ELSEVIER

Pattern Recognition Letters 20 (1999) 1201–1209

Pattern Recognition Letters

www.elsevier.nl/locate/patrec

Representing the behaviour of supervised classification learning algorithms by Bayesian networks

I. Inza [*], P. Larrañaga, B. Sierra, R. Etxeberria, J.A. Lozano, J.M. Peña

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

Abstract

In this paper, an approach to study the nature of the classification models induced by Machine Learning algorithms is proposed. Instead of the predictive accuracy, the values of the predicted class labels are used to characterize the classification models. Over these predicted class labels Bayesian networks are induced. Using these Bayesian networks, several assertions are extracted about the nature of the classification models induced by Machine Learning algorithms. © 1999 Elsevier Science B.V. All rights reserved.

Keywords: Machine Learning; Classification learning algorithm; Joint behaviour; Bayesian networks; Structure learning

1. Introduction

The objective of a supervised classification learning algorithm is to induce a *general rule* that allows us to classify new examples $E* = \{e_{n+1}, \dots, e_{n+m}\}$ that are only characterized by their p descriptive variables. To generate this *general rule*, we have a set of n samples $E = \{e_1, \dots, e_n\}$ characterized by p descriptive variables $X = \{X_1, \dots, X_p\}$ and the class label $C = \{w_1, \dots, w_n\}$ to which they belong. The *general rule* (or classifier) can be seen as a classification hypothesis (or model) induced by the learning algorithm.

This problem was studied by the statistic community (Duda and Hart, 1973), using the term *Pattern Recognition*. In the Machine Learning literature, many representations for inducing classification hypotheses have been suggested (including decision trees, rule induction, Naive Bayes or k-NN), assuming the target function belongs to some restricted space of hypotheses.

To form a hypothesis structure, an algorithm makes assumptions, which are called biases in Machine Learning. Apart from the data, biases are the principal builders of a learning algorithm's hypothesis. A question of interest for researchers in Machine Learning is how to define the biases of existing algorithms and how to find out when a given bias is appropriate, based on background knowledge. Biases can be divided into two types (Kohavi, 1995a):

- *Restricted hypothesis space bias*. This bias assumes that the model belongs to some restricted space of hypotheses, typically defined in terms of their representation. For example, most decision tree algorithms restrict the hypothesis

[*] Corresponding author. Tel.: +34-943018000, x5106; fax: 34-943219306.
E-mail address: ccbincai@si.ehu.es (I. Inza)

space to the space of finite trees with univariate splits in its nodes, assuming that classes are separated by line segments parallel to the coordinate axes.

- *Preference bias*. This bias places a preference ordering on hypotheses. Many times the preference ordering is defined by how the search through the space of hypotheses is conducted. Most preference biases attempt to minimize some measure of syntactic complexity, following Occam's Razor principle of preferring simpler hypotheses.

Regarding the large amount of available algorithms, the user is frequently faced with the problem of selecting the ideal algorithm for a specific dataset, trying to select the learning algorithm of which the biases are best suited to the data. The ambitious objective of generating and selecting a unique winner algorithm for all datasets has been rejected by the empirical evidence of the 'No Free Lunch Theorem' (Kohavi et al., 1997). The selection of the 'best' algorithm, usually only based on the error percentage, had the effect of mainly focusing the attention of the Machine Learning community on predictive accuracy. A no-end career can be felt in Machine Learning, with the aim of constructing the algorithm with the highest predictive accuracy for each dataset. In this way, Clark (1998) mentioned us the obsession of the Machine Learning community with summary statistics.

Since our aim is to study the learning algorithm induced hypothesis' nature, we think that the error percentage, which is a measure of the accuracy of the generated model, cannot help us. Instead of the predictive accuracy, the class predictions will be used as the external expression of the hypothesis induced by the learning algorithm. The probability distribution of the class predictions of a set of learning algorithms over a dataset will be studied in a joint manner by a Bayesian network, displaying the joint behaviour of the hypotheses induced by these algorithms.

