks as the models for representing the probability distribution of a set of candidate solutions in our FSS problem, using the application of automatic learning methods to induce the right distribution model in each generation in an efficient way.

3.2. Bayesian networks

3.2.1. Definition

A Bayesian network (Castillo [18], Lauritzen [56], Pearl [71]) encodes the relationships contained in the modelled data. It can be used to describe the data as well as to generate new instances of the variables with similar properties as those of given data. A Bayesian network encodes the probability distribution $p(\boldsymbol{x})$, where $\boldsymbol{x} = (X_1, \ldots, X_d)$ is a vector of variables, and it can be seen as a pair (M, θ). M is a directed acyclic graph (DAG) where the nodes correspond to the variables and the arcs represent the conditional (in)dependencies among the variables. By X_i, both the variable and the node corresponding to this variable is denoted. M will give the factorization of $p(\boldsymbol{x})$:

$$p(\boldsymbol{x}) = \prod_{i=1}^{d} p(x_i|\pi_i),$$

where Π_i is the set of parent variables (nodes) that X_i has in M and π_i its possible instantiations. The number of states of Π_i will be denoted as $|\Pi_i| = q_i$ and the number of different values of X_i as $|X_i| = r_i$. $\theta = \{\theta_{ijk}\}$ are the required conditional probability values to completely define the joint probability distribution of X. θ_{ijk} will represent the probability of X_i being in its kth state while Π_i is in its jth instantiation. This factorization of the joint distribution can be used to generate new instances using the conditional probabilities in a modelled dataset.

Informally, an arc between two nodes relates the two nodes so that the value of the variable corresponding to the ending node of the arc depends on the value of the variable corresponding to the starting node. Every probability distribution can be defined by a Bayesian network. As a result, Bayesian networks are widely used in problems where uncertainty is handled using probabilities.

Two following sections relate the learning process of Bayesian networks from data and the generation of new instances from the Bayesian networks.

3.2.2. Learning Bayesian networks from data

The key step of any EDA is the estimation of the probability distribution $p(\boldsymbol{x}|D_{l-1}^s)$. Depending on how it is estimated, the results of the algorithm will vary. In this section, we will show how this can be done automatically using Bayesian networks. Selected individuals will be treated as data cases which form a data set D_{l-1}^s. Our goal will be to set a method which, in each generation, obtains $p(\boldsymbol{x}|D_{l-1}^s)$ as fast as possible in a multiple connected form.

Let D be a data set of S selected cases and $p(\boldsymbol{x}|D)$ the probability distribution we want to find. If we represent as $\mathcal{M}$ the possible DAGs, then from probability theory we obtain:

$$p(\boldsymbol{x}|D) = \sum_{M \in \mathcal{M}} p(\boldsymbol{x}|M, D)\, p(M|D).$$

This equation is known as *Bayesian model averaging* (Madigan et al. [60]). As it requires the summing of all possible structures, its use is unfeasible and usually two approximations are used instead of the afore mentioned approach.

The first is known as *selective model averaging*, where only a reduced number of promising structures is taken into account. In this case, denoting the set of this promising structures by $\mathcal{M}^{\mathcal{S}}$, we have:

$$p(\boldsymbol{x}|D) \approx \sum_{M \in \mathcal{M}^{\mathcal{S}}} p(\boldsymbol{x}|M, D)\, p(M|D),$$

where

$$\sum_{M \in \mathcal{M}^{\mathcal{S}}} p(M|D) \approx 1.$$

The second approximation is known as *model selection* where $p(\boldsymbol{x}|D)$ is approximated in the following manner:

$$p(\boldsymbol{x}|D) \approx p(\boldsymbol{x}|\widehat{M}, D), \tag{1}$$

where

$$\widehat{M} = \arg \max_{M \in \mathcal{M}^{\mathcal{S}}} p(M|D).$$

This means that only the structure with the maximum posterior likelihood is used, knowing that for large enough D, we have $p(\widehat{M}|D) \approx 1$.

Obviously, better results must be obtained using model averaging, but due to its easier application and lower cost model selection, it is preferred most of the times. In our case, we will also use the second approximation, remembering that the estimation of $p(\boldsymbol{x}|D)$ must be done quickly. In Heckerman et al. [35] it is shown that under some assumptions, for any structure M:

$$p(\boldsymbol{x}|M, D) = \prod_{i=1}^{d} E[\theta_{ijk}|M, D], \tag{2}$$

where $E[\theta_{ijk}|M, D]$ is the expected probability of the variable X_i being in its kth state when its parent nodes in M, Π_i, are in their jth configuration.

Furthermore, in Cooper and Herskovits [23] it is shown that:

$$E[\theta_{ijk}|M, D] = \frac{N_{ijk} + 1}{N_{ij} + r_i}. \tag{3}$$

N_{ijk} is the number of times that X_i is in its kth state and Π_i in its jth configuration in D. $N_{ij} = \sum_k N_{ijk}$.

Combining (1), (2) and (3), we obtain:

$$p(\boldsymbol{x}|D) \approx p(\boldsymbol{x}|\widehat{M}, D) = \prod_{i=1}^{d} \frac{N_{ijk} + 1}{N_{ij} + r_i} = \prod_{i=1}^{d} p(x_i|\pi_i)$$

which allows us to represent the probability distribution $p(\boldsymbol{x}|D)$ using a Bayesian network whose structure has the maximum posterior likelihood, and whose parameters can be computed directly from the data set.

But to get things working, we must be able to find $\widehat{M}$, we must be able to learn it from the data. $\widehat{M}$ is the structure with the maximum posterior likelihood. From Bayes theorem:

$$p(M|D) \propto p(D|M)p(M).$$

$p(M)$ is the prior probability of the structures. In our case, we know nothing about these structures, so we will set in a uniform way. Thus,

$$p(M|D) \propto p(D|M).$$

Therefore, finding the structure with the maximum posterior likelihood becomes equivalent to finding the structure which maximizes the probability of the data. Under some assumptions, it has been proved that $p(D|M)$ can be calculated in closed form (Cooper and Herskovits [23], Heckerman et al. [35]); however, in our case we will use the BIC approximation (Schwarz [80]) because being asymptotically equivalent, it has the appealing property of selecting simple structures (Bouckaert [12]), which reduces the computation cost:

$$\log p(D|M) \approx BIC(M, D) = \sum_{i=1}^{n}\sum_{j=1}^{q_i}\sum_{k=1}^{r_i} N_{ijk} \log \frac{N_{ijk}}{N_{ij}} - \frac{\log N}{2}\sum_{i}(r_i - 1)q_i,$$

where N_{ijk} and N_{ij} and q_i are defined as before.

Unfortunately, finding $\widehat{M}$ requires searching through all possible structures, which has been proven to be NP-hard (Chickering et al. [21]). Although promising results have been obtained using global search techniques (Larrañaga et al. [53,54], Etxeberria et al. [28], Chickering et al. [22], Wong et al. [87]) their computation cost makes them unfeasible for our problem. We need to find $\widehat{M}$ as fast as possible, so a simple algorithm which returns a good structure, even if it is not optimal, will be preferred.

In our implementation, Algorithm B (Buntine [14]) is used for learning Bayesian networks from data. Algorithm B is a greedy search heuristic. The algorithm starts with an arc-less structure and at each step, it adds the arc with the maximum increase in the BIC approximation (or whatever measure is used). The algorithm stops when adding an arc does not increase the utilized measure.

