Selection and Ranking of Attributes for Classification

Patricio Serendero[1], Miguel Toro[2]

pserende@ualg.pt, mtoro@lsi.us.es

[1]Dep. of Electronics & Computer Science, U. of Algarve, Campus Gambelas, Faro, Portugal

[2] Dep. of Languages and Inf. Systems, U. of Seville, Av. Reina Mercedes s/n, Seville, Spain

Abstract. Determining most relevant attributes and their input order is crucial in decision trees and instance-based methods for supervised learning. We present a new algorithm to identify and rank relevant attributes for the prediction model. Attribute values are projected into a one-dimensional space of equal width intervals. Restricted class frequencies are used as criterion to determine the degree of attribute relevance. Additionally, we show that increasing interval size helps to reduce the problem size. This low complexity algorithm shows a 4% increase in predictive accuracy when attributes are ordered using this technique for the same classification tool.

1 Introduction

Two well-known problems arise when using decision tree structures and instance-based algorithms for supervised learning. First, the attribute's input order determines heavily the predicting skills of the algorithm. Choosing the wrong order of attributes could move values apart in the hyperspace that otherwise would be closer.

Secondly, some attributes contribute more than others in building the prediction hypothesis [2], and attributes considered irrelevant increases the computational cost and can mislead distance metrics calculations [13]. This is particularly true for nearest neighbour algorithms [7]. Based on these, attributes are classified as *relevant* or *irrelevant*, in terms of their degree of contribution to the classification model[14, 15][1].

Intuitively, one wants to set first in the input order, attributes with larger discriminatory power with respect to classes, as done for instance with some rule induction algorithms [22, 6, 7, 23].

The complexity of feature selection algorithms depends on the number and quality of its attributes. Searching relevant attributes cannot be exhaustive in many cases. The dimension of datasets is exponential in the number of attributes. Hence, verifying every other combination of attributes is, in many cases, out of the question [17].

In this paper, we present a low computational and simple empirical algorithm for the supervised learning task in order to a) establish a criterion to decide which attributes are relevant. b) Which should be the best attribute input order for

[1] These authors still identify *redundant* attributes, a situation which we do not address here.

processing. c) How to reduce the number of intervals when discretization[2] is used. The overall goal is to diminish the classification algorithm's complexity as well as increasing or at least preserving its predictive skills.

2 Overview of the classification algorithm

We have previously developed an algorithm for supervised learning based on instances and the nearest neighbour paradigm [25]. For that purpose, we build a permanent multi-way tree (or "*trie*") to store discretized training patterns of the type $P = <p_1, p_2, p_i, p_n, c>$ where p_i is a previously discretized attribute value and c its class. Tree growth is done branching sequentially using each p_i value.

Classification is done extracting first from the tree two nearest patterns with respect to the unknown instance. Branching for pattern extraction is done at each node applying a normalized Euclidean distance. After distance comparison, the small one is chosen as the new p_i element. If distances are equidistant, frequencies are used as weight to break any ties. Next, the algorithm analyses the characteristics of all three patterns. It checks for the presence of *exclusive* and *semi exclusive* values; it also measures pattern *strength* and *frequency*. In datasets with a clear class bias distribution, a *majority class* parameter is also used. The class from the selected pattern is assigned to the new unseen instance.

3 Basic definitions

Let us consider the closed universe formed by a training data file composed of a finite set of records *r*. Each record is formed by a finite sequential set of attributes A_i, belonging to set *S*. Every attribute $A_i \hat{I} S$ can take v_i values belonging to a set T_i, called the domain value. Additionally, every record can be associated with classes $c_1, c_2,..,c_k$, belonging to a set L, where *k* is th e number of existing classes in the whole dataset. Hence, each record *r* is formed by the Cartesian product of attributes represented by the pair attribute/value and a class label *c*, such that:

$$r = <v_1, v_2,.., v_i,.., v_n, c > \; v_i \in T_i, c \in L . \quad (1)$$

Using function ord_i, we convert every attribute value $v_i \in r$ into pattern p_i to form pattern *p* as a sequence formed by *n* values. Every p_i value will fit into one of s_i partitions belonging to attribute A_i:

$$p = <p_1, p_2,.., p_i,.., p_n > p \in \{1..S\}, p_i = ord_i(v_i) ; p_i = \{1..s_i\} . \quad (2)$$

Notice that the number of partitions s_i is not the same for all attributes[3].

