Feature Selection Based on Run Covering

Su Yang[1], Jianning Liang[1], Yuanyuan Wang[2], and Adam Winstanley[3]

[1] Shanghai Key Laboratory of Intelligent Information Processing, Dept. of Computer Science and Engineering, Fudan University, Shanghai 200433, China
[2] Dept. of Electronic Engineering, Fudan University, Shanghai 200433, China
[3] National Centre for Geocomputation, Dept. of Computer Science, National University of Ireland, Maynooth, Co. Kildare, Ireland
suyang@fudan.edu.cn

Abstract. This paper proposes a new feature selection algorithm. First, the data at every attribute are sorted. The continuously distributed data with the same class labels are grouped into runs. The runs whose length is greater than a given threshold are selected as "valid" runs, which enclose the instances separable from the other classes. Second, we count how many runs cover every instance and check how the covering number changes once eliminate a feature. Then, we delete the feature that has the least impact on the covering cases for all instances. We compare our method with ReliefF and a method based on mutual information. Evaluation was performed on 3 image databases. Experimental results show that the proposed method outperformed the other two.

1 Introduction

For pattern recognition problems, the data represented in feature space can be of very high dimensionality. However, some features are redundant and do not provide extra information over the others. In some worse cases, feature extraction could introduce noise, which does not contribute to pattern classification but degrade the classification performance. Thus, how to find a compact and effective feature subset is a significant issue, to which a great deal of effort has been devoted so far. There are two types of methodologies for dimensionality reduction: The unsupervised methods like PCA and the supervised methods, for which the class labels of the training samples are prior known. In this study, we foucse on the supervised dimensionality recduction, which is referred to as feature selection. Feature selection plays an important role in a variety of applications, including image classification [9,10]. Some reviews on feature selection can be found in [1-3]. According to [4], feature selectors can be sorted into two different groups: wrappers and filters. Wrappers employ a given classifier to evaluate features such that the feature selection is optimized for the given classifier. Filters evaluate features according to some measurements of class separability. In general, filters are less computationally complex than wrappers. As for filters, some methods measure the power of every independent feature in terms of class separability while some other methods measure the power of a subset of features as a whole. According to [3], only exhaustive search and the branch and bound methods

L.-W. Chang, W.-N. Lie, and R. Chiang (Eds.): PSIVT 2006, LNCS 4319, pp. 208 – 217, 2006.

[12,13] are optimal feature selectors. However, the branch and bound methods are based on an assumption that a performance index drops monotonously. In fact, investigations on developing new feature selectors have never stopped. Recently, mutual information based methods have received much attention [7,14-15].

In this study, we propose a new feature selection method, which belongs to the filter category. Its implementation is outlined as follows. First, the data at every attribute are sorted. The continuously distributed data with the same class labels are grouped into runs. The runs whose length is greater than a given threshold are selected as "valid" runs, which imply that the instances falling into such runs are separable from the other classes because enough instances from an identical class occupy spatially close positions. Second, we count how many runs cover every instance and check how the covering number changes once eliminate a feature. We delete the feature that has the least impact on the covering cases for all instances.

We compare our method with ReliefF [5], which is member of the Relief family [6], and the method based on mutual information [7]. Both methods belong to the filter category. We evaluate the 3 methods on 3 image databases provided in UCI Machine Learning Repository [16]. Experimental results show that the proposed method outperformed the other two.

2 The Method

The feature selection method is based on run covering. First, we sort the data values at every attribute. After the sorting, the data at every attribute can be divided into some segments, where the class labels of the elements in every segment should be identical. Such a segment is referred to as a run. If an instance is covered by at least one run (One of its attribute is included in the run.) whose length is greater than a given threshold, it means that this instance is separable from the other classes. By eliminating recursively such attributes that the removal of them will not affect the class separability in terms of run covering, a feature subset can then be selected. In the following, we first give the definition of runs. Then, we describe the feature selection algorithm. Finally, we provide a feature ranking method by which we can identify the least important feature and delete it in every loop.