We will consider the qualitative part of one kind of probabilistic graphical models known as Bayesian networks to represent the joint behaviour of learning algorithms. Using the semantics of the Bayesian networks, regularities are found, based on the definition of conditional independence, about the classification hypotheses induced by common Machine Learning inducers over a set of medical datasets. The method proposed in this paper does not compare or study algorithms from the point of view of classification accuracy. The nature of the hypotheses induced by a set of algorithms is our main interest: thus, for this purpose, class predictions are used.

The work is organized as follows. Section 2 introduces Bayesian networks, based on the conditional independence concept. Various approaches for inducing Bayesian networks are also related. Section 3 presents the datasets and Machine Learning algorithms used and the chosen methodology for inducing the Bayesian networks. The proposed concepts to extract conclusions from the Bayesian networks about the joint behaviour of the algorithms also appear in this section. Section 4 shows the results obtained in tested domains and their interpretations. A resumé and future work appear in Section 5.

2. Bayesian networks

Bayesian networks (BNs) (Pearl, 1988) constitute a probabilistic framework for reasoning under uncertainty. From an informal perspective, BNs are directed acyclic graphs (DAGs) where the nodes are random variables and the arcs specify the independence assumptions that must be held between the random variables. BNs are based upon the concept of conditional independence among variables. This concept makes a factorization of the probability distribution of the n-dimensional random variable $(Z_1, \ldots, Z_n)$ possible in the following way:

$$P(z_1, \ldots, z_n) = \prod_{i=1}^{n} P(z_i \mid \mathrm{pa}(z_i)),$$

where z_i represents the value of the random variable Z_i and $\mathrm{pa}(z_i)$ represents the value of the random variables parents of Z_i.

Thus, in order to specify the probability distribution of a BN, one must give prior probabilities for all root nodes (nodes with no predecessors) and conditional probabilities for all other nodes, given all possible combinations of their direct predecessors. These numbers, in conjunction with the DAG, specify the BN completely. Once the network is constructed, it constitutes an efficient device to perform probabilistic inference. Nevertheless, the problem of building such a network remains. The structure and conditional probabilities necessary for characterizing the network can be either provided externally by experts or obtained, as in this paper, from an algorithm which automatically induces them. Due to the availability of large amounts of data coming from uncertain domains and the difficulties in having an expert that represents the conditional independencies, a development of automatic or semi-automatic methods that induce the Bayesian network structure has occurred, expressing the cited conditional independencies. The problem of inducing a Bayesian network structure from a set of data – also fixing for each variable the maximum number of parents – is NP-hard (Heckerman et al., 1995).

Structural learning methods can be classified using different criteria. One approach, with methods based on statistical tests of conditional independence (Kreiner, 1989), constructs a list of conditional dependencies and independencies for a three variable set. A Bayesian network is then created with the aim of reflecting the conditional (in)dependencies of the list. However attractive, this approach has lost relevance against the approach so called 'score + search'.

In this second approach, a metric or score (measuring the accuracy of the hypothesis structure) and a search procedure are defined. The procedure guides the search in an intelligent way in the huge space of Bayesian network structures, whose cardinality is expressed by Robinson's formula (Robinson, 1977). Between the quality measures we will cite Bayesian approaches (Cooper and Herskovits, 1992; Heckerman et al., 1995), approaches based on an information criterion (Herskovits and Cooper, 1990) and ones based on the minimum description length (Lam and Bacchus, 1994). There are many search strategies: we can cite greedy procedures (Cooper and Herskovits, 1992), tabu search (Bouckaert, 1995) or genetic algorithms (Larrañaga et al., 1996; Etxeberria et al., 1997). For a review of automatic learning methods (Heckerman, 1995) can be consulted.

The objective of this work is to express the joint behaviour of a set of known Machine Learning algorithms using an automatically generated Bayesian network. Indeed, we will use the expressive capabilities of the Bayesian networks: conditional independence relations between subsets of three Machine Learning algorithms and the connections between the nodes of the net.