3.2.3. Sampling from Bayesian networks

Once we have represented the desired probability distribution using a Bayesian network, new individuals must be generated using the joint probability distribution encoded by the network. These individuals can be generated by sampling them directly from the Bayesian network, for instance, using the Probabilistic Logic Sampling algorithm (PLS, Henrion [36]).

PLS (see Fig. 4) takes advantage of how a Bayesian network defines a probability distribution. It generates the values for the variables following their ancestral ordering which guarantees that $\Pi_{\mu(i)}$ will be instantiated every time. This makes generating values from $p(X_{\mu(i)}|\pi_{\mu(i)})$ trivial.

3.2.4. Estimation of Bayesian Network Algorithm: EBNA

The general procedure of EBNA appears in Fig. 5. To understand the steps of the algorithm, the following concepts must be clarified:

PLS
  Find an ancestral ordering of the nodes in the Bayesian network (μ)
  For $i = 1, 2, \ldots, n$
    $x_{\mu(i)} \leftarrow$ generate a value from $p(x_{\mu(i)}|\pi_{\mu(i)})$

Fig. 4. Probabilistic Logic Sampling scheme.

EBNA
  $BN_0 \leftarrow (\widehat{M}_0, \theta_0)$.
  $D_0 \leftarrow$ Sample N individuals from BN_0.
  For $l = 1, 2, \ldots$ until a stop criterion is met
    $D^s_{l-1} \leftarrow$ Select S individuals from D_{l-1}.
    $\widehat{M}_l \leftarrow$ Find the structure which maximizes $BIC(M_l, D^s_{l-1})$.
    $\theta_l \leftarrow$ Calculate $\{\theta_{ijk} = \frac{N_{ijk}+1}{N_{ij}+r_i}\}$ using D^s_{l-1} as the data set.
    $BN_l \leftarrow (\widehat{M}_l, \theta_l)$.
    $D_l \leftarrow$ Sample N individuals from BN_l using PLS.

Fig. 5. EBNA basic scheme.

$\widehat{M}_0$ is the DAG with no arcs at all and $\theta_0 = \{\forall i\colon\ p(X_i = x_i) = 1/r_i\}$, which means that the initial Bayesian network BN_0 assigns the same probability to all individuals. N is the number of individuals in the population. S is the number of individuals selected from the population. Although S can be any value, we take the suggestion that appears in Etxeberria and Larrañaga [29] into consideration, being $S = N/2$. If S is close to N then the populations will not evolve very much from generation to generation. On the other hand, a low S value will lead to low diversity resulting in early convergence.

In the previous section we have shown how individuals are created from Bayesian networks and how Bayesian networks can estimate the probability distribution of the selected individuals but so far nothing has been said about how individuals are selected or when the algorithm is stopped.

For individual selection range based selection is proposed, i.e., selecting the best $N/2$ individuals from the N individuals of the population. However, any selection method could be used.

For stopping the algorithm different criteria can be used:

- fixing a number of generations,
- when all the individuals of the population are the same,
- when the average evaluation function value of the individuals in the population does not improve in a fixed number of generations,
- when any sampled individual from the Bayesian network does not have a better evaluation function value than the best individual of the previous generation.

A variation of the last criterion will be used, depending on the dimensionality (number of features) of the problem. This concept will be explained in the next section.

Finally, the way in which the new population is created must be pointed out. In the given procedure, all individuals from the previous population are discarded and the new population is composed of all newly created individuals. This has the problem of losing the

best individuals that have been previously generated, therefore, the following minor change has been made: instead of discarding all the individuals, we maintain the best individual of the previous generation and create $N-1$ new individuals.

An elitist approach has been used to form iterative populations. Instead of directly discarding the $N-1$ individuals from the previous generation replacing them with $N-1$ newly generated ones, the $2N-2$ individuals are put together and the best $N-1$ taken among them. These best $N-1$ individuals will form the new population together with the best individual of the previous generation. In this way, the populations converge faster to the best individuals found; however, this also implies a risk of losing diversity within the population.

4. Feature Subset Selection by Estimation of Bayesian Network Algorithm: FSS-EBNA

We will explain the proposed FSS-EBNA method, presenting its different pieces. First, the connection between the EBNA search algorithm and the FSS problem will be clarified. In the second subsection, the evaluation function of the FSS process will be explained. In a third subsection, several considerations about the final evaluation process and the stopping criterion of FSS-EBNA will be presented, coupled with a reflection on the 'overfitting' risk in FSS-EBNA.

4.1. FSS and EBNA connection and the search space

Once the FSS problem and EBNA algorithm are presented, we will use the search engine provided by EBNA to solve the FSS problem. FSS-EBNA, as a search algorithm, will seek in the feature subset space for the 'best' feature subset. Being an individual in the search space a possible feature subset, a common notation will be used to represent each individual: for a full d feature problem, there are d bits in each state, each bit indicating whether a feature is present (1) or absent (0). In each generation of the search, the induced Bayesian network will factorize the probability distribution of selected individuals. The Bayesian network will be formed by d nodes, each one representing a feature of the domain. Each node has two possible values or states (0: absence of the feature; 1: presence of the feature).

Bearing the general EBNA procedure in mind, Fig. 6 summarizes the FSS-EBNA method.

FSS-EBNA is an evolutionary, population-based, randomized search algorithm, and it can be executed when domain knowledge is not available. Although GAs share these characteristics, they need crossover and mutation operators to evolve the population of solutions. Otherwise, FSS-EBNA does not need these operators and must only fix a population size (N) and a size for the selection set (S). We have selected the following numbers:

- as explained in the former section, $S = N/2$ is used,
- S is fixed to 1000.

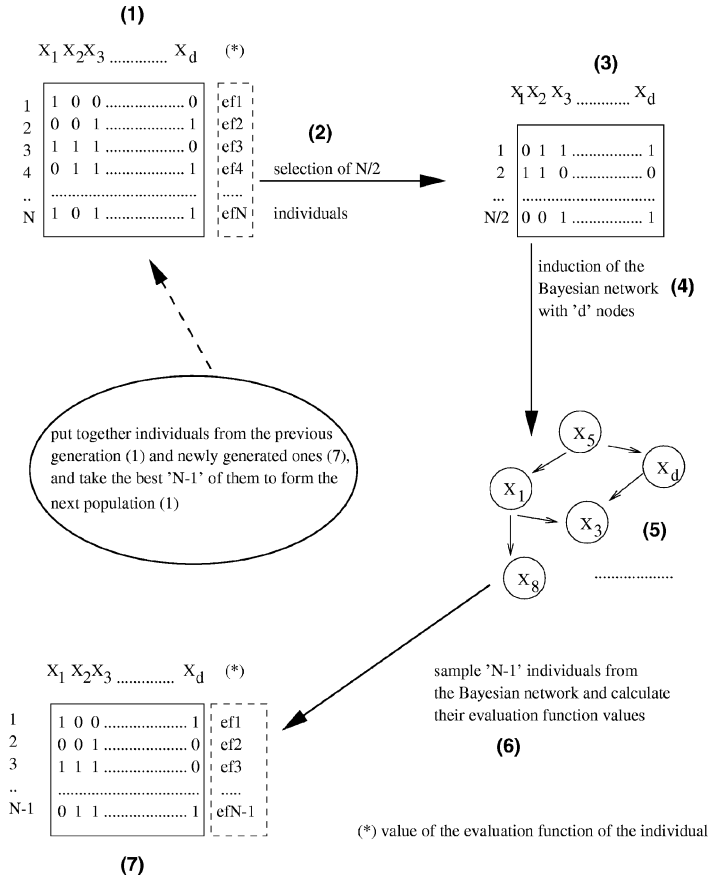

Fig. 6. FSS-EBNA method.