2.1 Runs

The runs at every attribute can be extracted via the following procedure:

(1) Suppose that there are N instances. After sorting the kth attribute, we obtain $x_{k1} \le x_{k2} \le \ldots \le x_{kN}$. Denote the corresponding class labels as $[C(x_{k1}), C(x_{k2}), \ldots, C(x_{kN})]$. Note that $C(x_{ki}) \in \{1,2,\ldots,L\}$, $i=1,2,\ldots,N$, if there are L classes. Also, the indices of the corresponding instances are denoted as $[I(x_{k1}), I(x_{k2}), \ldots, I(x_{kN})]$.

(2) If $x_{ki}=x_{k,i+1}=\ldots=x_{k,i+U}$ but $C(x_{ki})=C(x_{k,i+1})=\ldots=C(x_{k,i+U})$ does not hold at the same time, it means that $x_{ki}, x_{k,i+1}, \ldots, x_{k,i+U}$ are not separable. To denote that, we let $C(x_{ki})=C(x_{k,i+1})=\ldots=C(x_{k,i+U})=0$. Note that only $0 \notin \{1,2,\ldots,L\}$. Thus, it is not a valid class label.

(3) If $C(x_{ki})=C(x_{k,i+1})=\ldots=C(x_{k,i+U})\neq 0$, then, $[x_{ki},x_{k,i+1},\ldots,x_{k,i+U}]$ forms a run. The length of this run is $U+1$.

(4) Repeat (3) until all runs at every attribute have been found.

Some examples regarding the previously defined runs are shown in Fig. 1, 2, and 3, where the class labels distributed along a given attribute are illustrated. We can see that Fig. 1, 2, and 3 contains 2, 3, and 12 runs, respectively. Clearly, the case shown in Fig. 1 promises the best separability between the 2 classes while Fig. 3 corresponds with the worst case. The two cases shown in Fig. 1 and 2 are better in that the run length is greater. A longer run corresponds with a better case in terms of class separability. These examples show that the runs defined as above characterize the class separability to some extent. If the maximum run length at an attribute is too short as the case shown in Fig. 3, it means that the instances are not separable at this attribute. If we set a threshold of 5 and look for such runs whose length is greater this threshold, we can find out 2, 1, and 0 runs in Fig. 1, 2, and 3, respectively.

However, run length is a coarse characterization of class separability. It is known that N individually strong attributes are not certainly the best N attributes if combined together (N attributes performing well alone could perform unsatisfactorily as a team.). In this study, our focus is how to choose the best team, not the best N individuals. This can be achieved by using the run covering described in the next section.

111111112222222222 2222211111111122222 1122112211221122112212

Fig. 1. Class labels at a given attribute

Fig. 2. Class labels at a given attribute

Fig. 3. Class labels at a given attribute

2.2 Eliminate Redundant Attributes Based on Run Covering

Prior to describing the feature selection algorithm, we give some definitions as follows.

(1) $R=\{R_i\}$: The run set including all the runs at every attribute.

(2) $\|R_i\|$: The length of the run $R_i \in R$.

(3) A: The attribute set that contains all remainder attributes following the feature elimination process described below. Initially, this set contains all the attributes. After the feature elimination process stops, the residual attributes are the finally selected features.

(4) /* Comments on pseudo codes */.

Following is the feature selection (feature elimination) algorithm:

(1) Assign a score to each attribute to represent the individual power of every attribute in terms of its contribution to class separability. Let us denote these scores as $w(1)$, $w(2)$, …., and $w(K)$. If $w(i)<w(j)$, it means that the ith attribute is better than the jth attribute in terms of class separability. This is also referred to as feature ranking. The detailed ranking algorithm is provided in section 2.3.

