Information Sciences 181 (2011) 115–128

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Simultaneous feature selection and classification using kernel-penalized support vector machines

Sebastián Maldonado, Richard Weber *, Jayanta Basak [1]

Department of Industrial Engineering, University of Chile, República 701, Santiago de Chile, Chile
IBM India Research Lab, New Delhi, India

ARTICLE INFO

Article history:
Received 17 November 2009
Received in revised form 14 July 2010
Accepted 31 August 2010

Keywords:
Feature selection
Embedded methods
Support vector machines
Mathematical programming

ABSTRACT

We introduce an embedded method that simultaneously selects relevant features during classifier construction by penalizing each feature's use in the dual formulation of support vector machines (SVM). This approach called kernel-penalized SVM (KP-SVM) optimizes the shape of an anisotropic RBF Kernel eliminating features that have low relevance for the classifier. Additionally, KP-SVM employs an explicit stopping condition, avoiding the elimination of features that would negatively affect the classifier's performance. We performed experiments on four real-world benchmark problems comparing our approach with well-known feature selection techniques. KP-SVM outperformed the alternative approaches and determined consistently fewer relevant features.

1. Introduction

Classification is one of the most important data mining tasks. The performance of the respective models depends on – among other elements – an appropriate selection of the most relevant features which is a combinatorial problem in the number of original features and offers the following advantages [1]:

- A low-dimensional representation reduces the risk of *overfitting* [5,10].
- Using fewer features decreases the model's complexity which improves its generalization ability.
- A low-dimensional representation requires less computational effort.

Among existing classification methods, support vector machines (SVMs) provides several advantages such as adequate generalization to new objects, absence of local minima, and representation that depends on only a few parameters [21]. However, this method in standard formulation does not determine the importance of the features used [10] and is therefore not suitable for feature selection.

This fact has motivated the development of several approaches for feature selection using SVMs (see e.g. [7]). Those methods generally work as filters selecting features from a high-dimensional feature space prior to designing the subsequent classifier. They provide feature ranking but without considering the combination of variables that optimizes classification performance. In this paper a novel embedded method for feature selection using SVM for classification problems is introduced. This method, called kernel-penalized SVM (KP-SVM), determines simultaneously a classifier with high classification

* Corresponding author at: Department of Industrial Engineering, University of Chile, República 701, Santiago de Chile, Chile. Tel.: +56 2 9784072; fax: +56 2 678 7895.
E-mail addresses: semaldon@ing.uchile.cl (S. Maldonado), rweber@dii.uchile.cl (R. Weber), basakjayanta@yahoo.com (J. Basak).
[1] The author is presently affiliated with NetApp Bangalore India, Advanced Technology Group.

0020-0255/$ - see front matter © 2010 Elsevier Inc. All rights reserved.
doi:10.1016/j.ins.2010.08.047

accuracy and an adequate feature subset by penalizing each feature's use in the dual formulation of the respective mathematical model. In numerical experiments using four well-known data sets, KP-SVM outperforms existing approaches.

This paper is structured as follows. Section 2 introduces SVM for classification. Recent developments for feature selection using SVMs are reviewed in Section 3. KP-SVM, the proposed embedded method for feature selection based on SVM is presented in Section 4. Section 5 provides experimental results using four real-world data sets. Several important aspects that arise from this work are discussed in Section 6. A summary of this paper can be found in Section 7 where we provide its main conclusions and address future developments.

2. Classification with SVM

Vapnik [21] developed SVMs for binary classification. This section introduces the respective approach using the following terminology. Given training vectors $\mathbf{x}_i \in \Re^n$, $i = 1,\ldots,m$ and a vector of labels $\mathbf{y} \in \Re^m$, $y_i \in \{-1,+1\}$, SVM provides the optimal hyperplane $f(\mathbf{x}) = \mathbf{w}^T \cdot \mathbf{x} + b$ that aims to separate the training patterns. In the case of linearly separable classes this hyperplane maximizes the sum of the distances to the closest positive and negative training patterns. This sum is called *margin*. To construct the maximum margin or optimal separating hyperplane, we need to classify correctly the vectors $\mathbf{x}_i$ of the training set into two different classes y_i, using the smallest norm of coefficients $\mathbf{w}$.

