Feature selection based on bootstrapping

Norberto Díaz–Díaz, Jesus S. Aguilar–Ruiz, Juan A. Nepomuceno and Jorge García
BIGS, BioInformatics Group Seville
University of Seville, Spain
mail: {ndiaz, aguilar}@lsi.us.es

***Abstract*— The results of feature selection methods have a great influence on the success of data mining processes, especially when the data sets have high dimensionality. In order to find the optimal result from feature selection methods, we should check each possible subset of features to obtain the precision on classification, i.e., an exhaustive search through the search space. However, it is an unfeasible task due to its computational complexity. In this paper we propose a novel method of feature selection based on bootstrapping techniques. Our approach shows that it is not necessary to try every subset of features, but only a very small subset of combinations to achieve the same performance as the exhaustive approach. The experiments have been carried out using very high–dimensional datasets (thousands of features) and they show that it is possible to maintain the precision at the same time that the complexity is reduced substantially.**

I. Introduction

The success of the application of data mining algorithms is due to different factors. The quality of input data, for example, is one of these factors, in such a way that if data contain irrelevant or redundant information, or incorporate noise, then the learning process will be more difficult throughout the search space.

Feature selection methods allow to identify and eliminate part of the irrelevant or redundant information. This type of process lies in selecting a subset of optimum features, from input data, which maximizes the efficiency of data mining algorithm over the initial information.

A feature selection process is divided into four steps [1]: determining the possible features subset to carry out the problem representation, evaluating the features subset generated in previous step, checking if selected subset satisfies the search stopping criteria, and verifying the quality of features subset selected.

These processes can be classified in different ways depending on the step. If we analyze the selection function (second step), feature selection processes can be classified into three categories [2]: filters, wrappers [3], [4] and hybrid models [5]. In the filter model the selection procedure is fulfilled independently of the evaluation function (classification). The filter methods usually employ one of the following evaluation measures: distance, information, dependency and consistency. Examples of systems using these measures are: ReliefS [6], DTM [7], POE&AAC [8] and SOAP [9]. The wrapper model combines the search in the feature space with the learning algorithm, evaluating feature subsets and selecting the most appropriate one. They are costlier than the filters [3], although they usually obtain better results. The hybrid model attempts to take advantage of the two models by exploiting their different evaluation criteria in different search stages.

Independently of the evaluation function, the feature selection methods must carry out a search among the different candidates of features subsets. The search can be [2]: complete [10], sequential [11] or random. The complete search (or exhaustive) guarantees to find the optimum result according to the evaluation criterion. In contrast, it is computationally expensive ($\Theta(2^n)$), which makes that it is unapproachable when the number of features (n) is large. The sequential search, with cost $\Theta(n^2)$, is not complete and might not find the optimal subsets, because it is based on previous ranking, which has been produced by other techniques([7], [9], etc.). The random search, which does not assure the optimal either, starts with a randomly selected subset and proceeds with a sequential search (i.e., random–start [10]) or generating the next subset in a completely random manner (for example, Las Vegas algorithm [12]).

The bootstrap [13], in the context of classification, lies in replaying the whole classification experiment a large number of times and estimating the prediction accuracy from these replicated experiments. Thus, to estimate the error rate from few samples, a large number B of bootstrap replicated samples are created, each sample being a replication (randomly chosen) of the original sample. That is, a random sample of size m is taken from the original sample by sampling with replacement. Sampling with replacement means that some data points might be omitted. In addition, some data points will appear more than once in the bootstrap sample. Each bootstrap sample is used to build a classification rule which is then used to predict the classes of those original data that were unused in the training set. This gives one an estimate of the error rate for each bootstrap sample. The average error rate over all bootstrap samples are then combined to provide an estimated error rate for the original rule.

The exhaustive search is unfeasible in those cases in which the input data contain a large number of attributes. Thus, to compare our approach with the exhaustive one, we are going to limit the complete search according to the number of features which take part at each step, i.e., instead of analysing each possible combination of features, we are going to generate those combinations whose size (number of features involved) is not greater than a parameter value (K).

