Projection-based measure for efficient feature selection

Roberto Ruiz*, José C. Riquelme and Jesús S. Aguilar-Ruiz
Department of Computer Science, University of Seville, Avda. Reina Mercedes S/n, 41012 Sevilla, Spain
E-mail: {rruiz, riquelme, aguilar}@lsi.us.es

Abstract. The attribute selection techniques for supervised learning, used in the preprocessing phase to emphasize the most relevant attributes, allow making models of classification simpler and easy to understand. Depending on the method to apply: starting point, search organization, evaluation strategy, and the stopping criterion, there is an added cost to the classification algorithm that we are going to use, that normally will be compensated, in greater or smaller extent, by the attribute reduction in the classification model. The method proposed in this work utilizes a measure based on projections to guide the selection of the attributes. The algorithm (SOAP: Selection of Attributes by Projection) has some interesting characteristics: lower computational cost (O(mn log n) m attributes and n examples in the data set) with respect to other typical algorithms due to the absence of distance and statistical calculations; its applicability to any labelled data set, that is to say, it can contain continuous and discrete variables, with no need for transformation. The performance of SOAP is analysed in two ways: percentage of reduction and classification. SOAP has been compared to CFS [4] and ReliefF [8]. The results are generated by C4.5 before and after the application of the algorithms.

1. Introduction

The data mining researchers, especially those dedicated to the study of algorithms that produce knowledge in some of the usual representations (decision lists, decision trees, association rules, etc.), usually make their tests on standard and accessible databases (most of them of small size). The purpose is to independently verify and validate the results of their algorithms. Nevertheless, these algorithms are modified to solve specific problems, for example real databases that contain much more information (number of examples) than standard databases used in training. To accomplish the final tests on these real databases with tens of attributes and thousands of examples is a task that takes a lot of time and memory size.

It is advisable to apply to the database preprocessing techniques to reduce the number of attributes or the number of examples in such a way as to decrease the computational time cost. These preprocessing techniques are fundamentally oriented to either of the next goals: feature selection (eliminating non-relevant attributes) and editing (reduction of the number of examples by eliminating some of them or calculating prototypes [1]). Our algorithm belongs to the first group.

In this paper we present a new method of attribute selection, called SOAP (Selection of Attributes by Projection), which has some important characteristics:

- Considerable reduction of the number of attributes.
- Lower computational time O(mn log n) than other algorithms.
- Absence of distance and statistical calculations: correlation, information gain, etc.
- Conservation of the error rates of the classification systems.

The hypothesis on which the heuristic is based is: "place the best attributes with the smallest number of label changes". The next section discusses related work. Section 3 describes the SOAP algorithm. Section 4 presents the results. Which deal with several databases from the UCI repository [3]. The last section summarises the findings.

*Corresponding author. Tel.: +34 95 455 38 67; Fax: +34 95 455 71 39.

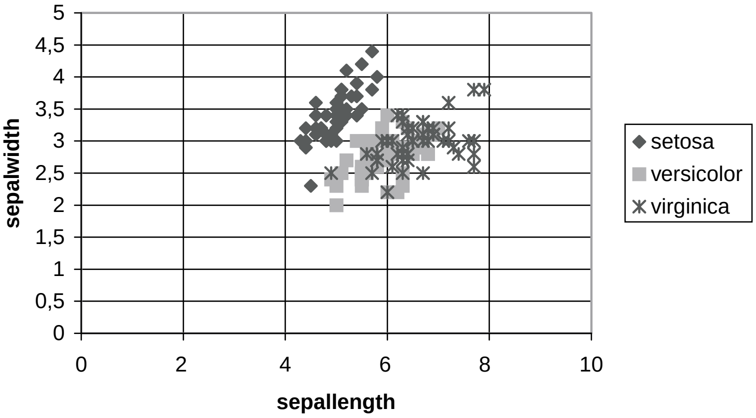

Fig. 1. Projection of Iris for the three classes on dimensions 1 and 2.