We define functions *pat(r)* and *label(r)* such that

[2] See Section 6

[3] This is due to changes in the number of partitions for selected attributes as we show later in section 6.

$$pat(v) = p, \text{ if } p_i = ord_i(v_i) .$$

$$label(r) = c, \text{ if } r = \langle v_1, v_2, .., v_n, c\rangle . \quad (3)$$

In every pattern *p* from equation (2) we can find *n* sub-patterns q_i, which can be viewed as the prefix portion of pattern *p*

$$q_i = \langle p_1, p_2, .., p_i\rangle, \; q_i \text{ a subsequence of p, } i = \{1..i\} . \quad (4)$$

We define function *freq*, which returns the number of records with pattern *p*.

$$freq\,(p) = \left| \{ r \in R \mid pat(v) = p \} \right| \text{ if } r = \langle v, c\rangle . \quad (5)$$

We can also apply this function to sub-patterns obtaining parameter λ:

$$\lambda_i = freq\,(q_i), \text{ the frequency of sub-patterns } q_i. \quad (6)$$

Every pattern *p* has a given label[4] *c*. We define function *labels (p)*, which return the set of labels associate to the subset of records with pattern *p*.

$$labels\,(p) = \{ c \in L \mid \exists\, r \in R_T \bullet label(r) \} \; ; \; c = \{1..k\}. \quad (7)$$

The number of labels attached to a given pattern *p* is given by next function:

$$nlabels(p) = \left| labels(p) \right| . \quad (8)$$

We can extend this concept to sub-patterns *q* as follows:

$$(nlq)_i = nlabels\,(q_i) \; . \quad (9)$$

4 Ordering attributes

In general, our method ranks attributes by its capacity of predicting classes without taking into consideration other attributes from the original sequence. We postulate that this capacity increase, when for a given interval an attribute shows a larger frequency of values fully or predominantly associated with one class.
Our objective is *to find the most discriminative attributes from the point of view of usefulness to the predictor, with the purpose of improving its prediction accuracy* [12]. This heuristic criterion has been used successfully before. [18]

The basic assumption of our classification algorithm is that each pattern *p* is uniquely associated with one label *c* in the entire dataset. On the other hand, sub-patterns can be associated with more than one label. Short sub-patterns relating to only one class represent homogeneous regions within the data hyperspace. In contrast, sub-patterns requiring more attributes to become associated with one class represent areas of larger entropy with respect to class distribution. If a new instance to be classified falls into one homogeneous class region, its chances of correct classification increase. Most of its neighbours will share the same label. For this reason, one would like to be able to look at the entire data space $\Re^n$ from the viewpoint of attributes where more of these disjoint class areas are "visible". These are areas where short

[4] In this article we use indistinctively the words label or class and attribute or feature.

sub-patterns are fully or predominantly distinctive of at least one class. The shortest sub-pattern of this type is the one formed by one attribute. This corresponds to consider a single attribute as independent from the influence of all others in the dataset. We say that attributes that allow this type of view are more *relevant* than others. This idea is similar to editing by ordered orthographic projections of disjoint class regions [24].

In order to identify which attributes are *relevant*, let's first define the concepts of *exclusive* and *semi-exclusive interval* values within a given attribute partition.

Function ord_i define the interval $I_{ij} \in T_i$ belonging to attribute A_i as:

$$I_{ij} = \{x \in Ti \mid ord_i(x) = j\} . \quad (10)$$

Let be S_{ij} the set of records belonging to attribute A_i whose values fall into the interval I_{ij}.

$$S_{ij} = \{r \mid v_i \in I_{ij}\}. \quad (11)$$

Interval S_{ij} is *exclusive* if all registers within interval j share the same class. It is *semi-exclusive* if the percentage of registers for a given class c within interval j is equal or greater than a user-defined limit $\ddot{o}$, which corresponds to class support in local class probabilistic models [19]. Thus, $S_{ij}(c)$ is the set of patterns in S_{ij} with class label c. More precisely, functions *exclusive* and *semi-exclusive* can be defined as:

$$ex(I_{ij}) \Leftrightarrow \exists c \bullet (|(S_{ij}(c)| = |S_{ij}|) \; ; \; semex(I_{ij}, j) \Leftrightarrow \exists c \bullet |S_{ij}(c)| / |S_{ij}| \geq j \quad . \quad (12)$$

Using both previous definitions we can define now the *strength of an Attribute* as:

$$d_i = strengtha(A_i, j) = \frac{\sum j : 1..s_i \mid semex(I_{ij}, j) \bullet |S_{ij}|}{|R|} \quad . \quad (13)$$

Notice that when $j = 1$ function $ex(I_{ij}) = semex(I_{ij},\ddot{o})$. For both, exclusive and semi-exclusive cases, larger $\ddot{a}$ values means attributes that are more *relevant*. The opposite means *irrelevant* attributes. Based on this, our method orders all attributes by its decreasing degree of relevance. Fig. 1 illustrates an example of projections.