In this work, we propose a novel feature selection method that classifies the features in descending order according to

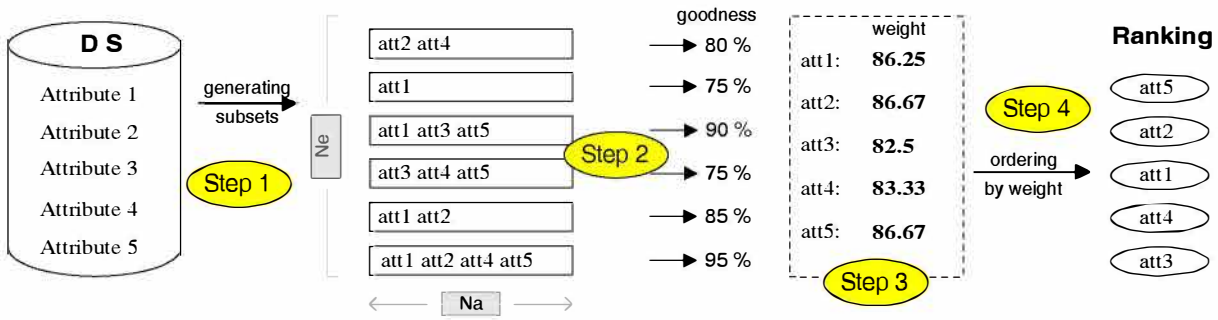

Fig. 1. Feature Selection Process Proposed.

their relevances. Our approach is based on the idea of reducing the cost of an exhaustive approach without compromising the precision of classification. With this aim, our method uses a random search based on the bootstrapping technique (sampling with replacement).

In broad outlines, the reminder of the paper is organized as follows. In Section II, the characteristics of our approach are described in detail. Later, in Section III, we discuss the results of our experiments and compare them to the exhaustive method (until size 3). At last, the conclusions are summarized in Section IV.

II. Description

The feature selection process presented in this paper can be divided into four steps: generating of features subset (*generation*), evaluating of each subset (*evaluation*), updating the weight of each feature (*update*) and ordering the features by their weights (*order*). In Figure 1 these steps are depicted over an artificial dataset with five features.

In the *first step*, the different subsets of features are generated. Each subset contains a maximum number of features which are randomly selected. They are chosen in such a way that the same attribute is not twice in the subset. Each subset will be generated independently, so it is possible to find two identical subsets. The number of generated subsets (Ne) and the maximum number of features in this (Na) are provided by the user. In Figure 1, these values are 6 subsets and 4 attributes as maximum, respectively. The three first subsets generated have been the following: an initial with two features (attribute2 and attribute4), with one (attribute1), and with three (attribute1, attribute3 and attribute5).

The *evaluation step* lies in classifying the original data with each feature subset generated in the previous step. The goal is to assign a value of merit to each subset, which is the average percentage of the classifications. As a result of this step, a goodness of 80 has been assigned to the first subset in Figure 1. The highest goodness generated have been 95 to the last subset, while the minimum, with value 75, is shared by the second and fourth subsets.

Feature weights are assigned in the *update step*. The weight of a feature f (P_f) is the average prediction accuracy of those subsets which contain f. For example, the attribute1 in Figure1 is present in the second, third, fifth and sixth subsets, so its weight is: $P_{a_1} = \frac{b_2+b_3+b_5+b_6}{4} = 86.25$, where b_i denotes the goodness of subset i.

The last step consists in ranking all the features in descending order according to their weights. The final ranking in the example of Figure 1 has been: attribute5, attribute2, attribute1, attribute4 and attribute3.

The main difference presented in the proposed method with respect to the exhaustive one resides in the first step. For example, an exhaustive method with $K = 1$ must generate one subset for each feature in the original data set. However, if we choose a value $K = 2$, it should generate a subset by each possible pair of features. So, the first step depends on the value of K for exhaustive methods, whereas the random methods only depends on the number of experiments we wish to do and the maximum number of features we wish to include in.