For a non-linear classifier, SVM maps the data points into a higher dimensional space $\mathcal{H}$, where a separating hyperplane with maximal margin is constructed. The following quadratic optimization problem has to be solved

$$\underset{\mathbf{w},b,\xi}{\text{Min}}\ \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{m}\xi_i, \tag{1}$$

subject to

$$y_i \cdot \left(\mathbf{w}^T \cdot \phi(\mathbf{x}_i) + b\right) \geqslant 1 - \xi_i, \quad i = 1,\ldots,m,$$
$$\xi_i \geqslant 0, \quad i = 1,\ldots,m,$$

where training data are mapped to the higher dimensional space $\mathcal{H}$ by the function $\mathbf{x} \to \phi(\mathbf{x}) \in \mathcal{H}$. A set of slack variables ξ is introduced for each training vector and C is a penalty parameter on the training error [21].

Under this mapping the solution of an SVM has the form:

$$f(\mathbf{x}) = sign\left(\sum_{i=1}^{m} y_i \alpha_i^* \phi(\mathbf{x}) \cdot \phi(\mathbf{x}_i) + b^*\right). \tag{2}$$

As can be seen it is sufficient to compute the scalar products of the form $\phi(\mathbf{x}) \cdot \phi(\mathbf{y})$ [18]. A kernel function $K(\mathbf{x},\mathbf{y}) = \phi(\mathbf{x}) \cdot \phi(\mathbf{y})$ which defines an inner product in $\mathcal{H}$ performs the respective mapping leading to the following decision function $f(\mathbf{x})$:

$$f(\mathbf{x}) = sign\left(\sum_{i=1}^{m} y_i \alpha_i^* K(\mathbf{x},\mathbf{x}_i) + b^*\right). \tag{3}$$

The optimal hyperplane is the one with maximal distance (in $\mathcal{H}$) to the closest image $\phi(\mathbf{x}_i)$ from the training data. The dual formulation can be stated as follows:

$$\underset{\boldsymbol{\alpha}}{\text{Max}}\ \sum_{i=1}^{m}\alpha_i - \frac{1}{2}\sum_{i,s=1}^{m}\alpha_i\alpha_s y_i y_s K(\mathbf{x}_i,\mathbf{x}_s), \tag{4}$$

subject to

$$\sum_{i=1}^{m}\alpha_i y_i = 0,$$
$$0 \leqslant \alpha_i \leqslant C, \quad i = 1,\ldots,m.$$

In most applications the polynomial or the radial basis function (Gaussian kernel) are chosen [18]:

1. Polynomial function: $K(\mathbf{x}_i,\mathbf{x}_s) = (\mathbf{x}_i \cdot \mathbf{x}_{s+1})^d$, where $d \in \mathbb{N}$ is the degree of the polynomial.
2. Radial basis function: $K(\mathbf{x}_i,\mathbf{x}_s) = \exp\left(-\frac{\|\mathbf{x}_i-\mathbf{x}_s\|^2}{2\sigma^2}\right)$, where $\sigma > 0$ is the parameter controlling the width of the kernel.

Empirically, the Gaussian kernel provided best classification performance in our previous studies [10] and will be used in the subsequent experiments. In Section 5 we also study the classification performance using different kernel functions on four benchmark data sets, in order to confirm our hypothesis.

3. Feature selection for SVMs

According to [5,7], there are three main directions for feature selection: filter, wrapper, and embedded methods. In this section we provide a brief overview of each one of these approaches, and present the methods that have been compared with

the proposed technique in the present paper. The first direction (*filter methods*) uses statistical properties of the features in order to filter out poorly informative ones. This is usually done before applying any classification algorithm.

