# Ranking Attributes Using Learning of Preferences by Means of SVM

Alejandro Hernández-Arauzo[1], Miguel García-Torres[2], and Antonio Bahamonde[1]

[1] Universidad de Oviedo, Centro de Inteligencia Artificial Gijón, España
{alex,antonio}@aic.uniovi.es
[2] Universidad de La Laguna, Dpto. Estadística, I. O. y Computación, La Laguna, España
mgarciat@ull.es

**Abstract.** A relaxed setting for Feature Selection is known as Feature Ranking in Machine Learning. The aim is to establish an order between the attributes that describe the entries of a learning task according to their utility. In this paper, we propose a method to establish these orders using Preference Learning by means of Support Vector Machines (SVM). We include an exhaustive experimental study that investigates the virtues and limitations of the method and discusses, simultaneously, the design options that we have adopted. The conclusion is that our method is very competitive, specially when it searchs for a ranking limiting the number of combinations of attributes explored; this supports that the method presented here could be successfully used in large data sets.

## 1 Introduction

In Machine Learning, the *Feature Selection* problem in classification or regression tasks can be formulated as a combinatorial problem. The aim is to find the subset of attributes from which can be induced the best hypothesis; throughout this paper we will use feature and attribute as synonyms, both terms will mean the descriptors used to represent the entries of a data set. Frequently, in classification tasks, the quality of a hypothesis is measured by the estimation of the success rate in the prediction of new (unseen in training) cases; although it is also possible to consider other kind of indicators. For instance, the complexity of the learned model, or some measurements that combine the successes and errors costs when these are available in the context of the problem. In any case, the optimization of a set of attributes tries to improve both the quality of the hypothesis learned, as well as to reduce the cost of the training and acquisition of new cases to classify, as it was pointed out by Guyon and Elisseeff in [4].

When the selection of features is faced using a searching approach, the space, which represents the set of attribute subsets, has an exponential size with respect to the number of attributes. Therefore, in practice to solve the problem, the use of heuristics is required to guide the search through a reasonable number of

D. Borrajo, L. Castillo, and J.M. Corchado (Eds.): CAEPIA 2007, LNAI 4788, pp. 100–109, 2007.
© Springer-Verlag Berlin Heidelberg 2007

subsets. Eventually, these heuristics may lead to measure the usefulness of each feature by means of some function that only considers the values of the attribute and the class to learn. This is the case of *filters* that, in general, are less effective than methods that somehow evaluate the usefulness of subsets of more than one attribute [9]. The task of establishing an attribute *ranking* based on its prediction power is a relaxed formalization of the selection of features [8] since it leaves the effective selection of a subset of features to a later phase.

In this paper we present a method to determine an attribute ranking that is inspired by the strategies LEM (Learnable Evolution Model) [11] of Michalski, and BAYES-OPT (Bayessian Classifier based Optimization) [12] of Miquélez et al. These two methods start sampling a collection of subsets of attributes (called *population*) and estimating their quality. The next step consists in learning a pattern able to explain the improvements in quality measurements. In order to search for this pattern, the algorithms LEM and BAYES-OPT assign a label to each subset (each *individual*), in a qualitative scale. The subsets of attributes are labeled according to the estimation of quality of the hypothesis that can be learned using them to describe the entries of the learning task. So, LEM divides the subsets of attributes in three categories called *good*, *regular* and *bad*; the aim is to allow a set of rules, learned by an inducer of classification rules, to distinguish between *good* and *bad* individuals. On the other hand, BAYES-OPT uses a hybrid strategy between EDAs (Estimation of Distribution Algorithms) and LEM; it only considers the categories *good* and *bad* to induce a probabilistic graphical model. Once these methods have learned the way to distinguish between those subsets, and following an evolutionary strategy, both methods generate new populations using the knowledge just learned.

In the method presented in this article, only one sample of subsets of attributes is built. The core idea is that subsets of attributes can be ordered according to a quality measure, and then we can establish that we *prefer* those of better quality. Thus we only need to learn, from these *preference judgments*, a function that tries to assign higher values to more preferable objects, that is, subsets of attributes. In other words, we propose to tackle the problem of constructing a ranking of features as a preference learning task that will be finally solved using Support Vector Machines [5, 1, 7].

The rest of the paper is organized as follows. In section 2 we present the formal framework of the problem and we describe the proposed method to create an attribute ranking. Throughout this section we introduce two strategies that are simpler than our proposal. Next, we report the experimental results obtained to evaluate our proposal. Here, following the evaluation methodology used in [8] and [14], we will use artificial data sets described in section 3. Finally, the paper is closed with section 4 where we discuss the achievements and draw the conclusions of the article.