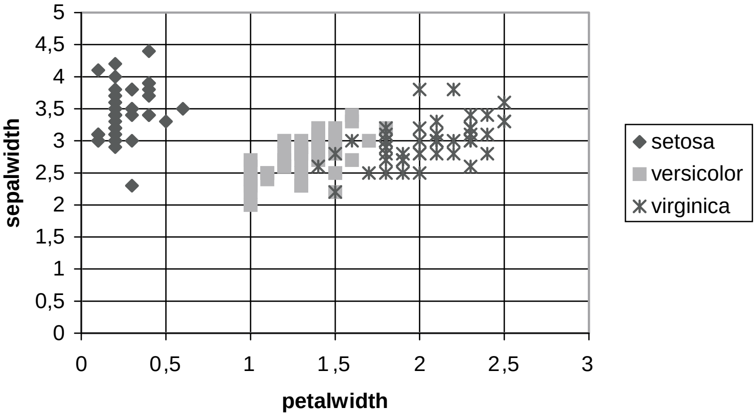

Fig. 2. Projection of Iris for the three classes on dimensions 4 and 2.

2. Related work

Several authors defined the feature selection by looking at it from various angles depending on the characteristic that we want to accentuate. In general, attribute selection algorithms perform a search through the space of feature subsets, and must address four basic issues affecting the nature of the search: 1) Starting point: forward and backward, according to whether it began with no feautures or with all features. 2) Search organization: exhaustive or heuristic search. 3) Evaluation strategy: wrapper or filter. 4) Stopping criterion: a feature selector must decide when to stop searching through the space of feature subsets. A predefined number of features are selected, a predefined number of iterations reached. Whether or not the addition or deletion of any feature produces a better subset, we also stop the search, if an optimal subset according to some evaluation function is obtained.

Algorithms that perform feature selection as a preprocessing step prior to learning can generally be placed into one of two broad categories: wrappers, Kohavi [7], which employs a statistical re-sampling technique (such as cross validation) using the actual target learning algorithm to estimate the accuracy of feature subsets. This approach has proved to be useful but is very slow to execute because the learning algorithm is called upon repeatedly. Another option called filter, operates independently of any learning algorithm. Undesirable features are filtered out of the data before induction begins. Filters use heuristics based on general the characteristics of the data to evaluate the merit of feature subsets. As a consequence, filter methods are generally much faster than wrapper methods, and, as such, are more practical for use on data of high dimensionality. FOCUS [3], LVF [14] use class consistency as an evaluation meter. One method for discretization called Chi2 [13]. Relief [6] works by randomly sam-

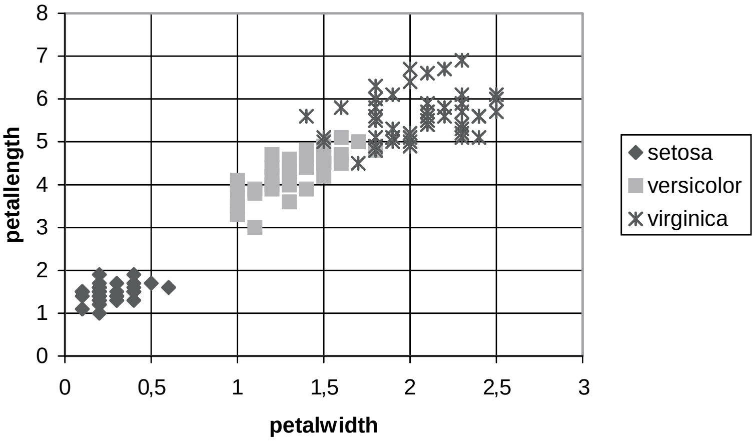

Fig. 3. Projection of Iris for the three classes on dimensions 4 and 3.

```
Input: E training (n examples, m attributes)
Output: E reduced (n examples, k attributes (k<=m))
         For each attribute $a_i$ with i in {1..m}
            $E_i$ ← QuickSort($E_i$,$a_i$)
            $NCE_i$ ← NumberChanges($E_i$,$a_i$)
         NCE Attribute Ranking
         Select the k first
```

Fig. 4. SOAP algorithm.

pling an instance from the data, and then locating its nearest neighbour from the same and opposite class. Relief was originally defined for two-class problems and was later expanded as ReliefF [8] to handle noise and multi-class data sets, and RReliefF [12] handles regression problems. Other authors suggest Neuronal Networks for an attribute selector [15]. In addition, learning procedures can be used to select attributes, like ID3 [10], FRINGE [9] and C4.5 [11]. Methods based on the correlation like CFS [4], etc.