The Fisher Criterion Score computes the importance of each feature independently of others by comparing the correlation of each variable with the output labels. The score $F(j)$ of feature j is given by:

$$F(j) = \left| \frac{\mu_j^+ - \mu_j^-}{\left(\sigma_j^+\right)^2 + \left(\sigma_j^-\right)^2} \right|, \tag{5}$$

where μ_j^+ $\left(\mu_j^-\right)$ represents the mean for the jth feature in the positive (negative) class and σ_j^+ $\left(\sigma_j^-\right)$ is the respective standard deviation.

A *wrapper method* explores the whole set of variables to score feature subsets according to their predictive power, optimizing a performance criterion of the subsequent algorithm that uses the respective subset for classification. These algorithms are computationally demanding, but often provide more accurate results than filter methods since they are performed in combination with the subsequent classification technique [5,10].

One of the most popular wrapper methods for SVMs was proposed by Guyon et al. [6] and is known as Recursive Feature Elimination (SVM-RFE). In this work we consider a version of SVMs that includes kernel functions as described in [7,14] and in [17] for multi-class classification [24]. The goal of this approach is to find a subset of size r among n variables ($r < n$) which maximizes the performance of the classifier. The method, given that one wishes to employ only $r < n$ input variables in the final decision rule, attempts to find the best subset of r features, i.e. the r features which lead to the largest margin of class separation. This problem is based on a sequential backward selection, removing one feature at a time until r features remain. The feature to be removed at each iteration is the one whose removal minimizes the variation of $W^2(\boldsymbol{\alpha})$:

$$W^2(\boldsymbol{\alpha}) = \sum_{i,s=1}^{m} \alpha_i \alpha_s y_i y_s K(\mathbf{x}_i, \mathbf{x}_s). \tag{6}$$

The vector $W^2(\boldsymbol{\alpha})$ is a measure of the model's predictive ability and is inversely proportional to the margin. The elimination of features is done applying the following procedure:

1. Given a solution $\boldsymbol{\alpha}$, for each feature p calculate:

$$W^2_{(-p)}(\boldsymbol{\alpha}) = \sum_{i,s=1}^{m} \alpha_i \alpha_s y_i y_s K\left(\mathbf{x}_i^{(-p)}, \mathbf{x}_s^{(-p)}\right), \tag{7}$$

where $\mathbf{x}_i^{(-p)}$ represents the training object i with feature p removed.
2. Eliminate the feature with smallest value of $\left|W^2(\boldsymbol{\alpha}) - W^2_{(-p)}(\boldsymbol{\alpha})\right|$.

The last approach (*embedded methods*) performs feature selection in the process of model construction. For example, the methods presented in [12,13] add an extra term that penalizes the cardinality of the selected feature subset to the standard cost function of SVM. By optimizing this modified cost function features are selected simultaneously to model construction.

Another embedded approach is the Feature Selection ConcaVe (FSV) [1], based on the minimization of the "zero norm": $\|\mathbf{w}\|_0 = |\{i : w_i \neq 0\}|$. Note that $\|\cdot\|_0$ is not a norm because the triangle inequality does not hold [1], unlike l_p-norms with $p > 0$. Since l_0-"norm" is non-smooth, it was approximated by a concave function:

$$\|\mathbf{w}\|_0 \approx \mathbf{e}^T(\mathbf{e} - \exp(-\beta|\mathbf{w}|)) \tag{8}$$

with an approximation parameter $\beta \in \Re_+$ and $\mathbf{e} = (1, \ldots, 1)^T$. The formulation for FSV follows:

$$\underset{\mathbf{w},\mathbf{v},b,\xi}{\text{Min}} \sum_{j=1}^{n} [1 - exp(-\beta v_j)] + C \sum_{i=1}^{m} \xi_i, \tag{9}$$

subject to

$$\begin{aligned} & y_i \cdot (\mathbf{w}^T \cdot \mathbf{x}_i + b) \geqslant 1 - \xi_i, \quad i = 1, \ldots, m, \\ & -v_j \leqslant w_j \leqslant v_j, \quad j = 1, \ldots, n, \\ & \xi_i \geqslant 0, \quad i = 1, \ldots, m. \end{aligned}$$

The problem (9) can be solved using an iterative method called Successive Linearization Algorithm (SLA) for FSV [2].