(2) Compute $C_l = \sum_{k,i} h_l(x_{kj}, R_i, T)$, where

$$h_l(x_{kj}, R_i, T) = \begin{cases} 1 & I(x_{kj}) = l \wedge x_{kj} \in R_i \wedge |R_i \rhd T \\ 0 & else \end{cases}.$$

/* If x_{kj} is a member of run R_i and the corresponding run length is greater than a threshold T, then, the corresponding instance $I(x_{kj})=l$ has been covered once. C_l corresponds with how many times the lth instance has been covered*/

(3) $\forall p \in A$, compute $C_{l,-p} = \sum_{k,i,k\neq p} h_l(x_{kj}, R_i, T)$.

/* The times that instance l has been covered without the kth attribute */

(4) Find $P = \{p : p \in A \wedge \sum_l | g(C_l) - g(C_{l,-p}) | = 0\}$, where

$$g(x) = \begin{cases} 1 & x > 0 \\ 0 & x \leq 0 \end{cases}.$$

/* P corresponds with the redundant attributes, the elimination of any of which will not cause a critical change on the times that each instance has been covered, where the critical change means that the covering times for any instance go down from a positive value to 0 suddenly after eliminating an attribute. */

(5) Find $q = \arg\max\{w(p) \mid p \in P\}$ and eliminate q from A.

/* Delete the least important feature in set P, where the criterion to select the least important feature refers to the feature ranking algorithm described in the next section */.

(6) If $P=\phi$, delete $q = \arg\max\{w(p) \mid p \in A\}$

/* If no feature satisfying that elimination of it will not change the covering case for every instance, then, delete the least important feature ranked by the feature ranking algorithm described in the next section. */

(7) Let $C_l=C_{l,-p}$ and Go to (3) until the number of the residual attributes in A is equal to the predefined desired number.

Some discussions about the above algorithm are given below. The central idea of this algorithm is: Look for such attributes that the class separability will not be affected if eliminating them. The run covering plays an important role in this algorithm. First, we select the runs whose length is greater than a given threshold T. Every selected run covers the instances that are separable at a given attribute since the instances from the same class distribute very closely to each other (They are within a run). As every instance has K attributes, it has the chance to be covered by K runs at most. If an instance is covered $V \leq K$ times by the runs, then, eliminating one attribute from the V attributes will not affect the classification of this instance because it is still covered by the runs at the other V-1 attributes, which means that this instance is still close to the instances from the same class at the V-1 attributes. Taking all the instances into account together, we hold the following idea. Suppose that $Q \leq N$ instances are covered by at least one run. When we eliminate one attribute, if the Q covered instances are still covered by at least one run, then, it means that this attribute

is redundant and contributes no additional information in contrast to the reminder attributes. Eliminating it should have no impact on the classification. In case there exist $R>1$ attributes that the removal of any of them will not chance the covering, we eliminate only one attribute among the R attributes and then recompute the covering. In such a case, the selection of the attribute to be eliminated is not random. It is based on a feature-ranking criterion. That is, we firstly score every attribute according to its individual significance in terms of class separability. Then, we always eliminate the least important one from the R attributes. The feature-ranking criterion is described in detail in the next section. The above procedure can be repeated to eliminate redundant attributes recursively.

In the above algorithm, T is the only parameter (See step 2), which determines how many runs are valid in counting the covering number. We let the threshold $T=0.1\times N$, where N denotes the number of all instances. We have tested a couple of different values for T and found that $T=0.1\times N$ is a satisfactory one in this study, which not only leads to a satisfactory overall classification performance but also promises a stable classification performance when $T\in[0.1\times N-\Delta,0.1\times N+\Delta]$, where Δ is a relative small positive value. Note that T can be scaled to adapt to problmes from different domains.

The above algorithm can be easily extended to multi-class classification. We only need to decompose the multi-classification into multiple two-class classifications (pairwise classification). Then, we look for such attributes the elimination of which do not affect the covering for every two-class classification. For example, if there are L classes, then, we decompose the L-class separability computation into $L(L-1)/2$ parallel two-class separability computations. Here, step (1)~(3) and step (7) are implemented as $L(L-1)/2$ parallel processes. In step (4), the intersection of the $L(L-1)/2$ solutions forms P. The other steps are the same as described prevoiusly.