Lauritzen et al.'s criteria (Lauritzen et al., 1990) to find conditional independencies inside the net will be used. The definition of conditional independence is as follows (Dawid, 1979).

Let X, Y and Z be three disjoint sets of variables, then X is said to be conditionally independent of Y given Z, if and only if $p(x \mid z, y) = p(x \mid z)$, for all possible values x, y and z of X, Y and Z, and then we write $I(X, Y \mid Z)$; otherwise X and Y are said to be conditionally dependent given Z, and then we write $D(X, Y \mid Z)$. The definition of conditional independence conveys the idea that once Z is known, knowing Y can no longer influence the probability of X. In other words, if Z is already known, knowledge of Y does not add any new information about X.

3. Proposed approach

3.1. Datasets used

Eleven medical databases from the *UCI Machine Learning Repository* (Murphy and Aha, 1994) are selected. Selecting the datasets from a specific domain, we hope to obtain more homogeneous conclusions. All databases have a separate set of training data and testing data in a 2/3:1/3 proportion. The characteristics of the databases are given in Table 1.

3.2. Classifiers

Fourteen well-known learning algorithms with different biases are used in experiments. Most relevant biases for each algorithm will be cited:

Table 1
Details of tested domains[a]

Domain	Number of examples	Number of classes	Number of attributes and types
Breast cancer	191 TRN – 95 TST	2	9 N
Breast (Wisconsin)	466 TRN – 233 TST	2	10 C
Cleveland	202 TRN – 101 TST	2	6 C, 7 N
Diabetes (Pima)	512 TRN – 256 TST	2	8 C
Echocardiogram	87 TRN – 44 TST	2	6 C, 1 N
Heart disease	180 TRN – 90 TST	2	13 C
Hepatitis	103 TRN – 52 TST	2	6 C, 13 N
Hungarian	196 TRN – 98 TST	2	8 C, 5 N
Hypothyroid	2,108 TRN – 1,055 TST	2	7 C, 18 N
Liver (BUPA)	230 TRN – 115 TST	2	6 C
Lymphography	98 TRN – 50 TST	4	19 N

[a] TRN: training; TST: test; C: continuous; N: nominal.

- *ID3* decision tree algorithm (Quinlan, 1986) which only pre-prunes the tree. It carries out univariate splits in tree nodes.
- *C4.5* decision tree algorithm (Quinlan, 1993). Other than *ID3*, it makes a post-pruning phase, based on an error-based-pruning algorithm. Univariate splits in tree nodes are carried out.
- *OC1* oblique decision tree algorithm (Murthy and Salzberg, 1994). It builds a hyperplane (considering all the features) at each tree node, using a randomized mechanism. It also incorporates a post-pruning strategy.
- *T2* two-level univariate-split decision tree algorithm (Auer et al., 1995). It builds the two-level decision tree that minimizes the number of errors in the training set.
- *Naive Bayes* (*NB*) algorithm (Cestnik, 1990). It is based on Bayesian rules and assumes independence between the occurrences of feature values to predict the class.
- *Naive Bayes Tree* (*NBTree*) algorithm (Kohavi, 1996). It runs *NB* at the leaves of a univariate-split decision tree.
- *HOODG* builds oblivious graphs bottom-up (Kohavi, 1995a). It can be seen like a decision tree that tests the same single attribute in all the nodes of the same level. It is coupled with feature subset selection (wrapper form (Kohavi, 1995a)) and discretization of data.
- *IB1* Aha et al.'s instance-based inducer that uses homogeneous weights for all attributes to compute the dissimilarity function, assuming that all features do not have the same relevance to define that class (Aha et al., 1991).
- *IB4* Aha et al.'s instance-based inducer that incorporates a weight learning capability: different weights are computed for each feature, according to its relevance, assuming that all features should not have the same importance to define that class (Aha et al., 1991).
- *PEBLS* instance-based inducer (Cost and Salzberg, 1993). It incorporates MVDM distance metric to deal with symbolic features, a modification of VDM metric.
- *Table-majority* is based on decision table paradigm (Kohavi, 1995b). It stores a table of all instances, predicting according to the table. If an instance is not found, it predicts the majority class. As for *HOODG*, it is coupled with feature subset selection and discretization of data.
- *OneR* is a simple learning algorithm inspired by Occam's razor. It computes the one-level decision tree that minimizes the number of errors in the training set (Holte, 1993).
- *CN2* rule induction algorithm, based on the work of Clark and Nibblet (1989). It uses statistical tests to expand classification rules. It does not have a post-prune mechanism.
- *Ripper* rule induction algorithm, based on the work of Cohen (1995). Other than *CN2*, *Ripper* post-prunes the generated rules by the "Reduced Error Pruning" technique (Quinlan, 1993).