Attribute: Clump Thickness

S_{ij}	0	1	2	3	4	5	6	7	8	9	total	ä
$\ddot{e}_j$	77	32	62	49	79	20	12	23	11	43	408	54/408
Class	*	*	*	*	*	*	*	*	**4**	**4**	54	=0.132

Attribute: Uniformity of Cell Size

S_{ij}	0	1	2	3	4	5	6	7	8	9	total	ä
$\ddot{e}_j$	227	23	35	17	17	14	12	13	5	45	408	89/408
Class	*	*	*	*	**4**	**4**	*	**4**	*	**4**	89	=0.218

Note: An asterisk means more than one class exists for that partition; i.e. nlabels (p_i) > 1
Shadow areas represent frequencies for exclusive interval values (all corresponding to class 4).

Fig. 1. Attribute projection and strength calculation in a one-dimensional space. Cancer dataset

For each attribute in the dataset, the algorithm projects all training values into a one-dimensional space previously partitioned into equal width intervals. It then calculates the attribute's *ä* value considering exclusive intervals first. For instance, in the Cancer dataset, most relevant attribute is "*Bland Chromatin*" *(ä₁* = 0.257*)* and worse is attribute "*Single Epithelial Cell Size*" (*ä₉* = 0.039). Next, the computed *ä* values are ranked in decreasing order obtaining list *ß*, which represents all attributes ordered by their degree of relevance

$$ß = < ä_1, ä_{2,..,}ä_{i,...,}än > \mid \{ä_n > ä_{(n+1)}\} . \quad (14)$$

Attributes where $ä_i = 0$, are pushed in their original order to the end of the list. If the number of positive ä values in list ß are less than some minimum, say $\sqrt{n}$, then for all attributes where $ä_i = 0$, new complimentary $ä_i'$ values are calculated using now semi-exclusive interval values. A second list is created out of the remaining attributes using semi-exclusive intervals with a user-defined value for *ö*. Last, a final list is produced concatenating both previous lists, with the elements from the exclusive values list in first place. Elements within each sub-list are ranked by their *ä* value; all remaining attributes come afterwards. See the example of the Pendigits dataset next.

Table 1. Final attribute list formed by different types of intervals in Pendigits dataset

Interval types	Attribute Nº	Frequency
Exclusive	A_4	45
	A_2	26
	A_{14}	7
Semi-exclusive (ö = 75%)	A_6	284
	A_{12}	132
	A_{16}	43
	A_{10}	5
Irrelevant	A_{11}- A_{16}	0

5 Selection and Ranking of Attributes

The selection of attributes becomes an easy task after we have determined which attributes are relevant and which are not. Irrelevant attributes are eliminated, thus reducing the algorithm's complexity. But this reduction cannot be done without a cost. The trade-off is done at the expense of losing predictive accuracy. For this reason our goal is *to find a minimum subset of attributes S' such that when the classification algorithm is applied we can obtain a new predictive accuracy P´:*

$$S' <= S, \text{ that satisfied } P' <= P + å \ . \quad (15)$$

The new set *S'* obtained by list *ß* from (14) includes only *relevant* attributes, and discard all *irrelevant* ones as it happens with attributes 11 till 16 in Table 1. The classification algorithm rebuilds the tree as well as other parameters. At running time, and using some user-accepted error å, say 4% over the existing prediction value *P*, a new *P'* value is obtained. If this value is within the established restriction by equation

(15), then *S'* is adopted as the new set of attributes to be used. Otherwise, the selection algorithm described in section 4 will be run again reducing the class probabilistic threshold value *ö*. The new goal is to increase the number of elements in list â and consequently the number of attributes in subset *S*'. See Table 4 for results on attribute reduction.

The actual selection algorithm not only discriminates the attribute type. All ä elements from Equation (14) are ranked in decreasing order according with their value, such that most relevant attributes can be processed first. Therefore, ranking attributes is a net requirement of the selection process.

6 Increasing Attribute Intervals

Different data attributes can have a different domain space due to the fact of having different data types. Because our classification algorithm and the ordering of attributes are based in a nominal feature space, all attributes are first discretized. We show first the method used for discretization. Next, we argue that in some cases increasing interval sizes reduce tree dimension and can improve predictive accuracy.