The overall approach is illustrated in Figure 2. The algorithm is divided into two functions. The first, Generate_Ranking, is the main function and it has the following input parameters: the classification method which will be used to evaluate each selected subset of features U; the subset of features X; the number of experiments to carry out Ne and the maximum number of features that will take part in each experiment Na. The only output parameter is a list L that corresponds to the feature ranking to be returned. In this way, Generate_Ranking classifies the features in descending order according to their weights. To calculate the weight of value, for each feature f, the average of correct classification (U) over those subsets containing f is calculated. These subsets of features are generated with the Generate_Combination_Attributes function (step1). According to the methodology described above, we could classify our approach into ranking techniques based on wrapper models.

III. Experiments

In order to show the performance of our approach with respect to the exhaustive method, we have selected four very high dimensional datasets (thousands of features). The characteristics of these datasets, which comprise gene expression data, are shown in Table I. Features represent genes, and examples represent conditions, so that, in each pair

```
Function Generate_Ranking
   Inputs:
      U: Evaluation Criteria;
      X: Attributes;
      Ne: num. Experiments;
      Na: num. Attributes
   Output:
      L: Attribute List (ranking)
 begin
   S := Generate_Combination_Attributes(X, Ne, Na)
   foreach subset of attributes S_i ∈ S
      C := Evaluate(S_i, U)
      Attribute_Update(S_i, C)
   end foreach
   L = Sort(X)
end Generate_Ranking
Function Generate_Combination_Attributes
   Inputs:
      X: Attributes;
      Ne: Num. Experiments;
      Na: maximum Num. Attributes
   Output:
      L: List of subsets of attributes
begin
   for i = 1 to Ne
      n := Generate_Index_Randomly(1, Na)
      S_i := Choose_Attributes(X, n)
      L := L + S_i
   end for
end Generate_Combination_Attributes
```

Fig. 2. Algorithm to generate the ranking of attributes.

```
Function Evaluate_Ranking
   Inputs:
      La: List of features(ranking to be evaluated);
      U: Evaluation Criteria;
      Nae: nº Features to evaluate
   Output:
      Le: List of evaluations
 begin
   ListAux := {}
   Le := {}
   for i = 1 until Nae
      at := feature i of L
      ListAux := ListAux + {att}
      C := Classify(ListAux, U)
      Le(i) := C
   end for
end Evaluate_Ranking
```

Fig. 3. Algorithm to evaluate the ranking.

($gene_i$,$condition_j$), $gene_i$ expression level is stored under $condition_j$.

TABLE I
DATA SETS.

			Characteristics		
	Ref	**Acronym**	nºEx	nºAtt	classes
Dataset_C	[14]	dsc	60	7129	2
Colon	[15]	col	62	2000	2
Leukemia	[16]	leuk	38	7129	2
Lymphoma	[17]	lym	96	4026	9

After generating the ranking for a dataset, we need to evaluate how good is that ranking. This evaluation is based on the concept of area under the learning curve (AUC). In general terms, each point of the curve (x, y) is calculated by looking at the number of features that take part in the classification of the original data set, where 'x' means the number of features and 'y' means the classification rate. The number of features selected for the calculation of each point is gradually increased by one until the given number of features.

The algorithm to evaluate the ranking is depicted in Figure 3. It is composed by just a function called Evaluate_Ranking. This function has three input parameters: feature ranking to be evaluated (*La*), classification method that will be used (U), and the whole number of features (*Na*). The output parameter is a list (*Le*) which contains at each position the value of the learning curve, i.e., the percentage of correct classification (using the criterion U). The number of features begins with value one and ends with a given number (*Nae*). Features are selected according to the order indicated in the ranking (*La*).

When those values are calculated, the curve is represented, so that, X–axis represents the number of features and Y–axis represents the evaluation of the classification obtained for that number of features. As the learning curve is used to calculate the area under it, this will be normalized, i.e., the values in the X–axis are normalized using linear normalization in [0,1]. In this way, a ranking is better than another one when its AUC is greater than that of the other one.