Fig. 4. Distribution of two classes along a given attribute

Fig. 5. Distribution of two classes along a given attribute

2.3 Feature Ranking

Suppose that there are M and N samples in class X and Y and the kth attribute of the two classes are $\{x_{k1},x_{k2},\ldots,x_{kM}\}$ and $\{y_{k1},y_{k2},\ldots,y_{kN}\}$, respectively.

We define the relationship between x_{ki} and y_{kj} as

$$H(x_{ki},y_{kj})=\begin{cases}1 & x_{ki}<y_{kj}\\ 0 & x_{ki}\geq y_{kj}\end{cases}. \tag{1}$$

The above definition means that if x_{ki} lies in the left side of y_{kj}, then, $H(x_{ki},y_{kj})=1$. Else, $H(x_{ki},y_{kj})=0$.

Based on the relationship between two instances, we define the overall relationship between the two classes in terms of the kth attribute as

$$d_k=\max\{\sum_{i=1}^{M}\sum_{j=1}^{N}H(x_{ki},y_{kj}),\sum_{i=1}^{M}\sum_{j=1}^{N}H(y_{kj},x_{ki})\}. \tag{2}$$

It summarizes the relationship between every class X sample and every class Y sample. Also, it reveals the separability between the two classes and can be understood as a distance measure between the two classes. This is explained via the following two examples.

See the example shown in Fig. 4, where the samples in the overlapping region are underlined. Suppose that, in the from left to right order, the "×" marks represent one-dimensional class X samples $x_1,x_2,\ldots,x_8$ and the "+" marks represent one-dimensional class Y samples $y_1,y_2,\ldots,y_7$, respectively. The underlined "×" corresponds with x_8 and the underlined "+" corresponds with y_1. With regard to x_1, all the 7 samples of the other class lie in the right side of it. So, we obtain $\sum_j H(x_1,y_j)=7$. With regard to x_8, only 6 samples of the other class lie in the right side of it. Thus, we hold $\sum_j H(x_8,y_j)=6$. In fact, $\sum_j H(x_i,y_j)$ figures out how many samples in class Y locate in the right side of x_i. In contrast, $\sum_j H(y_j,x_i)$ reveals how many samples in class Y locate in the left side of x_i. Therefore, $\sum_i\sum_j H(x_i,y_j)$ is a measure of the degree that class X locates in the left side of class Y and $\sum_i\sum_j H(y_j,x_i)$ characterizes the degree that class X locates in the right side of class Y. Obviously, $\max\{\sum_i\sum_j H(x_i,y_j), \sum_i\sum_j H(y_j,x_i)\}$ reveals the relative relationship between the two classes of interest. For the above example, $\sum_i\sum_j H(x_i,y_j)=55$ and $\sum_i\sum_j H(y_j,x_i)=1$. This means that most samples of class X locate in the left side of class Y. In accordance with Eq. (1), the separability measure between the two classes is 55. Now, consider another example shown in Fig. 5, where the overlapping region is larger than the case shown in Fig. 4. Correspondingly, the separability measure between the two class computed via Eq. (1) is 52. Taking into account the two examples, it is easy to see that a smaller separability measure corresponds with a more severe overlap between the two classes of interest, namely, a worse case in terms of separability. On the contrary, a greater separability measure, which corresponds with a smaller overlapping degree, means a better case in terms of separability.

Suppose that there are L classes and class j contains $N(j)$ samples, $j=1,2,\ldots,L$. Let $x_{ki}^{(j)}$ denote the kth attribute of the ith sample of class j. We further assume that every sample has K attributes. The feature-ranking algorithm is described below. Suppose that the input is $\{ x_{ki}^{(j)} \mid j=1,2,\ldots,L;\ i=1,2,\ldots,N(j);\ k=1,2,\ldots,K\}$. With regard to the kth attribute, compute the separability between every pair of classes via Eq. (1) and Eq. (2), that is, $\{ d_k^{(u,v)} \mid u=1,\ldots,L\text{-}1;\ v=u+1,\ldots,L\}$. Then, let $\sum_{u,v} d_k^{(u,v)}$ be the overall discrimination power of the kth attribute, according to which all attributes can be ranked.