## 2 Computation of Feature Ranking

### 2.1 General Framework

Let $\mathcal{T} = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$ be a data set that represents a classification learning task, where vectors $\mathbf{x}_i \in \mathbb{R}^d$ are the objects to be classified, and they

are described by $d$ attributes or features; on the other hand, $y_i$ are the labels of a finite set of classes.

In order to construct a ranking depending on the usefulness in a classification learning task, we will construct a function able to assign a value

$$At(i) \in \mathbb{R}, \forall\ i = 1, \ldots, d. \tag{1}$$

The attribute ranking will be given by the list of attributes ordered according to the values $At(i)$.

As it was explained in the Introduction, our approach starts from a sample of attribute subsets endowed with a quality measurement. This measurement is an estimation of the performance that can be reached in the task $\mathcal{T}$. Formally, we build a set $\mathcal{M} = \{(\mathbf{z}_1, a_1), \ldots, (\mathbf{z}_m, a_m)\}$, where $\mathbf{z}_j$ are binary vectors that represent subsets of $\{1, \ldots, d\}$ randomly selected, while $a_j$ are quality measurements estimated with an external learning algorithm that had the set $\mathcal{T}$ as input but considering only the attributes in $\mathbf{z}_j$.

A first attempt at searching for a pattern in $\mathcal{M}$, that we will call *Simple Method* from now on, consists in defining, as the value associated to each attribute $i$, the average qualities measurements that appear in $\mathcal{M}$ in the cases where the $i$-th attribute is present. In symbols:

$$Simple\_At(i) = \frac{\sum_{j=1}^{m} a_j z_j(i)}{\sum_{j=1}^{m} z_j(i)}, \forall\ i = 1, \ldots, d. \tag{2}$$

Later we will see that this is an excessively naive method: obviously it can not capture all complex relationships among attributes. However we will verify that this method yields quite good results in some kind of problems. In any case, all ranking approaches should have to outperform the Simple Method.

A second criterion for attribute ranking can be drawn from a regression model. In fact, notice that $\mathcal{M}$ can be read as a regression task where the attributes values would be, in this case, binary. In order to approximate the $a_j$ values independently of the dimension of $\mathbf{z}_j$ we can use a Support Vector Regression (SVR) [16]. Then, if the learned function from $\mathcal{M}$ is

$$f(x) = \sum_{j=1}^{d} w_j x_j + b, \tag{3}$$

we define the value for each attribute as the absolute value of the weight of that attribute in the previous regression function. That is,

$$Reg\_At(i) = |\mathbf{w}_i|, \forall\ i = 1, .., d. \tag{4}$$

### 2.2 Support Vector Machines to Learn Preferences

The target of a preference learning task is a function able to order a set of objects. Training examples may have different origins, but in our case they will

be pairs of entries of $\mathcal{M}$ $(\mathbf{z}_i, \mathbf{z}_j)$ such that the first one has an higher estimation of quality than the second one; that is, $a_i > a_j$. Such pairs will be called a *preference judgment* and they all form a set that we will denote by

$$PJ = \{(\mathbf{z}_i, \mathbf{z}_j) : a_i > a_j, (\mathbf{z}_i, a_i), (\mathbf{z}_j, a_j) \in \mathcal{M}\}. \tag{5}$$

In this context, the aim of preference learning is to find a valuation function

$$f : \mathbb{R}^d \to \mathbb{R} \tag{6}$$

such that maximizes the probability that $f(u) > f(v)$ whenever $(u, v) \in PJ$. We will call $f$ a *preference*, *ranking* or *utility* function.

Although non-linear functions could had been used, as we will see in the experimental results reported in section 3, in this case linear functions are good enough to obtain competitive scores. Thus, in the linear case, $f$ is determined by a vector $\mathbf{w}$ called weight or director vector. The value of the function in a vector $\mathbf{z}$ will be the scalar product

$$f(\mathbf{z}) = \langle \mathbf{w}, \mathbf{z} \rangle = \sum_{j=1}^{d} w_j z_j. \tag{7}$$

In order to determine the function $f$ or the weight vector $\mathbf{w}$, we will follow the approach found in [5, 1, 7]; that is to say, we will consider each preference judgment as a constraint for a binary classification since for $(\mathbf{u}, \mathbf{v}) \in PJ$,

$$f(\mathbf{u}) > f(\mathbf{v}) \Leftrightarrow 0 < f(\mathbf{u}) - f(\mathbf{v}) = f(\mathbf{u} - \mathbf{v}) = \mathbf{w}(\mathbf{u} - \mathbf{v}). \tag{8}$$