3. SOAP: Selection of Attributes by Projection

3.1. Description

In this work a measure called Projections, between an attribute and the class is proposed. To describe the algorithm we will use the well-known data set IRIS, because of the easy interpretation of their two-dimensional projections.

Three projections of IRIS have been made in two-dimensional graphs. In Fig. 1 it is possible to observe that if the projection of the examples is made on the

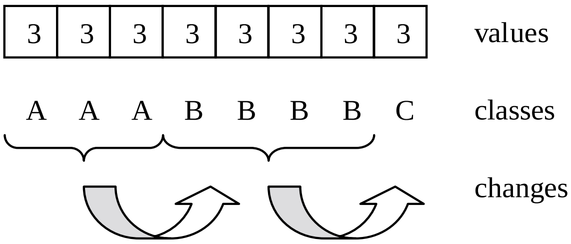

Fig. 6. Subsequence of the same value.

abscissas or ordinate axis we can not obtain intervals where any class is a majority, only can be seen the intervals [4.3,4.8] of Sepallength for the Setosa class or [7.1,8.0] for Virginica. In Fig. 2 for the Sepalwidth parameter in the ordinate axis clear intervals are not appraised either. Nevertheless, for the Petalwidth attribute is possible to appreciate some intervals where the class is unique: [0,0.6] for Setosa, [1.0,1.3] for Versicolor and [1.8,2.5] for Virginica. Finally in Fig. 3, it is possible to appreciate the class divisions, which are almost clear in both attributes. This is because when projecting the examples on each attribute the number of label changes is minimum. For example, it is possible to verify that for Petallength the first label change takes

```
Input: E training (n examples, m attributes)
Output: number of label changes
          For each example e_j ∈ E with j in {1..n}
             If att(u[j],i) ∈ Subsequence same value
                labelChanges += ChangesSameValue()
             Else
                If lab(u[j]) <> lab(u[j+1])
                   labelChanges++
```

Fig. 5. NumberChanges algorithm.

place for value 3 (setosa to Versicolor), the second in 4.5 (Versicolor to Virginica). there are other changes later in 4.8, 4,9, 5,0 and the last one is in 5.1.

SOAP is based on this principle: to count the label changes, produced when crossing the projections of each example in each dimension. If the attributes are in ascending order according to the number of label changes, we will have a list that defines the priority of selection, from greater to smaller importance. SOAP presumes to eliminate the basic redundancy between attributes, that is to say, the attributes with interdependence have been eliminated. Finally, to choose the more advisable number of features, we define a reduction factor, RF, in order to take the subset from attributes formed by the first of the aforementioned list.

Before formally exposing the algorithm, we will explain with more details the main idea. We considered the situation depicted in Fig. 2: the projection of the examples on the abscissas axis produces a ordered sequence of intervals (some of then can be a single point) which have assigned a single label or a set of them: {[0,0.6] Se, [1.0,1.3] Ve, [1.4,1.4] Ve-Vi, [1.5,1.5] Ve-Vi, [1.6,1.6] Ve-Vi, [1.7,1.7] Ve-Vi, [1.8,1.8] Ve-Vi, [1.9,2.5] Vi}. If we apply the same idea with the projection on the ordinate axis, we calculate the partitions of the ordered sequences: {Ve, R, R, Ve, R, R, R, R, R, R, R, R, R, R, Se, R, Se, R, Se}, where R is a combination of two or three labels. We can observe that we obtain almost one subsequence of the same value with different classes for each value from the ordered projection. That is to say, projections on the ordinate axis provide much less information that on the abscissas axis.