2.4 Computational Complexity

Suppose every class contains N samples. Let L denote the class number, K the feature number, and M the dimension of set A. The complexity of step 1, step 2, and the loop from step 3 to step 7 is roughly $O(K\times L\times(L\text{-}1)\times N^2)$, $O(L\times(L+1)\times K\times N)$, and $O(M\times(M+1)\times L\times(L+1)\times N)$, respectively. The overall complexity is basically the sum of the three parts.

3 Experimental Results

We tested the proposed algorithm with UCI machine learning databases [16]. The performance evaluation was conducted with the letter recognition database, the satellite image classification database, and the image segmentation database. The data properties of the 3 databases are summarized in Table 1. We also compare our method with 2 other methods: ReliefF [5] and the method based on mutual information [7]. In classifying every data set, we use 3 classifiers: 1-nearest neighbor (1-NN), decision tree, and support vector machine (SVM). Here, we use the weka software to implement Relief and the decision tree as well as the SVM classifier [17]. We apply 10-fold cross validation for performance evaluation [8].

The classification accuracy against the feature number for the image segmentation data is illustrated in Fig 6, 7, and 8, where 1-NN, decision-tree, and SVM classifiers are applied, respectively. Obviously, the proposed method outperforms the other two methods. For the 1-NN classification based on the proposed feature selector, when the feature number is equal to 3, the classification accuracy reaches 97.23%. Then, the classification accuracy changes very little, between 96.49% and 97.58%. The classification accuracy using the full attributes is 96.62%, which is less than that using only 3 features selected by the proposed algorithm. See Fig. 6, the other two methods perform much worse than the proposed method. See Fig. 7 and Fig. 8, the same case takes place when comparing the 3 methods based on decision tree and SVM classification.

The classification accuracy against the feature number for the satellite image data is shown in Fig 9, 10, and 11, where 1-NN, decision-tree, and SVM classifiers are applied, respectively. It can be seen that the proposed method outperforms the other two methods given any feature number.

The classification accuracy against the feature number for the letter recognition data is exhibited in Fig 12, 13, and 14, where 1-NN, decision-tree, and SVM classifiers are applied, respectively. The proposed method promises comparable performance to ReliefF while both methods outperform the method based on mutual information.

In the above 3 benchmarks, we can see that different classifier leads to different classification performance but the comparison among different feature selection methods never changes with the choice of classifiers. According to Fig. 9~11, the proposed method approaches the best performance or a satisfactory perofrmance very quickly but the other two methods do not. The above comparisons show that the proposed method performs well in selecting useful features for image classification.

Table 1. Data properties

Data	#Attributes	#Instances	#Classes
Image	19	2310	7
SatImage	36	6435	6
Letter	16	20000	26

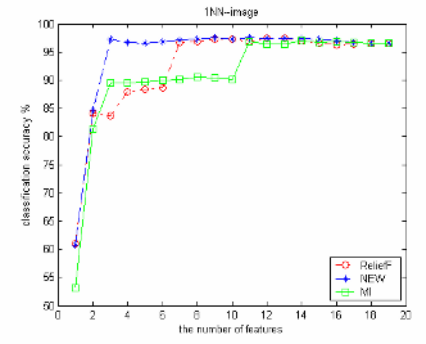

Fig. 6. Classification accuracy against feature number using 1-NN: image segmentation

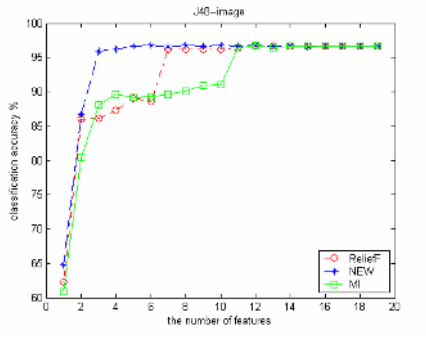

Fig. 7. Classification accuracy against feature number using decision tree: Image segmentation