The AUCs, result of evaluating the first one hundred features of rankings generated using our approach and exhaustive methods (with $K = 1$ and $K = 2$) over data set mentioned in Table I, are shown in Table II. The configuration in the generation process and in the ranking evaluation has been 3–NN ([18]) as the method of classification and 5–fold cross–validation, as the validation criteria. The value of *Na* has been limited to 0.4% for the random method, what provides a range between 8 and 28, depending on the dataset. The exhaustive method is run three times, all of them using the mentioned configuration, but with a different number of experiments: equivalent to the number of features in data (M) for the first execution ($1 \times M$); double for the second one ($2 \times M$); and triple for the third one ($3 \times M$).

In Table II, as well as showing the results of evaluating the ranking generated by different methods, it is also presented the number of steps needed for each one of them. In addition, the best AUC obtained for each data set is highlighted in bold type. For example, for data set dsc (nervous system tumor), the exhaustive method $K = 1$ has needed 7129 steps to obtain a 67.5% of classification success; whereas for the next value ($K = 2$), it has needed 25407756 steps to achieve a classification rate of 83.39. If we pay attention to the results obtained by the proposed method, we can observe that using the same number of steps than $K = 1$ we get a value (81.83) much better than to the one obtained by it (67.5) and very close to the value returned by $K = 2$ (83.39). It has been necessary 14258 steps to get a value greater than or equal to the last one, providing an excellent result 83.73%. In short, only 0.056% of

TABLE II

EXPERIMENTAL RESULTS.

	Exhaustive Search				Random Search					
	K=1		*K=2*		*1*M*		*2*M*		*3*M*	
BD	nºStep	AUC	nºStep	AUC	nºStep	AUC	nºStep	AUC	nºStep	AUC
dsc	7129	67.5	25407756	83.39	7129	81.83	14258	83.73	21387	**84.1**
col	2000	82.77	1999000	**84.23**	2000	83.05	4000	83.87	6000	84.06
leuk	7129	95.02	25407756	95.53	7129	96.22	14258	96.48	21387	**96.69**
lym	4026	78.84	8102325	78.82	4026	84.98	8052	86.43	12078	**87.47**

experiments has been necessary. The best result for the data set dsc has been 84.1 which was produced by the method based on random search ($3 \times M$), using 21387 steps. Comparing the results obtained by the proposed method to the exhaustive method, it is obvious that the more steps are run, the better results are achieved. The $3 \times M$ method has obtained the best result for each data set, except for *col*. The best result obtained with the last data set has been provided by $K = 2$. To explain the reason of this result, we must study the dependencies that our method has, with its only two input parameters (number of steps to do and maximum number of features at each step). Anyway, note that K=2 has needed 1999000 steps to produce 84.23, whereas $3 \times M$ has obtained a very similar value (84.06) with only 0.3% of the steps.

With the intention of studying the dependencies of our method with its input parameters, different experiments on data sets enumerated in Table I have been carried out. The configuration for the evaluation and for the ranking generation has been the same that the one used in previous experiments, although in this case, the maximum number of features in each case has been ranged between 1 and 5, increasing by 0.1. The number of steps has increased up to 6085200 ((1+2+3)* numFeatures for each dataset $\times 50$ "different Na values") as a result of having applied the study three times per each data set, changing the number of experiments done for $1 \times M$, $2 \times M$ and $3 \times M$. Since we have used 5–fold cross–validation, the total number of models has been 3042000 (5 folds $\times$ number of experiments).

In short, 600 different rankings, evaluated by the algorithm depicted in Figure 3, have been obtained. The evaluation results are represented in Figure 4 which is divided into four graphs, each of them belongs to a different data set. These results are shown by ranking evaluation tendencies obtained for the Na variations, so that the maximum number of features used by the method is indicated in the X–axis and the AUC obtained after evaluating the ranking is plotted in the Y–axis. The study of these tendencies has been divided according to the numbers of experiments done by the proposed method, so that for each one of the graphs, there are three curves related to the method tendencies, with $1 \times M$, $2 \times M$ and $3 \times M$ experiments, respectively.