In the intervals with multiple labels we will consider the worst case, that being the maximum number of label changes possible for a same value.

The number of label changes obtained by the algorithm in the projection of each dimension is: Petalwidth 16, Petallength 19, Sepallenth 87 and Sepalwidth 120. In this way, we can achieve a ranking with the best attributes from the point of view of the classification.

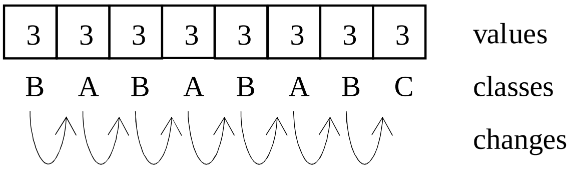

Fig. 7. ChangesSameValue.

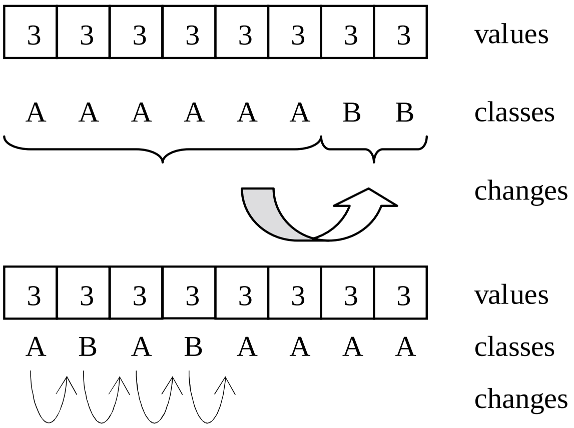

Fig. 8. Number of label changes.

This result agrees with what is common knowledge in data mining, which states that the width and length of petals are more important than those related to sepals.

3.2. Definitions

Definition 1. *Let the attribute A_i be a continuous or discrete variable that takes values in $I_i = [\min_i, \max_i]$. Then, A is the attributes space defined as $A = I_1 \times I_2 \times \ldots \times I_m$, where m is the number of attributes.*

Definition 2. *An example $e \in E$ is a tuple formed by the Cartesian product of the value sets of each attribute and the set C of labels. We define the operations att* *and lab to access the attribute and its label (or class):* $att : E \times N \to A$ *and lab:* $E \to C$, *where* N *is the set of natural numbers.*

Definition 3. *Let the universe U be a sequence of example from E. We will say that a database with n examples, each of them with m attributes and one class, forms a particular universe. Then* $U =< u[1], \ldots, u[n] >$ *and as the database is a sequence, the access to an example is achieved by means of its position. Likewise, the access to j-th attribute of the i-th example is made by* $att(u[i], j)$*, and for identifying its label* $lab(u[i])$.

Definition 4. *An ordered projected sequence is a sequence formed by the projection of the universe onto the i-th attribute. This sequence is sorted out in ascending order.*

Definition 5. *A partition in subsequences is the set of subsequences formed from the ordered projected sequence of an attribute in such a way as to maintain the projection order. All the examples belonging to a subsequence have the same class and every two consecutive subsequences are disjointed with respect to the class. Henceforth, a subsequence will be called a partition.*

Definition 6. *A subsequence of the same value is the sequence composed of the examples with identical value from the i-th attribute within the ordered projected sequence. This situation can be originated in continuous variables, and it will be the way to deal with the discrete variables.*

3.3. Algorithm

The algorithm is very simple and fast, see Fig. 4. It has the capacity to operate with continuous and discrete variables as well as with databases which have two classes or multiple classes. In the ascending-order-task for each attribute, the QuickSort algorithm is used [5]. This algorithm is O(n log n), on average. Once ordered by an attribute, we can count the label changes throughout the ordered projected sequence. NumberChanges in Fig. 5, considers whether we deal with different values from an attribute, or with a subsequence of the same value (this situation can be originated in continuous and discrete variables). In the first case, it compares the present label with that of the following value. Whereas in the second case, where the subsequence is of the same value, it counts as many label changes as are possible (function ChangesSameValue).