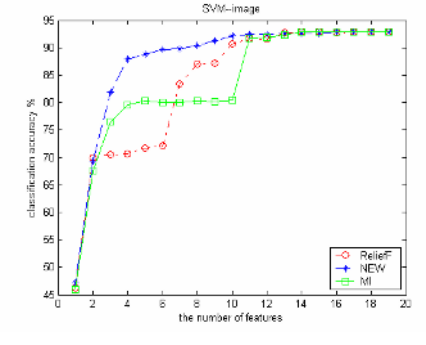

Fig. 8. Classification accuracy against feature number using SVM: Image segmentation

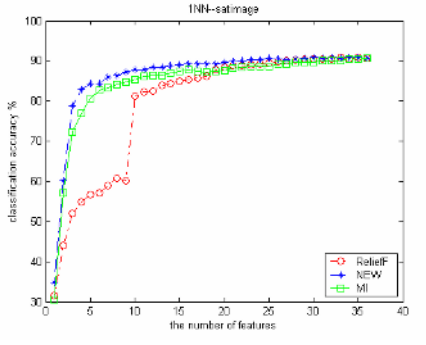

Fig. 9. Classification accuracy against feature number using 1-NN: Satellite image

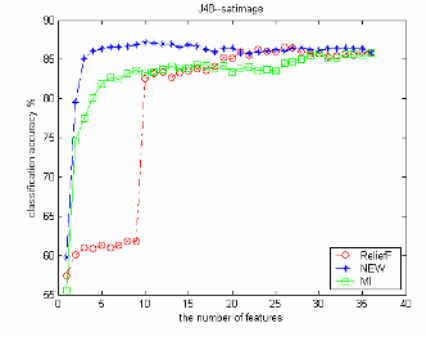

Fig. 10. Classification accuracy against feature number using decision tree: Satellite image

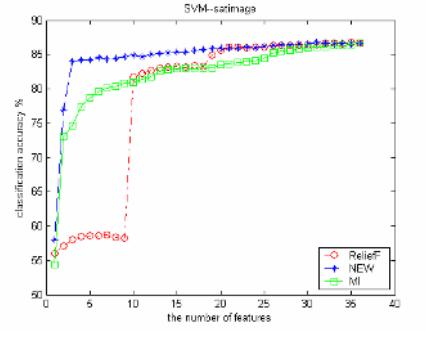

Fig. 11. Classification accuracy against feature number using SVM: Satellite image

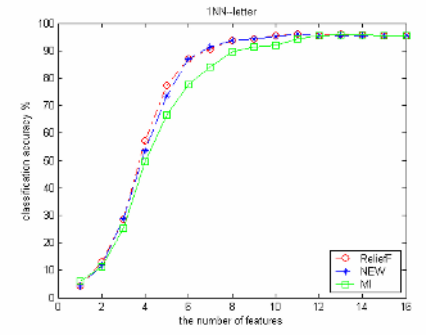

Fig. 12. Classification accuracy against feature number using 1-NN: Letter

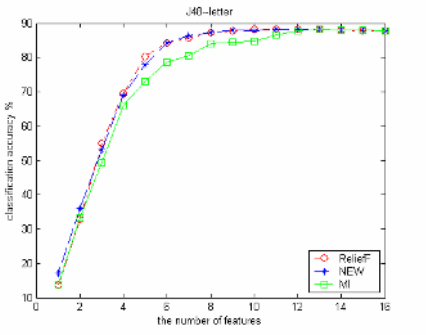

Fig. 13. Classification accuracy against feature number using decision tree: Letter

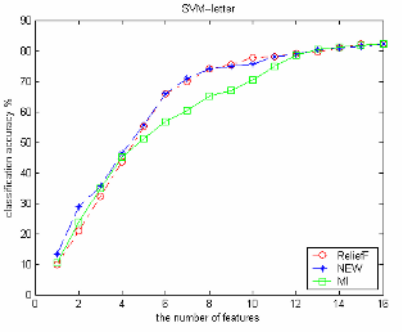

Fig. 14. Classification accuracy against feature number using SVM: Letter

4 Concluding Remarks

In this study, we propose a new feature selection method. It is based on run covering. The heart of this algorithm is to check whether the removal of a given attribute will change the covering of every instance. If not, it can be decided that this attribute is redundant. The run length plays an important role in judging whether an instance is separable from the other classes at a given attribute. The experiments confirmed the effectiveness of this method in terms of selecting useful features for image classification. Note that the run-length based method works with not only the linear separable attributes but also the attributes that are not linearly separable.