The election of the population size is related to the dimensionality of the domain and the used evaluation function. The justification of the population size will be explained after the presentation of the used datasets.

At this point of the explanation we would like to point out the similarities of the new algorithm with the work of Koller and Sahami [50]. They also use concepts from probabilistic reasoning to build a near optimal feature subset by a filter approach. They use concepts like conditional independence and Markov blanket, concepts which are used in the construction of Bayesian networks.

4.2. Characteristics of the evaluation function

A wrapper approach will be used to calculate the evaluation function value for each individual. The value of the evaluation function of a feature subset found by the EBNA search technique, once the classification algorithm is fixed, will be calculated by an error estimation in the training data.

The accuracy estimation, seen as a random variable, has an intrinsic uncertainty [44]. Based on Kohavi's [45] work on accuracy estimation techniques, a 10-fold cross-validation multiple times, combined with a heuristic proposed by Kohavi and John [47], will be used to control the intrinsic uncertainty of the evaluation function. This heuristic works as follows:

- If the standard deviation of the accuracy estimate is above 1%, another 10-fold cross-validation is executed.
- This is repeated until the standard deviation drops below 1%, a maximum of five times.
- In this way, small datasets will be cross-validated many times. However, larger ones possibly once.

4.3. Internal loop and external loop in FSS-EBNA

We consider that FSS-EBNA, as any machine learning algorithm to assess its accuracy, must be tested on unseen instances which do not participate in the selection of the best feature subset. Two accuracy estimation loops can be seen in the FSS process (see in Fig. 2):

- The *internal-loop* 10-fold cross-validation accuracy estimation that guides the search process is explained in the previous point. The internal loop represents the evaluation function of the proposed solution.
- The *external-loop* accuracy estimation, reported as the final 'goodness' of FSS-EBNA, is testing the feature subset selected by the internal loop on unseen instances not used in the search for this subset. Due to the non-deterministic nature of FSS-EBNA (two executions could not give the same result), five iterations of a two-fold cross-validation (5x2cv) have been applied as external-loop accuracy estimator.

4.4. The 'overfitting' problem and the stopping criterion

In the initial stages of the definition of FSS-EBNA, we hypothesized to report the accuracy of the internal loop as the final performance. However, Aha [3] and Kohavi [49], in personal communications, alerted us of the overtly optimistic nature of the cross-validation estimates which guide the search. Due to the search nature of FSS-EBNA, it is possible that one feature subset (from the big amount of subsets visited) could be very well adapted to the training set, but when presented to new instances not presented in the training process, the accuracy could dramatically decay: an 'overfitting' [78] can occur internally in the FSS process. Although it was not done by some authors, we recommend not to report the accuracy used to guide the search as the final accuracy of an FSS process.

Jain and Zongker [40] reported for a non-deceptive function in a pattern recognition problem that the quality of selected feature subsets for small training sets was poor; however, improved as the training set increased. Kohavi [46] also noted in a wrapper machine learning approach that the principal reason of 'overfitting' was the low amount of training instances.

To study this issue for FSS-EBNA, we have carried out a set of experiments with different training sizes of the *Waveform-40* dataset [15] with the *Naive-Bayes* classification

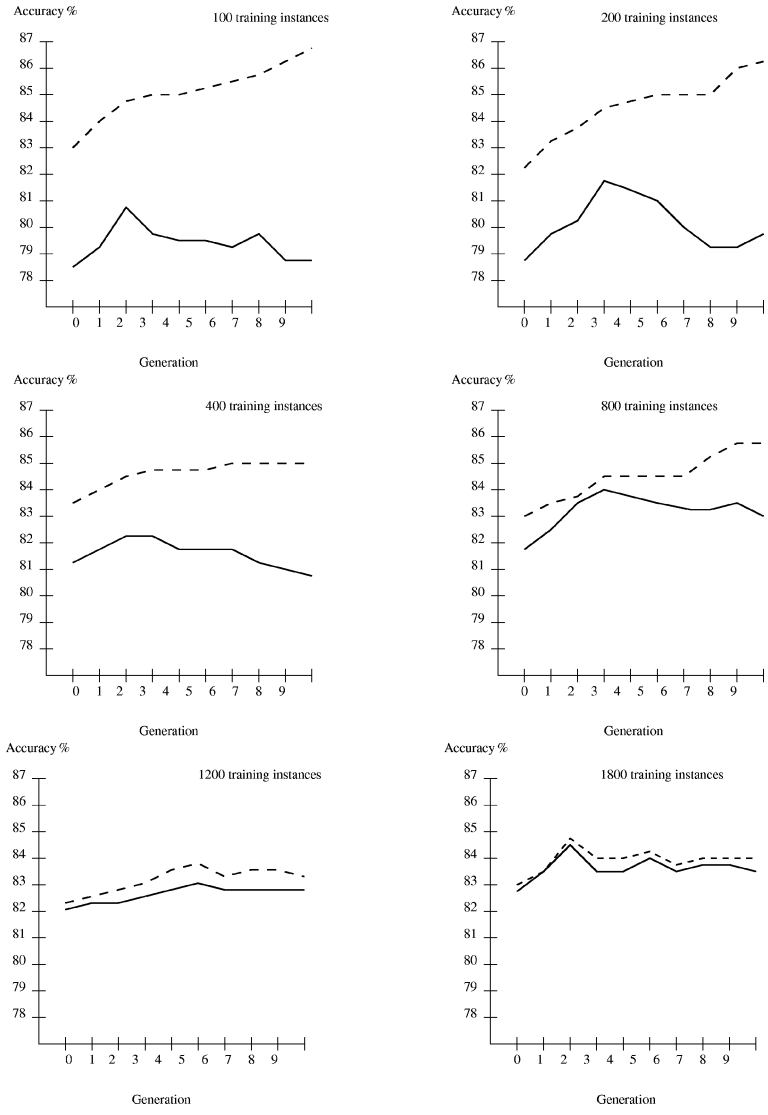

Fig. 7. Internal- and external-loop accuracy values in FSS-EBNA for different training sizes with the *Waveform*-40 dataset and the *Naive-Bayes* learning algorithm. The internal-loop accuracy 10-fold cross-validation is multiple times repeated until the standard deviation of the accuracy estimation drops below 1%. Dotted lines show the internal-loop accuracy estimation and solid lines the external-loop one. Both loop accuracies for the best solution of each search generation are represented. '0' generation represents the initial generation of the search.

algorithm [19]: training sizes of 100, 200, 400, 800 and 1600 samples and tested over a fixed test set with 3200 instances. Fig. 7 summarizes the set of experiments.

For 100, 200 and 400 training sizes, although the internal-loop cross-validation was repeated multiple times, differences between internal- and external-loop accuracies were

greater than twice the standard deviation of the internal loop. However, when the training size increases, the fidelity between internal- and external-loop accuracies increases, and the accuracy estimation of the external loop appears in the range formed by the standard deviation of the internal-loop accuracy.