Besides those curves, in order to compare our method with the exhaustive one, in each graph AUC, the tendencies are also represented as a result of evaluating the rankings obtained by the exhaustive method with $K = 1$ and $K = 2$. Note that the rankings generated by the exhaustive method do not depend on Na and, therefore, their tendencies will be represented by a 0 slope straight lines, whose values are shown in Table II, in columns "K=1" and K=2" for each data set.

We must indicate that the tendency with $K = 1$ for DataSet–C has not been represented for the sake of clarity. From those graphs we can see that depending on the chosen data set, the tendencies are increasing or decreasing. The correlation degree among the dataset features will partially determine the slope, which will be descending or ascending. From this reasoning the value of Na should be inversely proportional to the correlation degree. Observing the other input value of our method (Ne), we can say that the bigger it is, the better result we will get. This is so as the values of our method tendencies increase as more steps are taken into account.

The efficiency of both the exhaustive method and our approach is also compared. For Leukemia and Lymphoma, our approach improves the exhaustive method, whereas for the rest, the result depends on the number of features chosen (Na). However for DataSet–C, we can observe that in case of choosing a number of experiments equal to $3 \times M$, and setting Na lower than 1.3%, our approach will perform better. On the contrary, for Colon dataset, we should use a value for Na greater than 2.8%, independently of the number of experiments. In case of choosing $Ne = 3 * M$, then Na could take values over 0.9%.

Figure 5 is presented to demonstrate that from the tendencies of the different methods, the efficiency can be compared. The AUCs are shown as a result of applying the evaluation algorithm (depicted in Figure 3) over the rankings generated by the different methods (random with $1 \times M$, $2 \times M$ and $3 \times M$ and exhaustive one with $K = 1$ and $K = 2$) to DataSet-C. The number of selected features (according to the order indicated in the ranking to be evaluated) is found in the X–axis and classification rate is depicted in the Y–axis.

In Figure 5 are shown two graphs, whose configuration about those methods only differs in the value of Na, the rest of that configuration is the same as the one used in previous experiments. The selected values for Na have been those that make the learning curve of method $3 \times M$ inferior and superior to $K = 2$, in particular, 0.4 and 4.3, respectively. Therefore, it is possible to appreciate that the classification of the first 100 features are evaluated in comparison with the classification obtained by $K = 2$, and they agree with the tendencies because they are superior in case of $Na = 0.4$ (from 30 attributes) and lower in the other case (the curve for $3 \times M$ is completely under the curve $K = 2$).