Discretization is the process of transforming the domain of a continuous attribute or feature into a finite number of intervals [15]. We use the *Equal Interval Width* single feature discretization method [9] without taking into account domain specific information [1]. Each partition within domain T_i is divided into s_i equal sized user-defined intervals ($s_i > 0$)[5], where M_i and m_i are the maximum and minimum values in the partition. For attributes represented by numeric values the method computes the value of interval *z* is calculated as:

$$z = ord_i(v_i) = \frac{v_i - m_i}{M_i - m_i} \times s_i \quad . \tag{16}$$

For categorical domain attributes, $ord_i(v_i) = p_i$, where p_i is the order number for v_i. Some of the reasons for using discretization other than our own algorithmic requirement are its simplicity and a lower computational cost. To partially avoid the loss of information produced by discretization, we keep class distribution information in a separate file, which is used at classification time.

Increasing the size of numeric intervals means considering larger and larger areas in the n^{th} dimensional space. Besides the beneficial effect of decreasing tree size and computational cost as well as diminishing the effect of noisy data, we are interested in maintaining or increasing prediction accuracy as well. Look at the empty interval 3 in Fig.2. After increasing the interval size we apply the *principle of continuity*[6] to patterns and their classes.

[5] In the actual implementation this information is stored into a dictionary file.

[6] The principle states that if, from the nature of a particular problem, a certain number of solutions are expected, then there will be the same number of solutions in all cases, even including imaginary solutions.

Atribute Intervals	1	2	3	4	5	6	7	8
Class distribution	A	A	Nil	A	B	B	B	A
(a) Interval increased twice	A		A		B		*	
(b) Further increase using exclusive values	A				*			
(c) Increase using semi exclusives at 75%	A				B			

Fig. 2. Increasing interval size in one-dimension projection using classes A and B.
In case (a) the existing interval was doubled. Case (b), *z* is further increased using exclusive values. Case (c) is the same as (b) but using semi-exclusive values

Some algorithms set interval boundaries on the basis of information gain criterion [23], class information entropy [11] or equal-frequency intervals [15] among others. In our case, we use heuristics to increase the size of interval values based on the predictive accuracy of the algorithm. After each interval modification, the classification algorithm is run again to verify if an increase in prediction has occurred.

7 Results

We have tested these techniques on seven datasets from the UCI repository [23]. All records with unknown attribute values were eliminated. In datasets: Forest covert, Adult and Pendigits, we used the given number of records for training and test. In all others we ran the algorithm ten times, randomly splitting the full dataset every time in 60% for training and 40% for test records. Accuracy results are shown next.

Table 2. Predictive accuracy with new attribute order using original interval size

Datasets						Accuracy (%)		Variation
Nº	Num. Attrib	Nome	Nº records. training	Nº records. test	New Attribute Order (**Negrita** when relevant)	Original	Ordered	%
1	13	Adult	28,468	15,060	**11,12,6,10,1**,13,2,3,4,5,7,8,9	68.7	72.7	+4.0
2	12	Forest covert	15,120	565,892	**1,10,5,6,4,12,8,7,9,3**,11,2	71.8	74.3	+2.5
3	10	Cancer–W.	407	273	**7,2,1,8,3,9,4,6,5**	94.5	95.2	+0.7
4	24	Hypothyroid	1598	1063	**18,23,21,1,20,22,7,5,13,24**,19,17,16,15,14,12, 11,10,9,8,6,4,3	97.2	99.2	+2.0
5	33	Dermatology	218	140	**20,22,27,29,6,12,8,25,33,34,24,15,10,31,26,30, 14,23,7**, 32,28,21,19,18,17,16,13,11,9,5,4,3,2,1	65.7	73.6	+7.9
6	8	Diabetes	462	306	**5,6,2,7,4,8,3,1**	65.4	71.9	+6.5
7	16	Pendigits	7494	3498	**4,2,14,6,12,16,10**,1,3,5,7,8,9,11,13,15(1)	53.4	58.4	+5.0
					Average variation due to attributes ordering			+4.1

(1) Using semi-exclusive values with ö = 75%

There is a low correlation between the number of attributes per dataset and the observed increases in Table 2. This is no surprise; the original attribute order done by

creators follows a criterion not equal in all cases. Moreover, the proportion of relevant and irrelevant attributes on each dataset is not equal.