Apart from these accuracy estimation differences between both loops, a serious 'overfitting' risk arises for small datasets: as the search process advances, the internal loop's improvement deceives us, as posterior performance on unseen instances does not improve. The difference between internal and external estimations would not be so important if both estimations had the same behaviour: that is, an improvement in the internal estimation coupled with an external improvement and a decrease in internal estimation coupled with an internal improvement. However, it clearly seems that this can not be guaranteed for small size training sets, where two curves show an erratic relation. Thus, FSS results generalization must be done with high care for small datasets.

It seems obvious that for small datasets it is not possible to see FSS-EBNA as an 'anytime algorithm' (Boddy and Dean [11]), where the quality of the result is a nondecreasing function in time. Looking at Fig. 7, we discard this 'monotonic-anytime idea' (more time $\leftrightarrow$ better solution) for small training set sizes. Our findings follow the work of Ng [70], who in an interesting work on the 'overfitting' problem, demonstrates that when cross-validation is used to select from a large pool of different classification models in a noisy task with too small training set, it may not be advisable to pick the model with minimum cross-validation error, and a model with higher cross-validation error will have better generalization error over novel test instances.

Regarding this behaviour, so related with the number of instances in the training set, the next heuristic as stopping criterion is adopted for FSS-EBNA:

- For datasets with more than 2000 instances (more than 1000 instances in each training subset for the 5x2cv external-loop accuracy estimation), the search is stopped when in a sampled new generation no feature subset appears with an evaluation function value improving the best subset found in the previous generation. Thus, the best subset of the search, found in the previous generation, is returned as FSS-EBNA's solution.
- For smaller datasets (less than 1000 instances in each training subset for the 5x2cv external-loop accuracy estimation), the search is stopped when in a sampled new generation no feature subset appears with an evaluation function value improving, at least with a p-value smaller than 0.1,[3] the value of the evaluation function of the feature subset of the previous generation. Thus, the best subset of the previous generation is returned as FSS-EBNA's solution.

An improvement in the internal-loop estimation is not the only measure to take into account to allow the continuation of the search in FSS-EBNA. The number of instances of the dataset is also critical for this permission. For larger datasets the 'overfitting' phenomenon has a lesser impact and we hypothesize that an improvement in the internal-loop estimation will be coupled with an improvement in generalization accuracy on unseen instances. Otherwise, for smaller datasets the 'overfitting' phenomenon has a

[3] Using a 10-fold cross-validated paired *t* test between the folds of both estimations, taking only the first run into account when 10-fold cross-validation is repeated multiple times.

greater risk in occurring and the continuation of the search is only allowed when a significant improvement in the internal-loop accuracy estimation of best individuals of consecutive generations appears. We hypothesize that when this significant improvement appears, the 'overfitting' risk decays and there is a basis for further generalization accuracy improvement over unseen instances.

5. Datasets and learning algorithms

5.1. Used datasets

Table 1 summarizes some characteristics of the selected datasets. Five real datasets come from the *UCI repository* [68]. The image dataset comes from the Statlog project [85].

LED24 (Breiman et al. [15]) is a well known artificial dataset with 7 equally relevant and 17 irrelevant binary features. We designed another artificial domain, called Redundant21, which involves 21 continuous features in the range [3,6]. The target concept is to define whether the instance is nearer (using the Euclidean distance) to $(0, 0, \ldots, 0)$ or $(9, 9, \ldots, 9)$. The first nine features appear in the target concept and the rest of the features are repetitions of relevant ones, where the 1st, 5th and 9th features are respectively repeated four times.

As the wrapper approach is used we must take into account the number of instances in order to select the datasets. Although the used learning algorithms (they we will be explained in the next point) are not computationally very expensive, the running times could be extremely high for datasets with more than 10,000 instances.

In order to select the datasets, another basic criterion is the number of features of the dataset. Once Bayesian networks are used to factorize the probability distribution of the best solutions of a population, a sufficient number of solutions must fixed to reliably estimate the parameters of the network. If we choose datasets of a larger dimensionality (more than 50 features), we would need an extremely large number of solutions (much more than the actual population size, 1000), associated with the cost of the calculation of

Table 1
Details of experimental domains. C = continuous. N = nominal

Domain	Number of instances	Number of classes	Number of features
(1) Ionosphere	351	2	34 (34-C)
(2) Horse-colic	368	2	22 (15-N, 7-C)
(3) Anneal	898	6	38 (32-C, 6-N)
(4) LED24	1000	10	24 (24-N)
(5) Image	2310	7	19 (19-C)
(6) Redundant21	2500	2	21 (21-C)
(7) Sick-euthyroid	3163	2	25 (7-C,18-N)
(8) Chess	3196	2	36 (36-N)

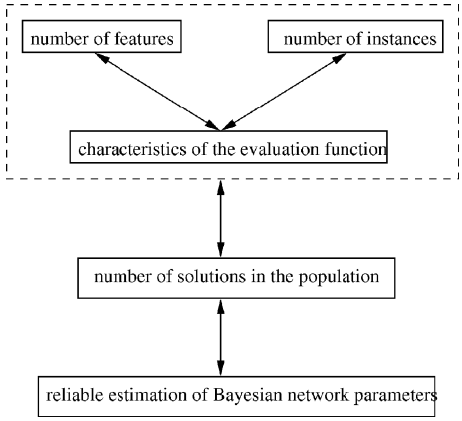

Fig. 8. Relations between relevant concepts to estimate a reliable Bayesian network.

their evaluation functions by wrapper approach, to reliably estimate the parameters of the network.

FSS-EBNA is independent to the evaluation function used and a filter approach could also be used. In this way, before the execution of FSS-EBNA, we must take into account the quantity of available computational resources in order to fix the following parameters for the estimation of a reliable Bayesian network: characteristics of the evaluation function, number of instances and features of the dataset and number of solutions in the population. Fig. 8 shows the relations between these concepts.

5.2. Learning algorithms

Two learning algorithms from different families are used in our experiments. [4]

- *ID*3 (Quinlan [77]) classification tree algorithm uses the gain-ratio measure to carry out the splits in the nodes of the tree. It does not incorporate a post-pruning strategy in the construction of the tree. It only incorporates a pre-pruning strategy, using the chi-squared statistic to guarantee a minimum dependency between the proposed split and the class.
- *Naive-Bayes* (*NB*) (Cestnik [19]) algorithm uses a variation of the Bayes rule to predict the class for each instance, assuming that features are independent to each other for predicting the class. The probability for nominal features is estimated from data using maximum likelihood estimation. A normal distribution is assumed to estimate the class conditional probabilities for continuous attributes. In spite of its simplicity Kohavi and John [47] noted NB's accuracy superiority over C4.5 (Quinlan [79]) in a set of real tasks.

ID3 has an embedded low capacity for discarding irrelevant features. It may not use all the available features in the tree structure, but it tends to make single class 'pure' folds in the decision tree, even if they only have a single training sample. Its tendency to 'overfit'

[4] It must be noted that in the 'wrapper' schema any classifier can be inserted.

the training data and damage the generalization accuracy on unseen instances has been noticed by many authors (Caruana and Freitag [17], Kohavi and John [47], Bala et al. [6]). Because one must not trust ID3's embedded capacity to discard irrelevant features, FSS can play a 'normalization' role to avoid these irrelevant splits, hiding the attributes from the learning algorithm which may 'overfit' the data in deep stages of the tree and do not have generalization power.