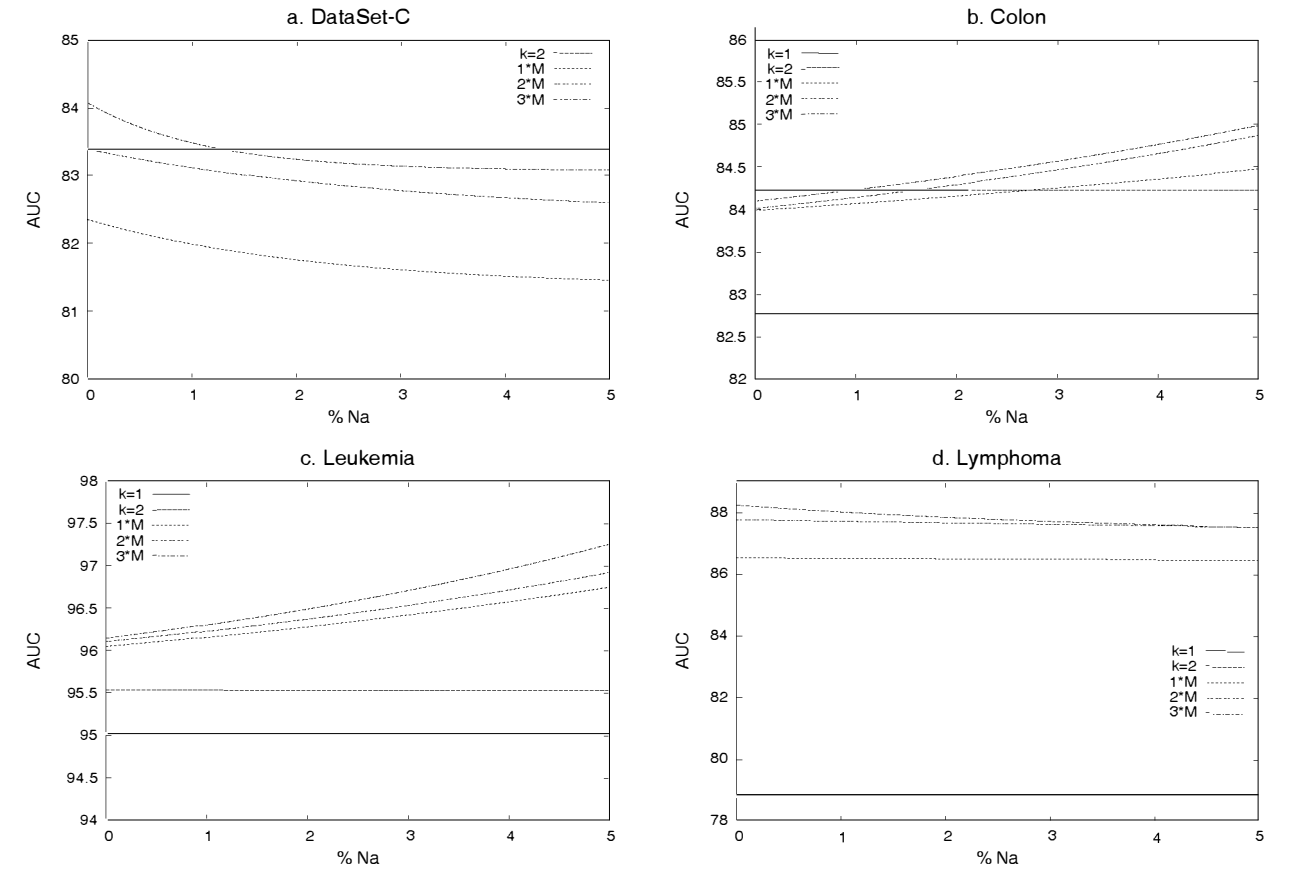

Fig. 4. Tendencies.

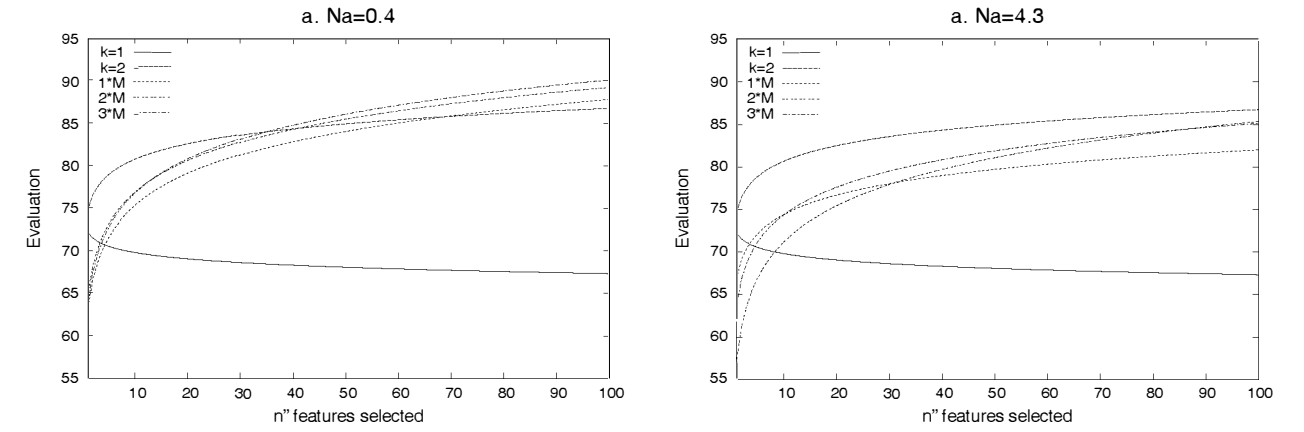

Fig. 5. DataSet-C. Delimiting curves.