Thus, the ranking function can be induced by a Support Vector Machine (SVM) [15] that seeks the hyperplane (with director vector $\mathbf{w}$) that pass through the origin and leaves, with the maximum separation margin, in the positive semi-space most of the vectors $\mathbf{u} - \mathbf{v}$, with $(\mathbf{u}, \mathbf{v}) \in PJ$. Finally, to get a ranking of attributes, as in the case of regression, we define

$$Pref\_At(i) = |\mathbf{w}_i|, \forall\ i = 1, .., d. \tag{9}$$

To apply this method we have to notice that the number of preference judgments that can be created from $\mathcal{M}$ is on the order of the square of the size of $\mathcal{M}$, in our case $O(m^2)$. In practice, fortunately, not all preference judgments are necessary and so we will consider different heuristics, that will be explained in the next section, to select a sample of pairs from $\mathcal{M}$ in order to build a reduced but representative set of preference judgments.

## 3 Experimental Results

### 3.1 Evaluation of Methods for Building Ranking of Attributes

To evaluate and compare algorithms for attribute ranking, sometimes are used collections of classification tasks taken from a well known and accepted by the

community repository, as it is the case of the UCI Machine Learning Repository. In these cases, the evaluation of ranking algorithms must be done by means of indirect methods. Since the *correct* order of the attributes is not known, the comparison is performed using the success rate in cross validation achieved for instance with the best 25, 50 or 100 attributes according to the order given by the algorithms involved in the comparison. Nevertheless this comparison method is not suitable: a good classification inducer could *fix* small errors of a ranking algorithm.

In order to overcome this problem we will use a direct method that evaluates the order given by each algorithm. This comparison method was previously used, for example, in [8] and [14]. The evaluation will be done considering only artificially generated data sets whose classification rules are known beforehand. The order given by ranking algorithms will then be compared with the correct order using the Area Under ROC Curve (AUC): the so called ROC-FS [8] that we will define later.

**Specifications and Construction of Data Sets.** We constructed data sets $\mathcal{T}$ with $n$ examples described by pairs $(\mathbf{x}_i, y_i)$, where each $\mathbf{x}_i$ is a vector described by $d$ quantitative variables, and its corresponding $y_i \in \{-1, +1\}$ is a qualitative attribute that stands for the associated class to the vector. So $x_{ij}$ will represent the $j$-th component of the $i$-th example.

The definition of the data sets were based on 5 parameters (see [8]) that specify the nature and difficulty of the corresponding learning tasks. Thus a data set is defined by means of the tuple $(n, d, r, l, \sigma)$, where $n$ represents the number of examples, $d$ the total number of attributes, $r$ the number of relevant attributes, $l$ the type of classification rule, and $\sigma$ the noise rate in the attributes.

An attribute is considered to be relevant to the learning task if it is present in the classification rule definition. In this case we considered two types of classification rules: linear ($l = 1$) and non-linear. For an example $i$ of the set $\mathcal{T}$, the definition of the linear classification rule [8] is:

$$y_i = \begin{cases} +1 \text{ if } \sum_{j=1}^{r} x_{ij} > \mu \\ \text{-1} \quad \text{otherwise} \end{cases} \tag{10}$$

where $\mu$ is the threshold given by $\mu = r/2$. For the non-linear case [14], we generated a $r \times 2$ $(c_{j,m})$ random matrix with coefficients in $[-2, -1] \cup [+1, +2]$. We used this range to avoid coefficients with values close to 0, which would falsify the subset of relevant attributes. Then, we build a polynomial of degree 2, and for each example $i$ we define:

$$p_i = \prod_{k=1}^{2} \Big( \sum_{j=1}^{r} (c_{jk} x_{ij}) + b_k \Big); \quad y_i = \begin{cases} +1 \text{ if } p_i > \mu \\ \text{-1} \quad \text{otherwise} \end{cases} \tag{11}$$

where $b_k$ is a random independent term to assure that all monomials of degree 1 and 2 are generated, and $\mu$ is the median of $p_i, i = 1, \ldots, n$.

Each element $x_{ij}$ were drawn uniformly in $[0, 1]$. The label $y_i$ of each example $\mathbf{x}_i$ was assigned considering the equations 10 and 11.