Despite its good scaling with irrelevant features, NB can improve its accuracy level discarding correlated and redundant features. NB, based on the independence assumption of predictive features to predict the class, is hurt by correlated features which violate this independence assumption. Thus, FSS can also play a 'normalization' role to discard these groups of correlated features, ideally selecting one of them in the final model. Although Langley and Sage [52] proposed a forward feature selection direction for detecting these correlations, Kohavi and John [47] proposed the backward direction.

6. Experimental results

As 5 iterations of a 2-fold cross-validation were applied, the reported accuracies are the mean of ten accuracies. The standard deviation of the mean is also reported. Tables 2 and 3 respectively show the accuracy of ID3 and NB, both with and without FSS-EBNA feature subset selection. Tables 4 and 5 respectively show the average cardinality of features used by ID3 and NB. Once 5 iterations of a 2-fold cross-validation were executed, a 5x2cv F (Alpaydin [5]) test was applied to determine whether accuracy differences between FSS-EBNA approach and no feature selection are significant or not. 5x2cv F test is a variation of the well known 5x2cv paired t test (Dietterich [26]). The p-value of the test is reported, which is the probability of observing a value of the test statistic that is at least as contradictory to the null hypothesis (compared algorithms have the same accuracy) as the one computed from sample data (Mendenhall and Sincich [61]).

Table 2
A comparison of accuracy percentages of ID3 with and without FSS-EBNA

Domain	ID3 without FSS	ID3 & FSS-EBNA	p-value
(1) Ionosphere	87.97 ± 3.68	88.77 ± 1.99	0.35
(2) Horse-colic	78.42 ± 4.16	83.65 ± 1.57	0.01
(3) Anneal	99.42 ± 0.55	99.40 ± 0.50	0.99
(4) LED24	58.21 ± 1.73	71.40 ± 1.72	0.00
(5) Image	95.52 ± 0.60	95.73 ± 0.86	0.95
(6) Redundant21	79.32 ± 1.11	79.32 ± 1.11	1.00
(7) Sick-euthyroid	96.78 ± 0.36	96.78 ± 0.41	1.00
(8) Chess	98.93 ± 0.40	99.05 ± 0.39	0.93
Average	86.81	89.06	

Table 3
A comparison of accuracy percentages of NB with and without FSS-EBNA

Domain	NB without FSS	NB & FSS-EBNA	p-value
(1) Ionosphere	84.84 ± 3.12	92.40 ± 2.04	0.00
(2) Horse-colic	78.97 ± 2.96	83.53 ± 1.58	0.01
(3) Anneal	93.01 ± 3.13	94.10 ± 3.00	0.10
(4) LED24	72.53 ± 0.91	72.78 ± 0.67	0.96
(5) Image	79.95 ± 1.52	90.01 ± 1.83	0.00
(6) Redundant21	79.48 ± 0.82	93.42 ± 0.90	0.00
(7) Sick-euthyroid	84.77 ± 2.70	96.14 ± 0.65	0.00
(8) Chess	87.22 ± 1.79	94.23 ± 0.35	0.00
Average	83.48	89.57	

Table 4
Cardinalities of selected feature subsets for ID3 with and without FSS-EBNA. It must be taken into account that ID3 carries out an embedded FSS and it can discard some of the available features in the construction of the decision tree. The third column shows the full set cardinality

Domain	ID3 without FSS	ID3 & FSS-EBNA	Full set
(1) Ionosphere	9.00 ± 1.15	6.50 ± 1.17	34
(2) Horse-colic	10.60 ± 1.17	3.30 ± 1.25	22
(3) Anneal	10.00 ± 1.15	8.70 ± 1.22	38
(4) LED24	24.00 ± 0.00	7.00 ± 0.82	24
(5) Image	11.50 ± 1.17	5.70 ± 1.05	19
(6) Redundant21	9.00 ± 0.00	9.00 ± 0.00	21
(7) Sick-euthyroid	9.40 ± 1.17	4.00 ± 0.66	25
(8) Chess	26.50 ± 2.01	21.20 ± 2.09	36

Table 6 shows in which generation stopped each of the ten runs of the 5x2cv procedure. Table 7 shows the average running times (in seconds) for these ten single folds. Experiments were run in a SUN-SPARC machine. The MLC++ software (Kohavi et al. [48]) was used to execute Naive-Bayes and ID3 algorithms.

- FSS-EBNA has helped ID3 to induce decision trees with significantly fewer attributes coupled with a maintenance of the predictive accuracy in the majority of databases. We place this result near the assertion made by Kohavi and John [47] on the preprocessing step already made to many real datasets which only include relevant features. ID3's accuracy is specially damaged by irrelevant features, and when the dataset is already preprocessed a FSS process is only able to detect a smaller feature subset that equalizes the accuracy of features used in the tree when no FSS process

Table 5
Cardinalities of selected feature subsets for NB with and without FSS-EBNA. It must be taken into account that when no FSS is applied to NB, it uses the full feature set to induce the classification model

Domain	NB without FSS = Full set	NB & FSS-EBNA
(1) Ionosphere	34	13.40 ± 2.11
(2) Horse-colic	22	6.10 ± 1.85
(3) Anneal	38	20.50 ± 3.13
(4) LED24	24	11.20 ± 1.61
(5) Image	19	7.10 ± 0.73
(6) Redundant21	21	9.00 ± 0.00
(7) Sick-euthyroid	25	9.80 ± 2.09
(8) Chess	36	17.30 ± 2.58

Table 6
Generation in which stopped each of the ten runs of the 5x2cv procedure. It must be noted that the subset returned by the algorithm was the best subset of the previous generation respect the stop. The initial generation is considered as '0'

Domain	ID3 & FSS-EBNA	NB & FSS-EBNA
(1) Ionosphere	2, 2, 2, 2, 2, 1, 2, 1, 2, 1	1, 1, 2, 2, 2, 2, 2, 2, 2, 2
(2) Horse-colic	2, 2, 2, 3, 3, 2, 2, 2, 1, 1	4, 2, 2, 2, 2, 2, 3, 2, 3, 2
(3) Anneal	1, 1, 1, 1, 1, 1, 1, 1, 1, 1	2, 2, 2, 2, 1, 2, 1, 2, 2, 2
(4) LED24	2, 2, 2, 2, 2, 2, 2, 2, 2, 2	3, 3, 2, 3, 3, 3, 3, 3, 2, 2
(5) Image	2, 1, 2, 3, 1, 1, 2, 2, 2, 2	4, 4, 4, 3, 3, 3, 3, 4, 4, 3
(6) Redundant21	1, 1, 1, 1, 1, 1, 1, 1, 1, 1	3, 2, 3, 2, 3, 2, 3, 3, 2, 2
(7) Sick-euthyroid	2, 2, 1, 1, 2, 2, 3, 2, 2, 2	4, 4, 4, 3, 5, 2, 4, 2, 3, 4
(8) Chess	5, 5, 4, 4, 4, 4, 4, 4, 4, 4	3, 3, 3, 4, 4, 3, 3, 4, 3, 4

is made. The average accuracy improvement over the set of databases is due to only three domains. In Ionosphere domain, a slight accuracy improvement is achieved and in Horce-colic, the improvement is significant. In LED24 artificial domain, specially selected to test the robustness of FSS-EBNA wrapped by ID3, the 17 irrelevant features are always filtered by FSS-EBNA and only the 7 relevant features are finally returned by FSS-EBNA. Otherwise, when no FSS is made, all the irrelevant features also appear in the tree.