The k first attribute which NCE (number of label changes) under NCElim will be selected. NCElim is calculated applying the follow equation:

$$NCE_{\mathrm{lim}} = NCE_{\mathrm{min}} + (NCE_{\mathrm{max}} - NCE_{\mathrm{min}}) * RF \qquad (1)$$

RF: reduction factor.

After applying QuickSort, we might have repeated values with the same or different class. For this reason, the algorithm firstly sorts by value and, in case of equality, it will look for the worst of the all possible cases (function ChangesSameValue).

We could find the situation as depicted in Fig. 6. The examples sharing the same value for an attribute are ordered by class. The label changes obtained are two. The next execution of the algorithm may find another situation, with a different number of label changes. The solution to this problem consists of finding the worst case. The heuristic is applied to obtain the maximum number of label changes within the interval containing repeated values. In this way, the ChangesSameValue method would produce the output shown in Fig. 7, seven changes.

ChangesSameValue stores the relative frequency for each class within the interval. It is possible to be affirm that:

$$\begin{aligned} &\text{If } \exists rf_i > (\text{nelem}/2) \text{ them } (\text{nelem} - rfi) * 2 \\ &\text{Else nelem} - 1 \end{aligned} \qquad (2)$$

rf_i : relative frequency for each class, with i in $\{1, \ldots, k\}$ classes.

nelem: number of elements within the interval.

In Fig. 6 we can observe a subsequence of the same value with eight elements: three elements are class A, four class B and one C. Applying formula 2 there is no relative frequency greater than half of the elements. Then, the maximum number of label changes is nelem-1, seven. In Fig. 7 we verify it.

From Fig. 8 it can be seen that the function ChangesSameValue will return four label changes, because a relative frequency greater than nelem/2 exists (class A). Then, the result is $(8 - 6) * 2 = 4$. In this way, we always will find the maximum number of label changes within the interval containing repeated values.

This algorithm allows working with discrete variables. We consider each projection of this attribute like a subsequence of the same value.

Table 1
Number of selected features and the percentage of the original features retained

	Data	SOAP		CFS		RLF	
Data Set	Atts	Atts	%	Atts	%	Atts	%
autos	25	2.9	11.8	5.3	21.3	10.9	43.7
breast-c	9	1.5	16.7	4.1	45.9	3.7	41.6
breast-w	9	5.2	57.6	9.0	99.7	8.1	89.4
diabetes	8	2.8	34.9	3.1	38.9	0.0	0.0
glass2	9	3.2	35.7	4.0	43.9	0.3	3.6
heart-c	13	6.3	48.2	6.4	49.1	6.9	53.4
heart-stat	13	5.4	41.8	6.3	48.2	6.3	48.2
hepatitis	19	2.6	13.6	8.7	45.6	13.3	70.0
horse-c.OR.	27	2.3	8.6	2.0	7.4	2.3	8.6
hypothyroid	29	1.7	5.7	1.0	3.4	5.2	18.0
iris	4	2.0	50.0	1.9	48.3	4.0	100.0
labor	16	4.3	27.0	3.3	20.8	8.8	55.3
lymph	18	1.8	9.9	8.9	49.2	11.8	65.8
sick	29	1.0	3.4	1.0	3.4	7.1	24.5
sonar	60	3.0	5.0	17.8	29.7	3.9	6.5
vote	16	1.6	10.0	1.0	6.3	15.5	96.9
Average	19.0	3.0	23.7	5.2	35.1	6.8	45.3

4. Experiments

In order to compare the effectiveness of SOAP as a feature selector for common machine learning algorithms, experiments were performed using sixteen standard data sets from the UCI collection [3]. The data sets and their characteristics are summarized in Table 3. The percentage of correct classification with C4.5, averaged over ten ten-fold cross-validation runs, were calculated for each algorithm-data set combination before and after feature selection by SOAP (RF 0.35), CFS and ReliefF (threshold 0.05). For each train-test split, the dimensionality was reduced by each feature selector before being passed to the learning algorithms. The same fold were used for each feature selector-learning scheme combination.