Table 3. Variation in accuracy due to interval size increases. Using ordered sets

Nº	Dataset	New Interval Max value	Accuracy (%)		Variation (%)
			Original	Increased	
1	Adult	A1,A13,A4,A7 = 2; A11=10;A10=500	72.7	78.9	+6.2
2	Forest covert	A1=3;A10=20; A5=2; A6=20; A4=10	74.3	75.4	+1.1
3	Cancer –W.	A2 – A10 = 2	95.2	97.8	+2.6
4	Hypothyroid	A18=0.15; A23, A21, A1=2	99.2	99.3	+0.1
5	Dermatology	No changes possible (*)	73.6	73.6	0.0
6	Diabetes	A2=3; A5=3; A7=0.020; A6=0.10	71.9	77.8	+6.9
7	Pendigits	A1–A13=22; A14=12; A15–A16=2;	58.4	85.8	+27.4

(*) Most attributes are categorical. Therefore, is not possible to increase intervals.

Observed increases in Table 3 are similar to results obtained in datasets reported in a previous article where this technique was applied [25]. As expected, better results appear in datasets with numerical attributes. Dermatology and Hypothyroid datasets have mostly categorical attributes difficult to order, and hence the effect is minimum. In the Adult dataset, we have ordered some of the categorical attributes values making possible to increase interval sizes.

The explanation for these results could be better understood by looking at the example in Fig. 2. The new resulting interval size in (a) reinforces the assumption that when applying a distance metric, its value will relate to class *A*. Even more important is the effect on intervals 5 to 8 in the same example. Increasing interval size as in (c), and applying semi-exclusive values with *ö*= 75% simplifies label distribution to only two. Additionally, if exist "*outliers*" in the original interval nº 8, their negative effect would be eliminated altogether.

Table 4. Change in accuracy after reducing the number of attributes. Using ordered sets

Nº	Dataset	Number of attributes		Accuracy test set (%)		Variation
	Name	Original	Reduced	Ordered	Reduced	%
1	Adult	13	6	78.9	89.8	+10.9
2	Forest covert	12	5	75.4	76.1	+0.7
3	Cancer-W	9	4	97.8	98.2	+0.4
4	Hypothyroid	24	10	99.3	98.9	-0.4
5	Dermatology	33	19	73.6	80.2	+6.6
6	Diabetes	8	6	75.8	74.5	-1.3
7	Pendigits	16	11	85.8	83.0	-2.8

Decreasing the number of intervals can also have a strong impact in the algorithm's complexity by decreasing the search tree size. For instance, increasing the interval size for Adult and Cancer datasets using the figures from Table 2 decreases tree size of around 20% and 37% respectively.

Interval increase cannot be applied to categorical data that do not resist any ordering. The Dermatology dataset is an example of this as indicated in Table 3.

The largest absolute increases in Table 4 were observed in datasets nº 1 and 5. This is probably due to the presence of more irrelevant attributes in the original attribute set, eliminated afterwards. The decrease in the Diabetes dataset is explained by the fact that we eliminated 25% of *relevant* attributes as indicated in Table 2. The reduction in accuracy in Pendigits is counter-balanced by a significant reduction in tree size and hence, in search time. The small variation in accuracy observed in datasets 2, 3, and 4 is done at the expense of a reduction over 50% in the number of attributes. Finally, although out of the scope of this article and as a reference of our classification algorithm, we compare its results with Quinlan's landmark tool C4.5.

Table 5. Comparing our classifier with the popular C4.5

Nº	Dataset	Error in accuracy (%)	
		C4.5	Ours
1	Adult (Census 94,USA)	14.6	10.6
2	Forest cover	29.1	23.9
3	Cancer-W	5.8	1.8
4	Hypothyroid	0.7	0.7
5	Dermatology	3.9	19.8
6	Pima Indian Diabetes	26.8	24.2
7	Pendigits	7.2	14.2

Source for C4.5 results taken from [5, 16, 20, 21, 23].

In general, our classifier shows better performance than C4.5 for datasets with smaller class distribution and worse when the opposite is true.

8 Discussion

In this article we have presented a new attribute selection and ordering method useful for algorithms using decision trees and instance-based methods for supervised learning. Results from Table 3 show 4% average increase in accuracy for the classification algorithm when a new ordered attribute list is used. We have previously obtained similar results with other UCI datasets [25]. The basic reason for this is that reordering attributes and increasing interval sizes favours the criteria used by the classification algorithm. Having more relevant attributes located first in the tree hierarchy, favours the parameters used in the hypotheses model. The advantage of this algorithm is its simplicity and a polynomial degree of complexity.