4. The proposed method for feature selection

An embedded method for feature selection using SVMs is proposed in this section. The reasoning behind this approach is that we can improve classification performance by eliminating the features that affect on the generalization of the classifier by optimizing the kernel function. The main idea is to penalize the use of features in the dual formulation of SVMs using a

gradient descent approximation for kernel optimization and feature elimination. The proposed method attempts to find the best suitable RBF-type kernel function for each problem with a minimal dimension by combining the parameters of generalization (using the *2*-norm), goodness of fit and feature selection (using a *0*-"norm" approximation).

4.1. Notation and preliminaries

For this approach we use the anisotropic Gaussian kernel:

$$K(\mathbf{x}_i, \mathbf{x}_s) = \exp\left(-\sum_{j=1}^{n} \frac{(x_{ij} - x_{sj})^2}{2\sigma_j^2}\right), \tag{10}$$

in which the kernel shape is given by $\boldsymbol{\sigma} = [\sigma_1, \sigma_2, \ldots, \sigma_n]$, n being the number of variables. Considering different widths in different dimensions, the importance of feature j is determined by (σ_j). For example, if σ_j is very large, the particular variable j loses its importance since its contribution to the kernel function's exponent will be close to zero. On the other hand, if σ_j is very small then the contribution of the variable j to the exponent will be large thus increasing its importance.

We propose the following change of variables for (10), in order to convert the feature selection process into a minimization problem: $\boldsymbol{v} = \left[\frac{1}{\sigma_1}, \frac{1}{\sigma_2}, \ldots, \frac{1}{\sigma_n}\right]$, which leads to:

$$K(\mathbf{x}_i, \mathbf{x}_s, \boldsymbol{v}) = \exp\left(-\frac{||\boldsymbol{v} * \mathbf{x}_i - \boldsymbol{v} * \mathbf{x}_s||^2}{2}\right), \tag{11}$$

where $*$ denotes the componentwise vector product operator, which is defined as $\mathbf{a}*\mathbf{b} = (\mathbf{a}_1\mathbf{b}_1, \ldots, \mathbf{a}_n\mathbf{b}_n)$.

4.2. KP-SVM algorithm

The proposed approach (kernel-penalized SVM) incorporates feature selection in the dual formulation of SVMs. The formulation includes a penalization function $f(\boldsymbol{v})$ based on the 0-"norm" approximation (8) described in Section 3 and modifying the Gaussian kernel using an (anisotropic) width vector $\boldsymbol{v}$ as a decision variable. The feature penalization should be negative since the dual SVM is a maximization problem. The following embedded formulation of SVMs for feature selection is initially proposed:

$$\underset{\boldsymbol{\alpha}, \boldsymbol{v}}{\text{Max}} \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,s=1}^{m} \alpha_i \alpha_s y_i y_s K(\mathbf{x}_i, \mathbf{x}_s, \boldsymbol{v}) - C_2 f(\boldsymbol{v}), \tag{12}$$

subject to

$$\sum_{i=1}^{m} \alpha_i y_i = 0,$$
$$0 \leqslant \alpha_i \leqslant C, \quad i = 1, \ldots, m,$$
$$v_j \geqslant 0, \quad j = 1, \ldots, n.$$

Notice that the values of $\boldsymbol{v}$ are always considered to be positive, in contrast to the weight vector $\mathbf{w}$ in formulation (9), since it is desirable that the kernel widths be positive values. Considering the 0-"norm"approximation described in (8), $\|\boldsymbol{v}\|_0 \approx \mathbf{e}^T(\mathbf{e} - \exp(-\beta|\boldsymbol{v}|))$, and since $|v_j| = v_j \; \forall j$, it is not necessary to use the 1-norm in the approximation.