- FSS-EBNA has also helped NB to significantly reduce the number of features needed to induce the final models. This dimensionality reduction is coupled with considerable accuracy improvements for all except one domain. In LED24 NB tolerates the influence of the 17 irrelevant features and further FSS is only able to reduce the dimensionality maintaining the predictive accuracy. The average accuracy

Table 7
CPU times, in seconds, for FSS-EBNA. Reported numbers reflect the average times and standard deviation for the ten folds of 5x2cv

Domain	ID3 & FSS-EBNA	NB & FSS-EBNA
(1) Ionosphere	23,105 ± 4830	2466 ± 842
(2) Horse-colic	28,021 ± 5331	2901 ± 698
(3) Anneal	24,127 ± 724	5213 ± 873
(4) LED24	64,219 ± 4536	6333 ± 1032
(5) Image	103,344 ± 24,675	15,243 ± 1675
(6) Redundant21	78,218 ± 1322	14,361 ± 1545
(7) Sick-euthyroid	48,766 ± 9433	15,541 ± 3786
(8) Chess	104,229 ± 9278	16,106 ± 2768

with respect to all domains increases from 83.48% to 89.57%, which implies a 36.86% relative reduction in the error rate. In Redundant21 artificial domain, specially selected to test the robustness of FSS-EBNA wrapped by NB, FSS-EBNA is able to detect all the redundancies that hurt NB's accuracy and violate its independence assumption among features, selecting only once, the repeated features which appear in the target concept.

- Owing to the fact that the wrapper approach FSS-EBNA needs large CPU times for ID3. Our approach, based on the evolution of populations, needs a minimum amount of individuals to be evaluated in order to reliably induce the Bayesian networks that guarantee the evolution of the process. The times needed to induce the Bayesian networks in each generation are insignificant in comparison to the time needed to calculate the evaluation functions: more than 99% of the whole CPU time is employed 'wrapping' over both learning algorithms in all the domains. The induction of the Bayesian networks by the presented local search mechanism has demonstrated a low cost. In order to induce a Bayesian network over the best individuals 3 CPU seconds are needed by average in Image domain (the domain with fewer features) and 14 CPU seconds in Anneal (the domain with more features). Due to the simplicity of the NB learning algorithm to be trained and tested (storage of conditional probabilities for each attribute given the class), the overall times for FSS-EBNA are considerably smaller.
- To understand the CPU times of Table 7, the generations where the searches stop must be also taken into account (Table 6). Each generation supposes the evaluation of 1000 individuals and differences in the stop generation generate the presented standard deviations of CPU time.

7. Summary and future work

GAs, due to its attractive, randomized and population-based nature, have long been applied for the FSS problem by statistics, pattern recognition and machine learning

communities. This work presents FSS-EBNA, a new search engine which shares these interesting characteristics of GAs. In FSS-EBNA, the FSS problem, stated as a search problem, uses the EBNA (Estimation of Bayesian Network Algorithm) search engine, a variant of the EDA (Estimation of Distribution Algorithm) approach. EDA, also based as GAs on the evolution of populations of solutions, is an attractive approach because it avoids the necessity of fixing crossover and mutation operators (and respective rates) so needed in GAs. The selection of crossover and mutation operators and rates is still an open problem in GA tasks. However, EDA guarantees the evolution of solutions by the factorization of the probability distribution of best individuals in each generation of the search. In EBNA, this factorization is carried out by a Bayesian network induced by a cheap local search mechanism.

The work exposes the different roots of the FSS-EBNA method and related work for each concept: the FSS process as a search problem, the EDA approach and the Bayesian networks. Joining the pieces provided by these three concepts the FSS-EBNA process can be understood.

Once the basic pieces are exposed, the different parameters of the FSS-EBNA process itself are presented and justified. A reflexion on the 'overfitting' problem in FSS is carried out and inspired on this reflexion the stop criterion of FSS-EBNA is determined, so related with the number of instances of the domain.

Our work has included two different, well known learning algorithms: ID3 and NB. The wrapper approach is used to asses the evaluation function of each proposed feature subset and it has needed a large amount of CPU time with the ID3 learning algorithm. However, the induction of the Bayesian networks that guarantees the evolution has demonstrated to be very cheap in CPU time. FSS-EBNA has been able to filter in artificial tasks, the special kind of features that hurt the performance of the specific learning a As future work, we consider lengthening the work already done (Inza [38]) using EBNA for the Feature Weighting problem in Nearest Neighbor Algorithm. Continuing the work within the EDA approach for FSS, an interesting way to be explored when the presented CPU times are prohibitive, is the use of filter approaches to calculate the evaluation function. In order to deal with domains with much larger numbers of features (> 100), future work should address the use of simpler probability models to factorize the probability distribution of best individuals, models which assume fewer or no dependencies between the variables of the problem. Another way of research will be the employment of a metric which fixes for each domain, the number of individuals needed to reliably learn (Friedman and Yakhini [32]) the parameters of the Bayesian network.

Acknowledgements

This work was supported by grants PI 96/12 from the Basque Country Government, Departamento de Educación, Universidades e Investigación and by CICYT under TIC97-1135-C04-03 grant.