To perform the experiment with CFS and ReliefF we used the Weka (Waikato Environment for Knowledge Analysis, http://www.cs.waikato.ac.nz/ ml) implementation.

Table 1 shows the average number of features selected and the percentage of the original features retained. SOAP is a specially selective algorithm compared with CFS and RLF. If SOAP and CFS are compared, only in one dataset (labor) is the number of characteristics significantly greater than those selected by CFS. In six data sets there are no significant differences, and in nine, the number of features is significantly smaller than CFS. Compare to RLF, only in glass2 and diabetes, SOAP obtains more parameters in the reduction process (threshold 0.05 is not sufficient). It can be seen (by looking at the fourth column) that SOAP retained 23,7% of the attributes on average. Figure 9 shows the average number of feature selected by SOAP, CFS and ReliefF as well as the number present in the full data set.

Table 2 shows the results for attribute selection with C4.5 and compares the size (number of nodes) of the trees produced by each attribute selection scheme against the size of the trees produced by C4.5 with no attribute selection. Smaller trees are preferred as they are easier to interpret, but accuracy is generally degraded. The table shows how often each method performs significantly better (denoted by ○) or worse (denoted by ●) than when performing no feature selection (column 2 and 3). Throughout we speak of results being significantly different if the difference is statistically at the 5% level according to a paired two-sided t test. Each pair of points consisting of the estimates obtained in one of the ten, ten-fold cross-validation runs, for before and after feature selection. For SOAP, feature selection degrades performance on seven datasets, improves on one and it is equal on eight. The results are similar to ReliefF and a little worse than those provided by CFS. Analyzing the datasets in which SOAP lost to CFS, we can observe breast-c, lymph and sonar, where the number of feature selected by SOAP is 25% of CFS (breast-c 4,1 to 1,5 with SOAP, lymph 8,9-1,8 and sonar 17,8-3). Nevertheless the accuracy reduction is small: breast-c 72,9 (CFS) to 70,24 with SOAP, lymph 75,95–72,84 and sonar 74,38–70,05.

It is interesting to compare the speed of the attribute selection techniques. We measured the time taken in milliseconds to select the final subset of attributes (This is a rough measure. Obtaining true cpu time from within a Java program is quite difficult) SOAP is an

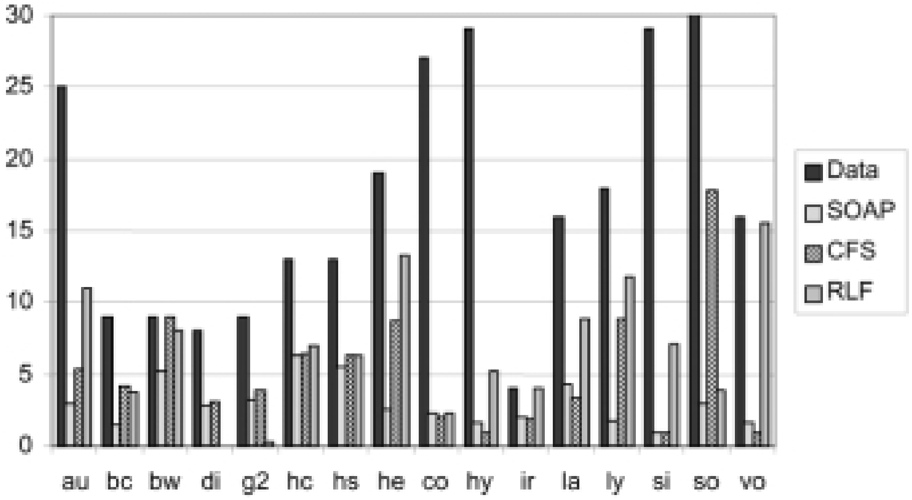

Fig. 9. Average number of feature selected by all the methods in comparison.