In summary, attending to comparisons done between bootstrapping based method and exhaustive one, we can state that the proposed method generates features rankings whose evaluation is similar (and even improves) to those generated by the exhaustive method, however using much less number of steps.

IV. Conclusions

In this paper we introduce a new feature selection algorithm to reduce the computational cost of an exhaustive feature selection method without compromising its efficiency. The search is based on the bootstrapping technique, which carries out a random selection with replacement.

After analizing our approach using four very high dimensional datasets, we conclude that our approach produces similar results to the exhaustive one with very low number

of steps. This reduction ranges around the ratio 1 : 100000, that is, per each 100000 iterations of the exhaustive method, our method needs only one to achieve the same performance, which is an important improvement on the computational cost.

REFERENCES

[1] M. Dash and H. Liu *Feature Selection for Classification.* Intelligent Data Analysis, Elsevier, Vol. 1, no. 3, pp 131 - 156, 1997

[2] H. Liu and Lei Yu *Toward Integrating Feature Selection Algorithms for Classification and Clustering.* IEEE Trans. on Knowledge and Data Engineering, vol. 17, no. 4, pp 491-502, 2005

[3] R. Kohavi and G.H. John *Wrappers for Feature Subset Selection.* Artificial Intelligence, vol. 97, no.1-2, pp. 273-324, 1997

[4] Jennifer G. Dy et al. *Feature Selection for Unsupervised Learning.* J. Mach. Learn. Res., vol. 5, pp. 845-889, 2004

[5] S. Das *Filters, Wrappers and a Boosting-Based Hybrid for Feature Selection.* Proc. 18th Int´l Conf. Machine Learning, pp. 74-81, 2001

[6] H. Liu et al. *Feature Selection with Selective Sampling.* Proc. 19th Int´l Conf. Machine Learning, pp. 395-402,2002

[7] C. Cardie *Using Decision Trees to Improve Case-Based Learning.* Proc. 10th Intl´l Conf. Machine Learning, P. Utgoff, ed., pp. 25-32, 1993.

[8] A.N. Mucciardi and E.E. Gose *A comparison of Seven Techinques for Choosing Subsets of Pattern Recognition.* IEEE Trans. Computer, vol. 20, pp. 1023-1031,1971.

[9] R. Ruiz, J. Riquelme, J. Aguilar-Ruiz *Projection-based measure for efficient feature selection.* Journal of Intelligent and Fuzzy System, vol. 12,pp 175-183, 2003

[10] J. Doak *An Evaluation of Feature Selection Methods and Their Apllication to Computer Security.* Technical report, Univ. of California at Davis, Dept. Computer Science,1992.

[11] H. Liu, H. Motoda *Feature Selection for Knowledge Discovery and Data Mining.* Boston: Kluwer Academic, 1998.

[12] G. Brassard et al. *Fundamentals of Algorithms.* New Jersey: Prentice Hall,1996.

[13] Efron, B. *Estimating the error rate of a prediction rule: improvements on cross-validation* J.Amer.Stat.Ass. vol 78, pp. 316-331, 1983

[14] Scott L. Pomeroy, et al *Prediction of Central Nervous System Embryonal Tumour Outcome based on Gene Expression.* NATURE, vol 415, pp. 436-442, 2002.

[15] U. Alon et al. *Broad patterns of gene expression revealed by clustering of tumor and normal colon tissues probed by oligonucleotide arrays.* PNAS, Vol 96, Issue 12, pp. 6745-6750, 1999.

[16] T.R. Golub, et al. *Molecular Classification of Cancer: Class Discovery and Class Prediction by Gene Expression Monitoring.* SCIENCE, vol 286, pp. 531-537, 1999.

[17] Ash A. Alizadeh, et al *Distinct types of diffuse large B-cell lymphoma identified by gene expression profiling.* NATURE, VOL 403, N° 3, pp. 503-511, February 2000.

[18] Dasarathy, Belur V. *Nearest neighbor(NN) norms: NN pattern classification techniques.* IEEE Comp. Soc. Press. 1990.