References

[1] D.W. Aha, D. Kibler, M.K. Albert, Instance-based learning algorithms, Machine Learning 6 (1991) 37–66.
[2] D.W. Aha, R.L. Bankert, Feature selection for case-based classification of cloud types: An empirical comparison, in: Proc. AAAI-94, Seattle, WA, 1994, pp. 106–112.
[3] D.W. Aha, Personal communication, 1999.
[4] H. Almuallin, T.G. Dietterich, Learning with many irrelevant features, in: Proc. AAAI-91, Anaheim, CA, 1991, pp. 547–552.
[5] E. Alpaydin, Combined 5x2cv F test for comparing supervised classification learning algorithms, Neural Comput. 11 (1999) 1885–1892.
[6] J. Bala, K. DeJong, J. Huang, H. Wechsler, H. Vafaie, Hybrid learning using genetic algorithms and decision trees for pattern classification, in: Proc. IJCAI-95, Montreal, Quebec, 1995, pp. 719–724.
[7] S. Baluja, Population-based incremental learning: A method for integrating genetic search based function optimization and competitive learning, Technical Report CMU-CS-94-163, Carnegie Mellon University, Pittsburgh, PA, 1994.
[8] S. Baluja, S. Davies, Using optimal dependency-trees for combinatorial optimization: Learning the structure of the search space, in: Proc. 14th International Conference on Machine Learning, Nashville, TN, 1997, pp. 30–38.
[9] M. Ben-Bassat, Pattern recognition and reduction of dimensionality, in: P.R. Krishnaiah, L.N. Kanal (Eds.), Handbook of Statistics—II, North-Holland, Amsterdam, 1982, pp. 773–791.
[10] A.L. Blum, P. Langley, Selection of relevant features and examples in machine learning, Artificial Intelligence 97 (1997) 245–271.
[11] M. Boddy, T. Dean, Deliberation scheduling for problem solving in time-constrained environments, Artificial Intelligence 67 (2) (1997) 245–285.
[12] R.R. Bouckaert, Properties of Bayesian belief network learning algorithms, in: Proc. 10th Annual Conference on Uncertainty in Artificial Intelligence, Seattle, WA, 1994, pp. 102–109.
[13] D. Boyce, A. Farhi, R. Weischedel, Optimal Subset Selection, Springer, Berlin, 1974.
[14] W. Buntine, Theory refinement in Bayesian networks, in: Proc. 7th Conference on Uncertainty in Artificial Intelligence, Los Angeles, CA, 1991, pp. 52–60.
[15] L. Breiman, J.H. Friedmann, R.A. Olshen, C.J. Stone, Classification and Regression Trees, Wadsworth, Belmont, CA, 1984.
[16] C. Cardie, Using decision trees to improve case-based learning, in: Proc. 10th International Conference on Machine Learning, Amherst, MA, 1993, pp. 25–32.
[17] R. Caruana, D. Freitag, Greedy attribute selection, in: Proc. 11th International Conference on Machine Learning, New Brunswick, NJ, Morgan Kaufmann, Los Altos, CA, 1994, pp. 28–36.
[18] E. Castillo, J.M. Gutiérrez, A.S. Hadi, Expert Systems and Probabilistic Network Models, Springer, Berlin, 1997.
[19] B. Cestnik, Estimating probabilities: A crucial task in machine learning, in: Proc. European Conference on Artificial Intelligence (ECAI-90), Stockholm, Sweden, 1990, pp. 147–149.
[20] M. Chen, J. Han, P. Yu, Data mining: An overview from database perspective, IEEE Transactions on Knowledge and Data Engineering 8 (6) (1996) 866–883.
[21] D.M. Chickering, D. Geiger, D. Heckerman, Learning Bayesian networks is NP-hard, Technical Report MSR-TR-94-17, Microsoft Research, Advanced Technology Division, Microsoft Corporation, Redmond, WA, 1994.
[22] D.M. Chickering, D. Geiger, D. Heckerman, Learning Bayesian networks: Search methods and experimental results, in: Preliminary Papers of the 5th International Workshop on Artificial Intelligence and Statistics, Ft. Lauderdale, FL, 1995, pp. 112–128.
[23] G.F. Cooper, E.A. Herskovits, A Bayesian method for the induction of probabilistic networks from data, Machine Learning 9 (1992) 309–347.
[24] A.P. Dawid, Conditional independence in statistical theory, J. Roy. Statist. Soc. Ser. B 41 (1979) 1–31.
[25] J.S. De Bonet, C.L. Isbell, P. Viola, MIMIC: Finding optima by estimating probability densities, in: Th. Petsche, M. Mozer, M. Jordan (Eds.), Advances in Neural Information Processing Systems 9, MIT Press, Cambridge, MA, 1997.

[26] T.G. Dietterich, Approximate statistical tests for comparing supervised learning algorithms, Neural Comput. 10 (7) (1998) 1895–1924.
[27] J. Doak, An evaluation of feature selection methods and their application to computer security, Technical Report CSE-92-18, University of California at Davis, CA, 1992.
[28] R. Etxeberria, P. Larrañaga, J.M. Picaza, Analysis of the behaviour of genetic algorithms when learning Bayesian network structure from data, Pattern Recognition Lett. 18 (11–13) (1997) 1269–1273.
[29] R. Etxeberria, P. Larrañaga, Global optimization with Bayesian networks, in: Proc. II Symposium on Artificial Intelligence (CIMAF99), La Habana, Cuba, 1999, pp. 332–339.
[30] F.J. Ferri, V. Kadirkamanathan, J. Kittler, Feature subset search using genetic algorithms, in: Proc. IEE/IEEE Workshop on Natural Algorithms in Signal Processing, Essex, 1993, pp. 23/1–23/7.
[31] F.J. Ferri, P. Pudil, M. Hatef, J. Kittler, Comparative study of techniques for large scale feature selection, in: E.S. Gelsema, L.N. Kanal (Eds.), Multiple Paradigms, Comparative Studies and Hybrid Systems, North Holland, Amsterdam, 1994, pp. 403–413.
[32] N. Friedman, Z. Yakhini, On the sample complexity of learning Bayesian networks, in: Proc. 12th Conference on Uncertainty in Artificial Intelligence, Portland, OR, 1996, pp. 274–282.
[33] J.J. Grefenstette, Optimization of control parameters for genetic algorithms, IEEE Trans. Systems Man Cybernet. SMC-16 (1) (1986) 122–128.
[34] G.R. Harik, F.G. Lobo, D.E. Goldberg, The compact genetic algorithm, IlliGAL Report 97006, University of Illinois at Urbana-Champaign, Illinois Genetic Algorithms Laboratory, Urbana, IL, 1997.
[35] D. Heckerman, D. Geiger, D. Chickering, Learning Bayesian networks: The combinations of knowledge and statistical data, Machine Learning 20 (1995) 197–243.
[36] M. Henrion, Propagating uncertainty in Bayesian networks by probabilistic logic sampling, in: J.F. Lemmer, L.N. Kanal (Eds.), Uncertainty in Artificial Intelligence 2, Elsevier Science, Amsterdam, 1988, pp. 149–163.
[37] J.H. Holland, Adaptation in Natural and Artificial Systems, University of Michigan Press, Ann Arbor, MI, 1975.
[38] I. Inza, Feature weighting for nearest neighbor algorithm by Bayesian networks based combinatorial optimization, in: Proc. Student Session of Advanced Course on Artificial Intelligence (ACAI-99), Chania, Greece, 1999, pp. 33–35.
[39] A.K. Jain, R. Chandrasekaran, Dimensionality and sample size considerations in pattern recognition practice, in: P.R. Krishnaiah, L.N. Kanal (Eds.), Handbook of Statistics—II, North-Holland, Amsterdam, 1982, pp. 835–855.
[40] A. Jain, D. Zongker, Feature selection: Evaluation, application, and small sample performance, IEEE Transactions on Pattern Analysis and Machine Intelligence 19 (2) (1997) 153–158.
[41] G. John, R. Kohavi, K. Pfleger, Irrelevant features and the subset selection problem, in: Proc. 11th International Conference on Machine Learning, New Brunswick, NJ, 1994, pp. 121–129.
[42] K. Kira, L.A. Rendell, The feature selection problem: Traditional methods and a new algorithm, in: Proc. AAAI-92, San Jose, CA, 1992, pp. 129–134.
[43] J. Kittler, Feature set search algorithms, in: C.H. Chen (Ed.), Pattern Recognition and Signal Processing, Sijthoff and Noordhoff, Alphen aan den Rijn, The Netherlands, 1978, pp. 41–60.
[44] R. Kohavi, Feature Subset Selection as search with probabilistic estimates, in: Proc. AAAI Fall Symposium on Relevance, New Orleans, LA, 1994, pp. 122–126.
[45] R. Kohavi, A study of cross-validation and bootstrap for accuracy estimation and model selection, in: Proc. IJCAI-95, Montreal, Quebec, 1995, pp. 1137–1143.
[46] R. Kohavi, Feature Subset Selection using the wrapper method: Overfitting and dynamic search space topology, in: Proc. First International Conference on Knowledge Discovery and Data Mining (KDD-95), Montreal, Quebec, 1995, pp. 192–197.
[47] R. Kohavi, G. John, Wrappers for feature subset selection, Artificial Intelligence 97 (1–2) (1997) 273–324.
[48] R. Kohavi, D. Sommerfield, J. Dougherty, Data mining using MLC++, a Machine Learning Library in C++, Internat. J. Artificial Intelligence Tools 6 (4) (1997) 537–566.
[49] R. Kohavi, Personal communication, 1999.
[50] D. Koller, M. Sahami, Toward optimal feature selection, in: Proc. 13th International Conference on Machine Learning, Bari, Italy, 1996, pp. 284–292.
[51] L. Kuncheva, Genetic algorithms for feature selection for parallel classifiers, Inform. Process. Lett. 46 (1993) 163–168.