Table 2
Result of attribute selection with C4.5. Accuracy and size of trees. ○,● Statistically significant improvement or degradation ($p = 0.05$)

	Original		SOAP		CFS		RLF	
Data set	Ac.	Size	Ac.	Size	Ac.	Size	Ac.	Size
autos	82.54	63.32	73.37●	45.84○	74.54 ●	55.66○	74.15●	85.74●
breast-c	74.37	12.34	70.24●	6.61○	72.90	18.94●	70.42●	11.31
breast-w	95.01	24.96	94.64	21.28○	95.02	24.68	95.02	24.68
diabetes	74.64	42.06	74.14	7.78○	74.36	14.68○	65.10●	1.00○
glass2	78.71	24.00	78.96	14.88○	79.82	14.06○	53.50●	1.70○
heart-c	76.83	43.87	77.06	34.02○	77.16	29.35○	79.60○	28.72○
heart-stat	78.11	34.58	80.67○	19.50○	80.63○	23.84○	82.33○	14.78○
hepatitis	78.97	17.06	80.19	5.62○	81.68○	8.68○	80.45	11.26○
horse-c.OR.	66.30	1.00	66.30	1.00	66.30	1.00	66.28	1.36●
hypothyroid	99.54	27.84	95.02●	4.30○	96.64●	5.90○	93.52●	12.52○
iris	94.27	8.18	94.40	8.12	94.13	7.98	94.40	8.16
labor	80.70	6.93	78.25	3.76○	80.35	6.44	80.00	5.88
lymph	77.36	28.05	72.84●	7.34○	75.95	20.32○	74.66	24.10○
sick	98.66	49.02	93.88●	1.00○	96.32●	5.00○	93.88●	1.00○
sonar	74.28	27.98	70.05●	7.00○	74.38	28.18	70.19●	9.74○
vote	96.53	10.64	95.63●	3.00○	95.63●	3.00○	96.53	10.64
Average	82.93	26.36	80.98	11.94	82.24	16.73	79.38	15.79

algorithm with a very short computation time. The results shown in Table 3 confirm the expectations. SOAP takes 400 milliseconds in reducing 16 datasets whereas CFS takes 853 seconds and RLF more than 3 minutes. In general, SOAP is faster than the other methods and it is independent of the classes number. Also it is possible to be observed that ReliefF is affected very negatively by the number of instances in the dataset, it can be seen in "hypothyroid" and "sick". Eventhough these two datasets were eliminated, SOAP is more than 3 times faster than CFS, and more than 75 times than ReliefF.

Figure 10 summarizes the power of our algorithm, SOAP. It diminishes, in a significant percentage, the number of attributes, obtaining simple classification models, with a computational time lower than that of the other methods and with a similar average accuracy.

5. Conclusions

In this paper we present a deterministic attribute selection algorithm. It is a very efficient and simple method used in the preprocessing phase. A considerable reduction of the number of attributes is produced in comparison to other techniques. It does not need distance nor statistical calculations, which could be very costly in time (correlation, gain of information, etc.).

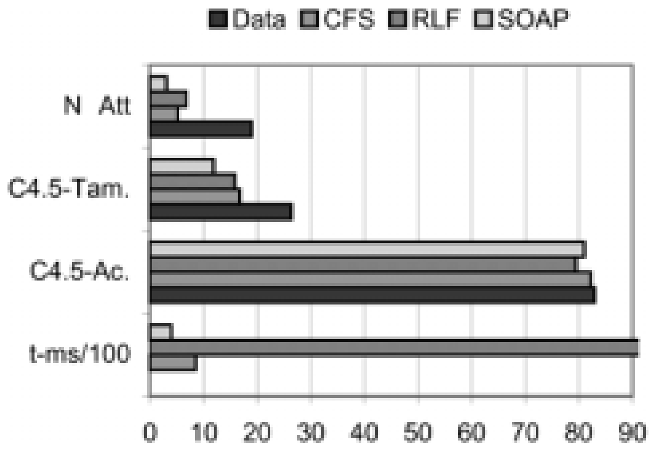

Fig. 10. Summary.