Along the lines of formula (8) the following feature penalization function is proposed, where the approximation parameter β is also considered. In [2], the authors suggest setting β to 5. We also try different values for this parameter to study the influence of β in the final solution (see Section 6)

$$f(\boldsymbol{v}) = \mathbf{e}^T(\mathbf{e} - \exp(-\beta\boldsymbol{v})) = \sum_{j=1}^{n} \left[1 - exp(-\beta v_j)\right]. \tag{13}$$

Since the formulation (12) is non-convex, we develop an iterative algorithm as an approximation for this formulation. We propose a 2-step methodology: first the traditional dual formulation of SVM for a fixed (isotropic) kernel width $\boldsymbol{v}$ is solved:

$$\underset{\boldsymbol{\alpha}}{\text{Max}} \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,s=1}^{m} \alpha_i \alpha_s y_i y_s K(\mathbf{x}_i, \mathbf{x}_s, \boldsymbol{v}), \tag{14}$$

subject to

$$\sum_{i=1}^{m} \alpha_i y_i = 0,$$
$$0 \leqslant \alpha_i \leqslant C, \quad i = 1, \ldots, m.$$

In the second step the algorithm solves, for a given solution $\boldsymbol{\alpha}$, the following non-linear formulation:

$$\underset{\boldsymbol{v}}{\text{Min}}\, F(\boldsymbol{v}) = \sum_{i,s=1}^{m} \alpha_i \alpha_s y_i y_s K(\mathbf{x}_i, \mathbf{x}_s, \boldsymbol{v}) + C_2 f(\boldsymbol{v}), \tag{15}$$

subject to

$$v_j \geqslant 0, \quad j = 1, \ldots, n.$$

The goal of formulation (14) is to find a sparse solution, making zero as many components of v as possible. We propose an iterative algorithm that updates the anisotropic kernel variable $\boldsymbol{v}$, using the gradient of the objective function, and eliminates the features that are close to zero (below a given threshold ϵ). The algorithm kernel width updating and feature elimination follows:

Algorithm 1. Kernel width updating and feature elimination

1. Start with $\boldsymbol{v} = v_0\mathbf{e}$;
2. cont = true; $t = 0$;
3. **while**(cont==true) **do**
4. train SVM (step 1) for a given $\boldsymbol{v}$;
5. $\boldsymbol{v}^{t+1} = \boldsymbol{v}^t - \gamma \Delta F(\boldsymbol{v}^t)$;
6. **for all** $\left(v_j^{t+1} < \epsilon\right)$ **do**
7. $v_j^{t+1} = 0$;
8. **end for**
9. **if** ($\boldsymbol{v}^{t+1} == \boldsymbol{v}^t$) **then**
10. cont = false;
11. **end if**
12. $t = t + 1$;
13. **end while**;

In the fourth line the algorithm adjusts the kernel variables by using the gradient descent procedure, incorporating a parameter γ, which has to be sufficiently small to avoid negative widths, especially at the first iterations. In this step the algorithm computes the gradient of the objective function in formulation (15) for a given solution of SVMs $\boldsymbol{\alpha}$, obtained by training an SVM classifier using formulation (14). For a given feature j, the gradient of formulation (15) is:

$$\Delta_j F(\boldsymbol{v}) = \sum_{i,s=1}^{m} v_j (x_{i,j} - x_{s,j})^2 \alpha_i \alpha_s y_i y_s K(\mathbf{x}_i, \mathbf{x}_s, \boldsymbol{v}) + C_2 \beta \exp(-\beta v_j). \tag{16}$$

The lines 6, 7, and 8 of the algorithm represent the feature elimination step. When a kernel variable v_j in iteration $t + 1$ is below a threshold ϵ, we consider this feature irrelevant because of the argument given in sub Section 4.1 and we eliminate this feature by setting $v_j = 0$. This variable will not be included in further iterations of the algorithm. The threshold ϵ has to be sufficiently small to avoid the elimination of relevant variables in the first iterations of the algorithm.

The lines 9, 10, and 11 of the algorithm represent the stopping criterion, which is reached when $\boldsymbol{v}^{t+1} \approx \boldsymbol{v}^t$. It is also possible to monitor the convergence by considering the measure $\|\boldsymbol{v}^{t+1} - \boldsymbol{v}^t\|_1$, which represents the variation of the kernel width between two consecutive iterations t and $t + 1$, as will be shown in Section 6.