For the experiments carried out in this article, we used data sets with $n = 200$ examples, while the number of attributes varied from 25 to 700. The number of relevant attributes was fixed to $r = 10$ for all data sets. Additionally, in order to increase the difficulty, the input values of the data sets were perturbed by adding Gaussian noise drawn after $\mathcal{N}(\mu, \sigma)$, with average $\mu = 0$ and variance $\sigma = 0$, 0.05, and 0.10. We generated samples with $|\mathcal{M}| = 100$ and $|\mathcal{M}| = 200$. To take into account the stochastic nature of data, for each data set specification (set of parameters values), 20 different random data sets were constructed. Thus, a total of 3600 data sets were used for the experiments reported in this article. To estimate the quality of the hypothesis learned with each subset of attributes, we used the average classification success in a test set independently generated with the same size of the training set. The learning algorithm employed was Naïve Bayes [2].

**Area Under the ROC Curve.** ROC (Receiver Operating Characteristics) curves allow us to see and measure the performance of classifiers. Inspired by these curves, Jong et al. [8] propose to evaluate the correctness of an attribute ranking by means of the curves that they call ROC-FS. However, we will use a slight variant: ROC-FR [14], where FR stands for Feature Ranking.

Given a ranking of $m$ attributes, the ROC-FR curve is defined by the set of points

$$\{(FPR(i), TPR(i)) : i = 1, \ldots, d\}, \tag{12}$$

where $TPR(i)$ (respectively $FPR(i)$) stands for True (False) Positive Rate and it is calculated as the fraction of true (false) relevant variables whose position in the ranking is higher than $i$. The curve starts at position $(0, 0)$, and for each index the curve will advance in vertical if in that position of the ranking it is placed a relevant attribute; on the other hand, the curve will advance in horizontal whenever the $i$-th attribute of the ranking is an irrelevant one. Finally, the curve will end when the point $(1, 1)$ is reached. A perfect ordering would imply that all relevant attributes fill the highest positions of the ranking over any irrelevant attribute. In this case the Area Under this Curve would be $AUC = 1$. Let us remark that a random ordenation would have $AUC = 0.5$.

### 3.2 Two Comparison Stages

We have performed two types of comparisons. In the first one we compared different versions of the strategy proposed in this paper; the aim is to ratify the options selected during the design of the method. These versions are defined by the heuristic strategy employed to choose the pairs of examples to become preference judgments. The general approach consisted in dividing the examples of the sample according to the quality values $\{a_1, \ldots, a_m\}$. Then, the preference judgment were constructed comparing each example from a group with a (randomly selected) example from other group. We studied three heuristics; the first one divides the sample in 4 groups (or bins) with equal frequency: we called it $\text{SVM}_{quartiles}$. The other two heuristics built 4 and 10 bins in $[min\{a_j\}, max\{a_j\}]$ of equal length; we call them $\text{SVM}_{4Bins}$ and $\text{SVM}_{10Bins}$ respectively.

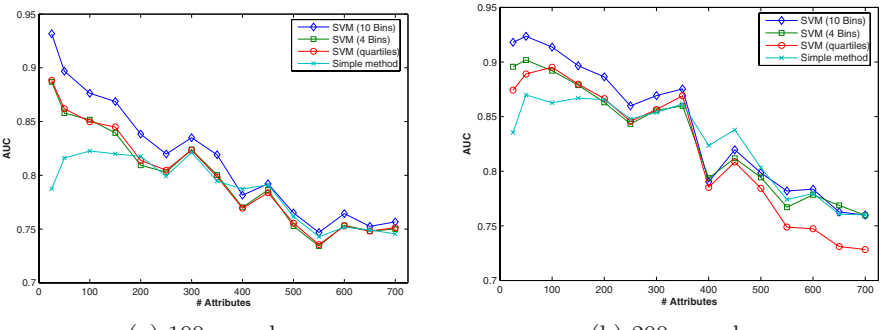
Once the *inner* comparisons were performed and the best version was selected, we compared it with a well-known ranking strategies: an evolutionary algorithm based on EDAs. The version considered was FSSEBNA [6], where the stopping criterion was modified so that it only evaluate a fixed number of individuals. It is necessary to remember that this method assigns a probability to each attribute based on the quality of the solutions in which it appears and this is done by means of probabilistics models. We considered that the higher is that probability for an attribute, the higher is the ranking position of the attribute. Additionally, we also included a simplified version of the approach proposed in this paper that uses regression (SVR from now on) as it was described in section 2.1. The objective is to show that it is not enough the estimations of qualities, in fact it is necessary to consider in the model the ordering relations among them. We called *outer* comparisons to this group experiments.