Experiments are run on a SUN-SPARC computer using the *MLC++* Machine Learning library of programs (Kohavi et al., 1997) and Hugin software (Andersen et al., 1989) for the management of Bayesian networks. Each algorithm is run with its default parameters. No special treatment is done for unknown values, exploiting for each algorithm its own characteristics. *PEBLS* and *HOODG* algorithms are not able to handle unknown values: thus, they are only used in the four datasets without unknown values (diabetes, heart, liver and lymphography).

For each database and algorithm, a classification model is induced using the specified training set: when run with fixed default parameters, this process is deterministic for all algorithms except for *OC1*. [1] This model was used to classify the instances of the test set and, then, the class predictions for these test instances are saved in a new file. In this way, running all the algorithms over a database, we construct a new file where the rows represent the samples of the test set and the columns represent the learning algorithms: thus, in the (i,j) position of the constructed file, the class label prediction made by the jth classifier for the ith instance of the test set will appear.

We will use this new file as input to induce the Bayesian network. For a domain, each column of this file summarizes the probability distribution of the hypothesis induced by a learning algorithm. It must be noted that the real class label of each test instance does not appear in this file. Neither does any accuracy measure of each algorithm. The Bayesian network will only reflect the joint probability distribution of the class label predictions of used inducers.

3.3. Proposed modelization for inducing the Bayesian network structures

The structure of the Bayesian network is induced, as cited in Section 2, by the 'score + search' approach. The implementation proposed in (Larrañaga et al., 1996) is used: with the Bayesian $K2$ metric (Cooper and Herskovits, 1992), the search is conducted by an elitist genetic algorithm, hybridized with a local optimizer.

[1] *OC1* incorporates a randomized component to form splitting hyperplanes: in the datasets tested, predictions of different 10 runs were nearly the same. Predictions of a randomly selected run are used.

Ten of the eleven databases (all except hypothyroid) have test sets with less than 300 instances. Due to this absence of data, a typical problem in medical domains, the Bayesian networks induced from these restricted sets had high level of connectivity, indicating an 'overfitting' to the data (Quinlan, 1989). A simplification of the induced Bayesian network is conducted using an intuitive mechanism:

- An ordered list of the arcs appearing in the 'overfitted' network is assembled. The order reflected the influence of the arc in the overall $K2$ metric of the network structure.
- Half of the most influencial arcs are maintained. Half of the least influencial arcs are removed.
- Then, we start a *Recover procedure* for the removed arcs: taking into account two of the least important arcs maintained in the network, we consider recovering the removed most important arc. If the influence of this removed arc is smaller than half of the mean of the influence of the two least important arcs maintained, then the *Recover procedure* is stopped. Otherwise, the removed arc is recovered in the network and the recovering of the next removed arc is considered (taking into account the arc recently recovered in the network structure). Continue until stop.

The *Recover procedure* can be seen as a heuristic of the 'likelihood ratio test statistic'. With this simplification procedure, interpretable structures are achieved. This simplification procedure is not done for the hypothyroid database because its Bayesian network is induced from 1055 cases, a sufficient number to obtain a 'not-overfitted' Bayesian network. Fig. 1 summarizes the explained process. As an example, the induced simplified Bayesian network for the *Breast cancer* dataset can be seen in Fig. 2.