Notice that the 1-norm penalty (LASSO penalty) can also be used instead of the 0-norm approximation. According to [2,13], the 1-norm by itself can lead to good feature selection and classification results, without considering the 2-norm for robustness. In order to improve feature selection, both papers consider the 0-norm penalization when the main objective is to find sparse solutions. Following this argument, we suggest using the proposed methodology with the zero norm approximation.

4.3. Feature ranking using KP-SVM

The approach KP-SVM presented in Sub Section 4.2 attempts to find an optimal subset of features for classification. Other feature selection methods for SVMs such as [5,7,14] find a subset of size r among the n features which maximizes the classifier's performance and use different validation methods in order to answer the question of how many ranked features must be provided to the classifier.

If the goal of feature selection is to find a subset of r features, KP-SVM can be modified in order to accomplish this goal as well. The main idea is to keep iterating until fewer than r features are selected. Considering k the number of features selected in iteration $t + 1$, $k \leqslant r$, we sort the eliminated features in the $(t + 1)$th iteration according to their respective value of v in the tth iteration. We replace the $r - k$ most relevant removed features (greater v) and include them in the solution of the $(t + 1)$th solution. The modified algorithm follows:

Algorithm 2. KP-SVM algorithm for feature ranking

1. Start with $\boldsymbol{v} = v_0\mathbf{e}$;
2. cont = true; $t = 0$;
3. **while**(cont==true) **do**
4. train SVM (step 1) for a given $\boldsymbol{v}$;
5. $\boldsymbol{v}^{t+1} = \boldsymbol{v}^t - \gamma\Delta F(\boldsymbol{v}^t)$;
6. **for all** $\left(v_j^{t+1} < \epsilon\right)$ **do**
7. $v_j^{t+1} = 0$;
8. **end for**
9. **if** $\left(|\{j : v_j^{t+1} \neq 0\}| \leqslant r\right)$ **then**
10. $\left(|\{j : v_j^{t+1} \neq 0\}| = k\right)$;
11. **Sort**($\boldsymbol{v}^t$, desc);
12. **if** $\left(v_j^t > 0 \text{ and } v_j^{t+1} = 0,\ \forall j \leqslant r - k\right)$ **then**
13. $v_j^{t+1} = v_j^t$
14. **end if**
15. cont = false;
16. **end if**
17. $t = t + 1$;
18: **end while**;

Notice that this approach is only valid when r is greater than the number of selected features given by the method's stopping criterion.

In this variation of the algorithm we modify its stopping criterion (line 9). Instead of reaching convergence, the algorithm stops when the number of available features, k, at this iteration is smaller or equal to the desired number of attributes, r. Then we recover the $r - k$ most relevant features removed in the past iteration and update $\boldsymbol{v}$.

4.4. Relation to other feature selection methods for SVMs

Different approaches for SVM-based feature selection are already available. SVM-RFE and other wrapper methods presented in [14] differ regarding the feature selection methodology and the stopping criterion. The proposed method directly obtains a variable subset that simultaneously attempts to improve classification performance with minimal dimension, and does not rank variables based on different criteria, such as a weight vector $\mathbf{w}$ [6] or a gradient-based measure [14,23]. Additionally, KP-SVM presents an explicit stopping criterion, unlike other wrapper methods cited.

In contrast to the proposed approach, many embedded methods penalize the weight vector of SVMs and therefore are limited to linear [2,12,13] or polynomial kernels [23]. The method proposed by Weston et al. [22] differs from ours in the objective function since it minimizes the R^2W^2 bound on the leave-one-out error *LOO* of a trained hard-margin SVM classifier instead of the dual formulation of SVM.