Table 3
Characteristic and time in milliseconds

		Original				
Data set	Instances	Att's	Classes	t-ms	t-ms	t-ms
Autos	205	25	7	15	50	403
breast-c	286	9	2	4	6	174
breast-w	699	9	2	6	35	1670
diabetes	768	8	2	6	39	1779
glass2	163	9	2	2	9	96
heart-c	303	13	5	6	10	368
heart-stat	270	13	2	4	12	365
hepatitis	155	19	2	4	9	135
horse-c.OR.	368	27	2	16	43	941
hypothyroid	3772	29	4	180	281	94991
Iris	150	4	3	3	3	44
Labor	57	16	2	1	3	21
Lymph	148	18	4	3	7	109
Sick	3772	29	2	120	252	93539
Sonar	208	60	2	21	90	920
Vote	435	16	2	9	4	651
Sum				400	853	196206

The datasets and their characteristics, and time taken in milliseconds to select the final subset of attributes.

The computational cost is lower than other methods O(m·n·log n).

In later works, we will focus our research on the selection of the subset of attributes once they have been obtained. Finally we will try to adapt SOAP to databases where redundant features have not been eliminated.

Acknowledgments

This work has been supported by the Spanish Research Agency CICYT under grant TIC2001-1143-C03-02.

References

[1] Aguilar-Ruiz, S. Jesús, Riquelme, C. José and Toro, Miguel, Data Set Editing by Ordered Projection, *Intelligent Data Analysis Journal* **5**(5) (2001), 1–13, IOS Press.

[2] H. Almuallim and T.G. Dietterich, Learning with many irrelevant features, *in Proceedings of the Ninth National Conference on Artificial Intelligence*, (1991), pp. 547–552, AAAI Press.

[3] C. Blake and E.K. Merz, *UCI Repository of machine learning databases* (1998).

[4] M.A. Hall, *Correlation-based feature selection for machine learning*, PhD thesis, Department of Computer Science, University of Waikato, Hamilton, New Zealand (1998).

[5] C.A.R. Hoare, QuickSort, *Computer Journal* **5**(1) (1962), 10–15.

[6] K. Kira and L. Rendell, A practical approach to feature selection, *in Proceedings of the Ninth International Conference on Machine Learning*, (1992) pp. 249–256, Morgan Kaufmann.

[7] R. Kohavi and G.H. John, Wrappers for feature subset selection, *Artificial Intelligence* **97** (1997), 273–324.

[8] I. Kononenko, Estimating attibutes: Analisys and extensions of relief, *in Proceedings of the Seventh European Conference on Machine Learning*, 1994, pp. 171–182, Springer-Verlag.

[9] G. Pagallo and D. Haussler, Boolean feature discovery in empirical learning, *Machine Learning* **5** (1990), 71–99.

[10] J. Quinlan, Induction of decision trees, *Machine Learning* **1**(1) (1986), 81–106.

[11] J. Quinlan, *C4.5: Programs for machine learning*, Morgan Kaufmann 1993.

[12] M. Robnik-Šikonja and I. Kononenko, An adaption of relief for attribute estimation in regression, *in Proceedings of the Fourteenth International Conference on Machine Learning*, 1997, pp. 296–304, Morgan Kaufmann.

[13] R. Setiono and H. Liu, Chi2: Feature selection and discretization of numeric attributes, *in Proceedings of the Seventh IEEE International Conference on Tools with Artificial Intelligence* 1995.

[14] R. Setiono and H. Liu, A probabilistic approach to feature selection-a filter solution, *in Proceedings of International Conference on Machine Learning* 1996, 319–327.

[15] R. Setiono and H. Liu, Neural network feature selectors, *IEEE Trans. On Neural Networks* **8**(3) (1997), 654–662.