3.4. Concepts for interpreting the joint behaviour

Once the Bayesian networks are induced, our aim is to extract assertions on the joint behaviour of Machine Learning inducers: assertions on the similarities and dissimilarities between algorithms hypotheses. For this purpose, the following three concepts are used.

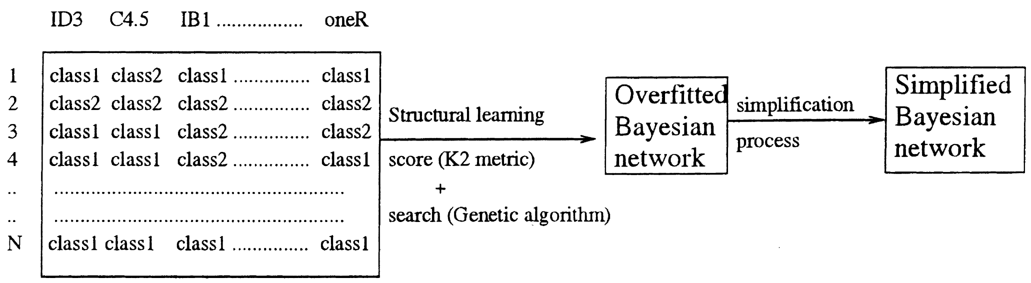
ID3 C4.5 IB1 .... oneR
1 class1 class2 class1 .... class1
2 class2 class2 class2 .... class2
3 class1 class1 class2 .... class2
4 class1 class1 class2 .... class1
N class1 class1 class1 .... class1
Structural learning
score (K2 metric)
+
search (Genetic algorithm)
Overfitted Bayesian network
simplification process
Simplified Bayesian network

Fig. 1. Proposed modelization process for each dataset. We start from the class predictions of Machine Learning inducers for the test instances. *N* indicates the number of instances in test set.

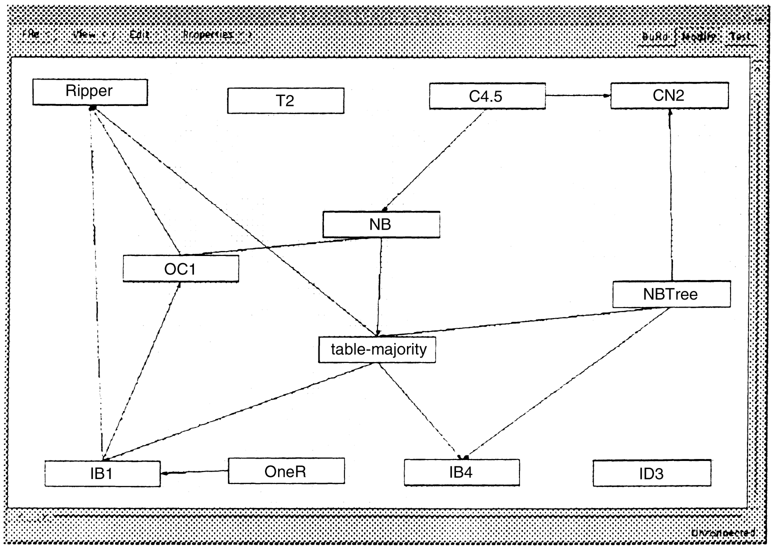

Fig. 2. Simplified Bayesian network of *Breast cancer* task.

3.4.1. Hard conditional independence

As *hard conditionally independent* we consider a learning algorithm that in a Bayesian network, given another algorithm, is conditionally independent of the rest of the algorithms. This condition does not easily occur in a Bayesian network. It implies that, giving only another algorithm, the hard conditionally independent one has a different behaviour regarding the other algorithms. We consider that the hard conditionally independent algorithm creates an 'original' hypothesis regarding the other algorithms: it has little relation to the hypotheses generated by the major part of the other algorithms. The probability distribution of its hypothesis needs just another algorithm to be explained in the whole representation of probability distributions induced by the Bayesian network.