[52] P. Langley, S. Sage, Induction of selective Bayesian classifiers, in: Proc. 10th Conference on Uncertainty in Artificial Intelligence, Seattle, WA, 1994, pp. 399–406.
[53] P. Larrañaga, C.M.H. Kuijpers, R.H. Murga, Y. Yurramendi, Learning Bayesian network structures by searching for the best ordering with genetic algorithms, IEEE Trans. Systems Man Cybernet.—Part A: Systems and Humans 26 (4) (1996) 487–493.
[54] P. Larrañaga, M. Poza, Y. Yurramendi, R.H. Murga, C.M.H. Kuijpers, Structure learning of Bayesian networks by genetic algorithms: A performance analysis of control parameters, IEEE Transactions on Pattern Analysis and Machine Intelligence 18 (9) (1996) 912–926.
[55] P. Larrañaga, R. Etxeberria, J.A. Lozano, B. Sierra, I. Inza, J.M. Peña, A review of the cooperation between evolutionary computation and probabilistic graphical models, in: Proc. II Symposium on Artificial Intelligence (CIMAF99), La Habana, Cuba, 1999, pp. 314–324.
[56] S.L. Lauritzen, Graphical Models, Oxford University Press, Oxford, 1996.
[57] H. Liu, R. Setiono, Feature selection and classification—A probabilistic wrapper approach, in: Proc. 9th International Conference on Machine Learning, Bari, Italy, 1996, pp. 284–292.
[58] H. Liu, H. Motoda, Feature Selection for Knowledge Discovery and Data Mining, Kluwer Academic, Norwell, MA, 1998.
[59] H. Liu, R. Setiono, Incremental feature selection, Applied Intelligence 9 (3) (1998) 217–230.
[60] D. Madigan, A.E. Raftery, C.T. Volinsky, J.A. Hoeting, Bayesian model averaging, in: Proc. AAAI Workshop on Integrating Multiple Learned Models, Portland, OR, 1996, pp. 77–83.
[61] W. Mendenhall, T. Sincich, Statistics for Engineering and the Sciences, Prentice Hall, Englewood Cliffs, NJ, 1998.
[62] A.J. Miller, Subset Selection in Regression, Chapman and Hall, Washington, DC, 1990.
[63] D. Mladenić, Feature subset selection in text-learning, in: Proc. 10th European Conference on Machine Learning, Chemnitz, Germany, 1998, pp. 95–100.
[64] A.W. Moore, M.S. Lee, Efficient algorithms for minimizing cross validation error, in: Proc. 11th International Conference on Machine Learning, New Brunswick, NJ, 1994, pp. 190–198.
[65] H. Mühlenbein, G. Paaß, From recombination of genes to the estimation of distributions. Binary parameters, in: H.M. Voigt et al. (Eds.), Parallel Problem Solving from Nature—PPSN IV, Lecture Notes in Computer Science, Vol. 1411, Springer, Berlin, 1996, pp. 178–187.
[66] H. Mühlenbein, The equation for response to selection and its use for prediction, Evolutionary Comput. 5 (3) (1997) 303–346.
[67] H. Mühlenbein, T. Mahnig, A. Ochoa, Schemata, distributions and graphical models in evolutionary optimization, J. Heuristics 5 (1999) 215–247.
[68] P. Murphy, UCI Repository of Machine Learning Databases, University of California, Department of Information and Computer Science, Irvine, CA, 1995.
[69] P. Narendra, K. Fukunaga, A branch and bound algorithm for feature subset selection, IEEE Trans. Comput. C-26 (9) (1977) 917–922.
[70] A.Y. Ng, Preventing 'overfitting' of cross-validation data, in: Proc. 14th Conference on Machine Learning, Nashville, TN, 1997, pp. 245–253.
[71] J. Pearl, Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference, Morgan Kaufmann, Palo Alto, CA, 1988.
[72] M. Pelikan, H. Mühlenbein, The bivariate marginal distribution algorithm, 1999, submitted for publication.
[73] M. Pelikan, D.E. Goldberg, E. Cantú-Paz, BOA: The Bayesian optimization algorithm, IlliGAL Report 99003, University of Illinois at Urbana-Champaign, Illinois Genetic Algorithms Laboratory, Urbana, IL, 1999.
[74] M. Pelikan, D.E. Goldberg, F. Lobo, A survey of optimization by building and using probabilistic model, IlliGAL Report 99018, University of Illinois at Urbana-Champaign, Illinois Genetic Algorithms Laboratory, Urbana, IL, 1999.
[75] F. Provost, V. Kolluri, A survey of methods for scaling up inductive algorithms, Data Mining and Knowledge Discovery 2 (1999) 131–169.
[76] P. Pudil, J. Novovicova, J. Kittler, Floating search methods in feature selection, Pattern Recognition Lett. 15 (1) (1994) 1119–1125.
[77] J.R. Quinlan, Induction of decision trees, Machine Learning 1 (1986) 81–106.

[78] J.R. Quinlan, Inferring decision trees using the minimum description length principle, Inform. and Comput. 80 (1989) 227–248.
[79] J.R. Quinlan, Programs for Machine Learning, Morgan Kaufmann, San Mateo, CA, 1993.
[80] G. Schwarz, Estimating the dimension of a model, Ann. Statist. 7 (1978) 461–464.
[81] W. Siedelecky, J. Skalansky, On automatic feature selection, Internat. J. Pattern Recognition and Artificial Intelligence 2 (1988) 197–220.
[82] D.B. Skalak, Prototype and feature selection by sampling and random mutation hill-climbing algorithms, in: Proc. 11th International Conference on Machine Learning, New Brunswick, NJ, 1994, pp. 293–301.
[83] S.D. Stearns, On selecting features for pattern classifiers, in: Proc. 3rd International Conference on Pattern Recognition, Coronado, CA, 1976, pp. 71–75.
[84] G. Syswerda, Uniform crossover in genetic algorithms, in: Proc. International Conference on Genetic Algorithms 3, Arlington, VA, 1989, pp. 2–9.
[85] C. Taylor, D. Michie, D. Spiegelhalter, Machine Learning, Neural and Statistical Classification, Paramount Publishing International, 1994.
[86] H. Vafaie, K. De Jong, Robust feature selection algorithms, in: Proc. 5th International Conference on Tools with Artificial Intelligence, Rockville, MD, 1993, pp. 356–363.
[87] M.L. Wong, W. Lam, K.S. Leung, Using evolutionary programming and minimum description length principle for data mining of Bayesian networks, IEEE Transactions on Pattern Analysis and Machine Intelligence 21 (2) (1999) 174–178.
[88] J. Yang, V. Honavar, Feature subset selection using a genetic algorithm, IEEE Intelligent Systems 13 (2) (1998) 44–49.
[89] Y. Yang, J.O. Pedersen, A comparative study on feature selection in text categorization, in: Proc. 14th International Conference on Machine Learning, Nashville, TN, 1997, pp. 412–420.