Another important issue is the stopping criterion, that is, what feature number is satisfactory to stop the feature elimination procedure. For the limited space of this paper, we did not present the criterion and the related performance evaluation. One stopping criterion can be: If the covering case for any instance changes after eliminating a feature, then, stop the feature selection. It is easy to implement. We just need to modify step (6) of the algorithm to be: If $P=\phi$, the desired feature number has been approached.

Acknowledgement. This work is supported in part by Natural Science Foundation of China under grant 60305002, China/Ireland Science and Technology Research Collaboration Fund under grant CI-2004-09, and National Basic Research Program of China under grant 2006CB705700.

References

[1] Guyon, I. and Elisseeff, A.: An introduction to variable and feature selection. Journal of Machine Learning Research 3 (2003) 1157-1182
[2] Liu, H. and Yu, L.: Toward integrating feature selection algorithms for classification and clustering. IEEE Trans. Knowledge and Data Engineering, 17 (2005) 491-502
[3] Jain, A. and Zongker, D.: Feature selection: Evaluation, application, and small sample performance. IEEE Transactions on Pattern Analysis and Machine Intelligence, 19 (1997) 153-158
[4] Kohavi, R. and John, G. H.: Wrappers for feature subset selection. Artificial Intelligence, 97 (1997) 273-324
[5] Robnik-Sikonja, M. and Kononenko, I.: Theoretical and empirical analysis of ReliefF and RreliefF. Machine Learning, 53 (2003) 23-69
[6] Kira, K. and Rendell, L.: A practical approach to feature selection. Proc. Int. Conf. Machine Learning, (1992) 249-256
[7] Peng, H., Long, F., Ding, C.: Feature selection based on mutual information: Criteria of max-dependency, max-relevance, and min-redundancy. IEEE Transactions on Pattern Analysis and Machine Intelligence, 27 (2005) 1226-1238
[8] Kohavi, R.: A study of cross-validation and bootstrap for accuracy estimation and model selection. Proc. Int. Joint Conf. Artificial Intelligence, (1995) 1137-1145
[9] Y. Rui, T. S. Huang, and S. Chang, "Image retrieval: Current techniques, promising directions and open issues," Visual communication and image representation, vol. 10, no. 4, pp. 39-62, 1999.
[10] Ho, T. K and Baird, H. S.: Pattern classification with compact distribution maps. Computer Vision and Image Understanding, 70 (1998) 101-110
[11] Cover, T. M.: The best two independent measurements are not the two best. IEEE Transactions on Systems, Man, and Cybernetics, 4 (1974) 116-117
[12] Narendra, P. M. and Fukunaga, K.: A branch and bound algorithm for feature subset selection. IEEE Transactions on Computers, 26 (1977) 917-922
[13] Somol, P., Pudil, P., Kittler, J.: Fast branch & bound algorithms for optimal feature selection. IEEE Transactions on Pattern Analysis and Machine Intelligence, 26 (2004), 900-912
[14] N. Kwak and C. H. Choi, "Input feature selection by mutual information based on Parzen windows," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 24, no. 12, pp. 1667-1671, 2002.
[15] Trappenberg, T., Ouyang, J., Back, A.: Input variable selection: Mutual information and linear mixing measures. IEEE Trans. Knowledge and Data Engineering, 18 (2006) 37-46
[16] http://www.ics.uci.edu/~mlearn/MLRepository.html
[17] http://www.cs.waikato.ac.nz/~ml