3.4.2. Conditional independence in a proposed family of algorithms

Our aim is to study the 'union-degree' of a proposed family of learning algorithms in a joint manner. The conditional independence between two algorithms of the same family given another algorithm from a different family will be studied in the Bayesian network. That is:

Let X, Y and Z be three different algorithms and X and Y be from the same family and Z from another family. Is $I(X, Y \mid Z)$ true or is $D(X, Y \mid Z)$ true? This type of conditional independence implies a serious difference between the probability distributions of both algorithms of the same family. This gives us an idea on the 'compactness' (or similarity degree) between algorithms of a proposed family.

Families are formed based on the history of classic paradigms exposed in the most common Machine Learning reviews (Mitchell, 1997):

- Decision-Trees (DT): Quinlan's Decision Tree classic algorithms, *ID3* and *C4.5*.
- Simple-Trees (ST): one-depth (*oneR*) and two-depth (*T2*) simple classification trees. They minimize the number of errors in the training set, based on their constraints.
- K-NN (KNN): algorithms based on Nearest Neighbor Aha et al.'s ideas, *IB1* and *IB4*.
- Decision-Rules (DR): decision rules classic algorithms, *Ripper* and *CN2*.
- Bayesian-Approaches (BC): *Naive Bayes* and *Naive Bayes Tree*.

Remaining algorithms are not considered to form families.

3.4.3. Conditional independence between proposed families

Using the family definitions of the previous point, the conditional independence of all the learning algorithms of a family on all the algorithms of another family, given at least any one algorithm which does not belong to any of the compared families, will be studied. This conditional independence between proposed families means that given any set of algorithms that do not belong to any of the compared families, the probability distribution of each component of a family does not need the probability distribution of any component of the other family to be explained. This gives us an idea on the 'dissimilarity' between proposed family paradigms.

4. Results from induced Bayesian networks

Assertions on different types of behaviour will be extracted, based on the number of times a learning algorithm or a set of algorithms shows one of the explained conditional independence variants in the Bayesian network structure. Based on the number of domains for which a learning algorithm or a set of algorithms presents one of the explained concepts, assertions about different types of behaviour of the studied algorithms can be extracted. It must be noted that the extracted assertions are restricted to the domains [2] included in the experiments.

4.1. Hard conditional independencies

Number of databases where each algorithm is 'hard conditionally independent':

- *ID3*: 7.
- *table-majority*: 6.
- *IB1*, *oneR*, *IB4*, *T2*: 5.
- *NBTree*, *C4.5*, *Ripper*: 3.
- *CN2*, *NB*: 2.
- *OC1*, *PEBLS*, *HOODG*: 1.

Looking at the extremes of the table, two types of behaviours can be differentiated:

- The trend of perfect memorizers [3] (*ID3* decision tree, nearest neighbor algorithms and *table-majority*) and ST algorithms to form more 'original' hypotheses than the rest of the algorithms must be noted. These algorithms tend to appear with fewer connections in the learned Bayesian networks.
- This contrasts with Bayesian classifiers (*NB* and *NBTree*) and algorithms which have a post-pruning strategy (*OC1*, *C4.5*, *Ripper* and *CN2*). They show this tendency in a lesser degree: their hypotheses do not tend to be different from the whole set of hypotheses. They mainly appear in the central part of the Bayesian networks, with a higher number of connections than perfect memorizers and ST.

4.2. Conditional independencies in a proposed family of algorithms

Number of databases where algorithms of a family are conditionally independent, given one algorithm from another family:

[2] *PEBLS* and *HOODG* are only run in four of eleven databases.

[3] Perfect memorizer algorithms are very faithful with respect to the training set when they induce the hypothesis, pruning or smoothing the formed model in small degree.

- Decision-Trees: 7.
- Simple-Trees: 7.
- Decision-Rules: 7.
- K-NN: 3.
- Bayesian-Approaches: 3.

Analysing the extremes of the former list, two types of behaviour can be differentiated:

- Decision-Tree, Simple-Tree and Decision-Rule families demonstrate a low degree of 'compactness' or similarity between their component algorithms.
- This constrasts with K-NN and Bayesian-Approaches family behaviour. These families show a more 'compact' behaviour than the former.