Increasing interval sizes improved the degree of accuracy of the classification algorithm in all datasets where numerical attribute types are dominant. This is probably due to the fact that many datasets in the real world present some degree of natural clustering among its observations, which differs from attribute to attribute as well as within intervals themselves.

There is a trade off between the reduction in the dimension of attributes and the precision of the classification algorithm. For shorter sub-patterns means increasing the chances of having identical sub-patterns associated with more than one class. This situation represents a constraint when reducing dimensionality using this technique.

9 References

1.Aha, D. W., Bankert, R., A comparative evaluation of sequential feature selection algorithms, Proc. of the Fifth Intl. Workshop on Artificial Intelligence and Statistics, pp., 1-7, 1995.

2. Aha, D. W, Bankert, R., Feature Selection for Case-Based Classification of Cloud Types: An Empirical Comparison, In D.W. Aha (Ed.) Case-Based Reasoning: Workshop (Technical Report WS-94-01, CA, AAI Press, 1994

5. Chou, Y., Shapiro, L. G. A Hierarchical Multiple Classifier Learning Algorithm, Proceedings of the Intl. Conference on Pattern Recognition, Vol. 2, pp. 152-155. 2000.

6. Cover, T. Hart, P. Nearest neighbour pattern classification, IEEE Transactions on Information Theory, Vol. 13:1, pp 21-27, 1967

7. De Mántaras, R., A Distance-Based Attribute Selection Measure for Decision Tree Induction, Machine Learning, Vol. 6, pp. 81-92. 1991

9. Dougherty, J., Kohavi, R., Sahami, M., Supervised and Unsupervised Discretization of Continuous Features, Proceedings of the 12th Intl. Conference on Machine Learning, Morgan Kaufman Pub. San Fco., 1995

11. Fayyad, U., Kebi, I. On the Handling of Continuous-Valued Attributes in Decision Tree Generation, Machine Learning, Vol. 8, pp. 87-102, 1992

12. Guyon, I., Introduction to the NIPS 2001 Workshop on Variable and Feature Selection, BC. Canada, 2001.

13. Indyk, P., Dimensionality Reduction Techniques for Proximity Problems, Proc. 11th. ACM-SIAM Symposium on Discrete Algorithms, pp. 371,378, 2000

14. Kohavi, R, John, G., Wrappers for Feature Subset Selection, Artificial Intelligence Journal, Special issue on relevance, Vol. 97, Nº 1-2, pp 273-324, 1997

15. Lebowitz, M., Categorizing Numeric Information for Generalization. Cognitive Science, Vol. 9, pp. 285-308, 1985

16. Li, J., Dong, G., Ramamohanarao, K., Instance-Based Classification by Emerging Patterns, Proc. Fourth European Conf. On Principles and Practice of Knowledge Discovery in Databases, Springer-Verlag, pp. 191-200, 2000

17. Lesh, N., Zaki, M., Ogihara, M. Mining Features for Sequence Classification, MERL, Technical Report Number: TR98-22, 1998

18. Liu, B., Ma. Y, Wong, C.K, Improving an Association Rule Based Classifier, 4th European Conference on Principles and Practice of Knowledge Discovery in Databases, Lyon, Springer-Verlag, pp. 504-509, 2000

19. Meretakis, D., Lu, H., Wuthrich, B., A study on the performance of Large Bayes Classifiers, 11th European Conference on Machine Learning, Spain, 2000

20. Murphy, P.M., Aha, D.W., UCI Repository of machine learning databases Irvine, CA: University of California, Department of Information and Computer Science. Patrick M. Murphy (Repository Librarian), 1994.

21. Payne, T., Edwards P., Implicit Feature Selection with the Value Difference Metric, Proceedings of the 13th European Conference on Artificial Intelligence, ECAI-98, John Wiley & Sons, New York, NY, pp. 450-454. 1998

22. Quinlan, J., Induction of Decision Trees, M. L., Vol. 1, pp. 81-106, 1986

23. Quinlan, J. R., "Bagging, Boosting, and C4.5, University of Sydney., 1988

24. Riquelme, J., Aguilar, J. Editado por Proyección Ordenada: EPO, Department of Languages and Information Systems, University of Seville, Spain, 2000

25. Serendero, P., Toro, M., Supervised Learning Using Instance-based Patterns, Proceedings of the IX Conferencia de la Asociación Española para la Inteligencia Artificial, Vol. I, pp. 83-92, Spain, 2001