The Simple Method (see equation 2) was used in all comparisons as a reference to emphasize the results that can be obtained without any theoretical sophistication. Only results significantly higher than those reached by this method are worthwhile to be considered.

All differences reported in both types of comparisons are statistically significant according to a one tail t-test with threshold $p < 0.05$

### 3.3 Inner Comparisons

In figures 1(a) and 1(b), for each ranking method based on learning preferences with SVM, we show the evolution of the AUC values depending on the number of input attributes. As was pointed out in the previous section, the *Simple Method* was included as a baseline. We can observe that the version $\text{SVM}_{10Bins}$ significantly outperforms the other strategies, and it reaches AUC values higher than 0.8 for datasets of least or equal 400 attributes. For more than 400 attributes, $\text{SVM}_{10Bins}$ does not improve the scores of the *Simple Method*. This is the case both for samples of size $|\mathcal{M}| = 100$ and $|\mathcal{M}| = 200$.



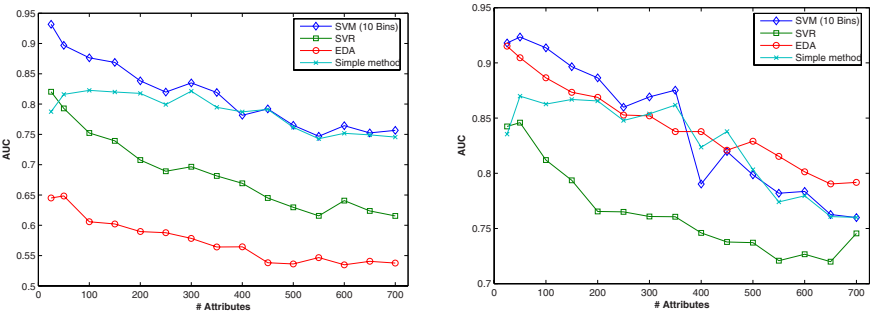

(a) 100 samples (b) 200 samples

**Fig. 1.** Comparison between different versions based on learning preferences with an SVM and the *Simple Method*

### 3.4 Outer Comparisons

In the previous subsection we have seen that $\text{SVM}_{10Bins}$ outperforms the other heuristics. Therefore, now we will compare the scores attained by preferences SVM endowed with this heuristic against other strategies not based on preferences that were already mentioned in sections 2 and 3.2.

Figures 2(a) and 2(b) show the comparison between $\text{SVM}_{10Bins}$, the raking based on EDA of Inza et al. [6], and the SVR version (recall Eq[4]). Again the *Simple Method* will be the baseline.




(a) 100 samples (b) 200 samples

**Fig. 2.** Comparison between the $\text{SVM}_{10Bins}$ and the methods based on EDA and SVR

With samples of size $|\mathcal{M}| = 100$ (figure 2(a)), the scores of the SVM and the *Simple Method* significatively outperform those obtained by EDA and SVR. Since the results of EDA and SVR are worse than those achieved by the *Simple Method*, we can conclude that they require much more individuals to produce quality results, specially in the case of EDA. Nevertheless the SVM method reaches AUC higher than 0.8 until data sets have 400 attributes; then the differences between SVM and the *Simple Method* are not significant. In general, we can say that the behaviour of the SVM is quite good when the sample size is small, getting results that significantly outperform those obtained with other strategies.

With 200 samples (figure 2(b)) the SVR method improves its scores slightly, but still they are too far from those achieved by the *Simple Method*; therefore, the regression is not suitable for these kind of problems. On the other hand, the EDA has noticeably improved the scores obtained when the sample size was $|\mathcal{M}| = 100$, (see figure 2(a)). In fact, for up to 150 attributes, the EDA provides better results than the *Simple Method*; nevertheless for data sets in the interval $[150, 450]$, there are no significant differences. Finally, for higher dimensional data sets the EDA becomes the algorithm with the best performance, providing AUC values higher than 0.75; moreover, the differences are statistically significant. Thus, we acknowledge that the EDA works well with data sets with a high number of irrelevant attributes. The SVM based on preferences is the best strategy for data sets up to 400 attributes as in the case of $|\mathcal{M}| = 100$. If the number of attributes increases, the results are similar to those of the *Simple Method*.