4.3. Conditional independencies between proposed families

Number of databases where conditional independencies between families of algorithms appear: [4]

- I(Bayesian-Approaches, Simple-Trees | –): 7.
- I(Decision-Trees, K-NN | –): 6.
- I(Simple-Trees, Decision-Trees | –): 5.
- I(Simple-Trees, Decision-Rules | –): 5.
- I(Bayesian-Approaches, Decision-Trees | –): 5.
- I(Bayesian-Approaches, K-NN | –): 4.
- I(Decision-Rules, K-NN | –): 3.
- I(Simple-Trees, K-NN | –): 3.
- I(Decision-Rules, Bayesian-Approaches | –): 3.
- I(Decision-Trees, Decision-Rules | –): 2.

Regarding the former list, for each family, assertions about its most similar and dissimilar families can be extracted for tested domains.

5. Resum and future work

From a homogeneous set of databases, we have carried out a study of the joint behaviour of the predictions made by a set of Machine Learning algorithms. Bayesian networks, induced from the learning algorithms class predictions, were used to research the behaviour of a set of known algorithms. From the obtained Bayesian networks, guided by the conditional independence concept, relations between the probability distributions of the hypotheses formed by different algorithms were found. Three different types of relations have been studied:

- given an algorithm, the conditional independence of another algorithm with the rest of algorithms;
- conditional independence of two algorithms of the same family, given another algorithm from another family;
- conditional independence between families of algorithms, given any set of algorithms that do not belong to any of the considered families.

In other approaches to research, unsupervised hierarchic classification can be used over the set of predictions of Machine Learning supervised algorithms to determine clusters or families of algorithms. Statistical tests can also be used to study the hypotheses formed by algorithms.

For further reading, see (Lauritzen, 1996).

Discussion

Brailovsky: The results that you presented are very interesting. Before you can ascribe certain properties to an algorithm you need to check a lot of things, for example the independence, for a given problem, of the training and test sample set. Have you done this?

Inza: You are right! I mentioned that we only extracted assertions or guidelines on different types of behavior. It is true that for each training- and test-set and different proportions the results are different. For that reason we have tried to make our research in 11 different domains and if we saw the same behavior in a number of domains, we extracted assertions or guidelines based on the number of different domains where the characteristic appeared.

Acknowledgements

This work was supported by the grant PI 96/12 from the Gobierno Vasco – Departamento de Educación, Universidades e Investigación.

[4] By the '–' symbol, we represent any set of algorithms that do not belong to any of the compared families.

References

Aha, D., Kibler, D., Albert, M.K., 1991. Instance-based learning algorithms. Machine Learning 6, 37–66.

Andersen, S.K., Olesen, K.G., Jensen, F.V., Jensen, F., 1989. HUGIN – a shell for building Bayesian belief universes for expert systems. In: 11th International Joint Conference on Artificial Intelligence, pp. 1128–1133.

Auer, P., Holte, R., Maass, W., 1995. Theory and applications of agnostic PAC-learning with small decision trees. In: Prieditis, A., Russell, S. (Eds.), Machine Learning: Proceedings of the 12th International Conference, Morgan Kaufmann, Los Altos, CA.

Bouckaert, R.R., 1995. Bayesian belief networks: from construction to inference. Ph.D Thesis, Department of Computer Science, Utrecht University, The Netherlands.

Cestnik, B., 1990. Estimating probabilities: a crucial task in machine learning. In: Proceedings of the European Conference on Artificial Intelligence, pp. 147–149.

Clark, P., 1998. Personal communication.

Clark, P., Nibblet, T., 1989. The CN2 induction algorithm. Machine Learning 3 (4), 261–283.

Cohen, W.W., 1995. Fast effective rule induction. In: Machine Learning, Proceedings of the 12th International Conference.

Cooper, G.F., Herskovits, E.A., 1992. A Bayesian method for the induction of probabilistic networks from data. Machine Learning 9, 309–347.

Cost, S., Salzberg, S., 1993. A weighted nearest neighbor algorithm for learning with symbolic features. Machine Learning 10 (1), 57–78.