## 4 Conclusions

We have presented a method to induce a ranking of attributes based on learning preferences with an SVM. Given a classification learning task, the method starts with a population of attribute subsets endowed with a measurement of the quality of the hypothesis that can be induced using those attributes.

The benefits of the method were measured with an exhaustive collection of artificially generated data sets that cover both linear and nonlinear classification rules. The method has exhibited a very competitive behavior in data sets of up to 400 (10 relevant and 390 irrelevant) attributes. The performance of the method attains AUC values significantly better than those reached by the methods based on EDA or SVR. In particular when the amount of information available to learn the ranking is limited (populations of 100 individuals). Therefore, our approach is an interesting alternative in learning tasks where the acquisition of training examples is costly or difficult.

We would like to emphasize that the use of regression in these kind of problems can not capture the complexity of the ordering relations between the individuals of the population. Finally we can conclude that the EDA has a good behaviour in case of high dimensional problems (up to 500 attributes) with only 10 relevant attributes when the size of the population is large enough.

## Acknowledgments

We want to thank to the EDA authors [6] for providing us the code of the version of the algorithm based on EDAs that was used in the experimentation described in this article. This research has been partially supported under grants TIN2005-08288, TIN2005-08404-C04-03 from the MEC (Ministerio de Educación y Ciencia), and PI042004/088 from the Gobierno de Canarias.

## References

[1] Bahamonde, A., Bayón, G.F., Díez, J., Quevedo, J.R., Luaces, O., del Coz, J.J., Alonso, J., Goyache, F.: Feature subset selection for learning preferences: A case study. In: Greiner, R., Schuurmans, D. (eds.) Proceedings of the International Conference on Machine Learning (ICML '04), Banff, Alberta (Canada), pp. 49–56 (July 2004)

[2] Duda, R., Hart, P.: Pattern Classification and Scene Analysis. Wiley, Chichester (1973)

[3] Egan, J.P.: Signal Detection Theory and ROC Analysis. Series in Cognition and Perception. Academic Press, New York (1975)

[4] Guyon, I., Elisseeff, A.: An introduction to variable and feature selection. Journal of Machine Learning Research 3, 1157–1182 (2003)

[5] Herbrich, R., Graepel, T., Obermayer, K.: Large margin rank boundaries for ordinal regression. In: Advances in Large Margin Classifiers, MIT Press, Cambridge, MA (2000)

[6] Inza, I., Larrañaga, P., Etxeberria, R., Sierra, B.: Feature subset selection by Bayesian networks based optimization. Artificial Intelligence 123(1-2), 157–184 (2000)
[7] Joachims, T.: Optimizing search engines using clickthrough data. In: Proceedings of the ACM Conference on Knowledge Discovery and Data Mining (KDD). ACM, New York (2002)
[8] Jong, K., Mary, J., Cornuéjols, A., Marchiori, E., Sebag, M.: Ensemble feature ranking. In: Boulicaut, J.-F., Esposito, F., Giannotti, F., Pedreschi, D. (eds.) PKDD 2004. LNCS (LNAI), vol. 3202, pp. 267–278. Springer, Heidelberg (2004)
[9] Kohavi, R., John, G.H.: Wrappers for feature subset selection. Artificial Intelligence 97(1-2), 273–324 (1997)
[10] Larrañaga, P., Lozano, J.: Estimation of Distribution Algorithms. A New Tool for Evolutionary Computation. Kluwer Academic Publishers, Norwell, MA (2001)
[11] Michalski, R.: Learnable evolution model: Evolutionary processes guided by machine learning. Machine Learning, 9–40 (2000)
[12] Miquélez, T., Bengoetxea, E., Larrañaga, P.: Evolutionary computations based on bayesian classifiers. International Journal of Applied Mathematics and Computer Science 14(3), 101–115 (2004)
[13] Larrañaga, P., Lozano, J.A.: Synergies between evolutionary computation and probabilistic graphical models. International Journal of Approximate Reasoning, 155–156 (2002)
[14] Quevedo, J.R., Bahamonde, A., Luaces, O.: A simple and efficient method for variable ranking according to their usefulness for learning. In: Computational Statistics and Data Analysis (to appear, 2007)
[15] Vapnik, V.: Statistical Learning Theory. John Wiley, Chichester (1998)
[16] Vapnik, V., Golowich, S., Smola, A.: Support vector method for function approximation, regression estimation, and signal processing. In: Mozer, M.C., Jordan, M.I., Petsche, T. (eds.) Advances in Neural Information Processing Systems, vol. 9, pp. 281–287. MIT Press, Cambridge, MA (1997)