Dawid, A.P., 1979. Conditional independence in statistical theory. Journal of the Royal Statistics Society Series B 41, 1–31.

Duda, R., Hart, P., 1973. Pattern Classification and Scene Analysis. Wiley, New York.

Etxeberria, R., Larrañaga, P., Picaza, J.M., 1997. Analysis of the behaviour of genetic algorithms when learning Bayesian networks structure from data. Pattern Recognition Letters 18 (11–13), 1269–1273.

Heckerman, D., 1995. A tutorial on learning with Bayesian networks. Technical Report, MSR-TR-95-06.

Heckerman, D., Geiger, D., Chickering, D.M., 1995. Learning Bayesian networks: the combination of knowledge and statistical data. Machine Learning 20, 197–243.

Herskovits, E., Cooper, G., 1990. Kutató – An entropy-driven system for construction of probabilistic expert systems from databases. Report KSL-90-22, Knowledge Systems Laboratory, Medical Computer Science, Stanford University.

Holte, R.C., 1993. Very simple classification rules perform well on most commonly used databases. Machine Learning 11, 63–90.

Kohavi, R., 1995a. Wrappers for performance enhancement and oblivious decision graphs. Ph.D. Thesis, Standford University, Computer Science Department, STAN-CS-TR-95-1560.

Kohavi, R., 1995b. The power of decision tables. In: Lavrac, N., Wrobel, S. (Eds.), Proceedings of the European Conference on Machine Learning. Lecture Notes in Artificial Intelligence, Vol. 914. Springer, Berlin, pp. 174–189.

Kohavi, R., 1996. Scaling up the accuracy of Naive-Bayes classifiers: a decision-tree hybrid. In: Proceedings of the Second International Conference on Knowledge Discovery and Data Mining.

Kohavi, R., Sommerfield, D., Dougherty, J., 1997. Data mining using MLC++, a Machine Learning Library in C++. International Journal of Artificial Intelligence Tools 6 (4), 537–566.

Kreiner, S., 1989. Graphical modelling using DIGRAM. Research Report 89/11. Statistical Research Unit. University of Copenhagen.

Lam, W., Bacchus, F., 1994. Learning Bayesian belief networks. An approach based on the MDL Principle. Computational Intelligence 10 (4).

Larrañaga, P., Poza, M., Yurramendi, Y., Murga, R.H., Kuijpers, C.M.H., 1996. Structure learning of Bayesian networks by genetic algorithms: A performance analysis of control parameters. IEEE Transactions on Pattern Analysis and Machine Intelligence 18 (9), 912–926.

Lauritzen, S.L., Dawid, A.P., Larsen, B.N., Leimer, H.G., 1990. Independence Properties of Directed Markov Fields. Networks 20, 491–505.

Lauritzen, S.L., 1996. Graphical Models. Oxford University Press, Oxford.

Mitchell, T.M., 1997. Machine Learning. McGraw-Hill, New York.

Murphy, P.M., Aha, D.W., 1994. UCI Repository of Machine Learning databases. http://www.ics.uci.edu/mlearn/MLRepository.html. Irvine, CA. University of California, Department of Information and Computer Science.

Murthy, S.K., Salzberg, S., 1994. A system for the induction of oblique decision trees. Journal of Artificial Intelligence Research 2, 1–33.

Pearl, J., 1988. Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference. Morgan Kaufmann, Los Altos, CA.

Quinlan, J.R., 1986. Induction of decision trees. Machine Learning 1, 81–106.

Quinlan, J.R., 1989. Inferring decision trees using the minimum description length principle. Information and Computation 80, 227–248.

Quinlan, J.R., 1993. C4.5: Programs for Machine Learning. Morgan Kaufmann, Los Altos, CA.

Robinson, R.W., 1977. Counting unlabeled acyclic digraphs. In: Little, C.H.C. (Ed.), Lectures Notes in Mathematics 622: Combinatorial Mathematics V, Springer, New York, pp. 28–43.