The embedded formulation proposed in [4] performs feature selection via adaptive scaling, which is similar to considering different kernel widths. However, this method differs from KP-SVM in the formulation of the optimization problem: instead of penalizing the features, the cited method restricts the number of features in order to perform feature selection. The authors also proposed an optimization scheme that differs from our gradient-based algorithm. Another approach, proposed in [3] performs feature selection by removing small scaling factors in a principal components space, where each principal component is scaled by a scaling factor. The differences from our approach are mainly the same: the feature selection process via kernel penalization and the algorithm for kernel updating.

5. Experimental results

We applied the proposed approach for feature selection on four well-known benchmark data sets: Two real-world data sets from the UCI repository [8], and two DNA microarray data sets. These data sets have already been used to compare feature selection algorithms (see, for example, [15,22]).

For model selection we follow the procedure presented in [16]: training and test subsets are obtained from the original data set by dividing it randomly, preserving the proportions of the different classes. For model and feature selection purposes, the training set is then further divided using 10-fold cross-validation in order to estimate prediction accuracy. After model selection, a model comparison is performed on the test subset following the procedure proposed in [14,15]: we split the test subset into two subsets of approximately 60% of the observations for training the final models and the remaining 40%

for the testing, while ensuring that the proportions of positive and negative classes are similar in both sets. A mean test error is finally obtained by averaging the results over 100 different splits of the test subset.

5.1. Description of data sets

In this subsection we briefly describe the different data sets mentioned above.

5.1.1. Diabetes data set (DIA)

The Pima Indians Diabetes (DIA) data set presents 8 features and 768 instances (500 tested negative for diabetes and 268 tested positive).

5.1.2. Wisconsin Breast Cancer (WBC)

This data set contains 569 observations (212 malignant and 357 benign tumors) described by 30 continuous features that are computed from a digitized image of a Fine Needle Aspirate (FNA) of a breast mass. They describe characteristics of the cell nuclei present in the image. As a preprocessing step the features were scaled between 0 and 1.

5.1.3. Colorectal Microarray data set (CMA)

This data set contains the expression of the 2000 genes with highest minimal intensity across 62 tissues (40 tumor and 22 normal). The genes are placed in order of descending minimal intensity. In contrast to our proposed method, which automatically determines a subset of features for classification, other filter and wrapper methods generate a feature ranking. In these cases we use the number of ranked variables as mentioned in [14]: 20, 50, 100, 250, 500, 1000 and 2000 (i.e. no variables removed).

5.1.4. Lymphoma Microarray data set (LMA)

The lymphoma problem contains the gene expression of 96 samples (61 malignant and 35 normal) described by 4026 features. We compare our results using the number of variables as mentioned in [14]: 20, 50, 100, 250, 1000, 2000 and 4026 (i.e. no variables removed).

Table 1 summarizes the relevant information for each benchmark data set:

5.2. Results using kernel-penalized feature selection

The first step of the experimentation is model selection. We compare the results of the best model found using a standard model selection procedure for three different kernel functions (linear, polynomial and Gaussian) without feature selection. The best combination of parameters will be used as input for KP-SVM (initial kernel parameter v_0 and C).

Table 2 presents the mean and standard deviation of the test error for the training subset using 10-fold cross-validation. The following set of values for the parameters (penalty parameter C, degree of the polynomial function d, and Gaussian kernel width σ) were used:

$$C \in \{0.1, 0.5, 1, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 200, 300, 400, 500, 1000\},$$
$$d \in \{2, 3, 4, 5, 6, 7, 8, 9\},$$
$$\sigma \in \{0.1, 0.5, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 20, 100\}.$$

Table 1
Number of variables, number of examples, and proportion of examples in the predominant class for all four data sets.

	Variables	Examples	Predominant class proportion
DIA	8	768	0.65
WBC	30	569	0.63
CMA	2000	62	0.65
LMA	4026	96	0.64

Table 2
Number of original features N, mean and standard deviation of effectiveness in percentage terms on four data sets using three different SVM with different kernel functions.

	N	SVM linear	SVM poly	SVM RBF
DIA	8	76.95 ± 1.4	77.08 ± 1.7	77.34 ± 1.5
WBC	30	94.55 ± 2.4	96.49 ± 2.2	98.25 ± 2.0
CMA	2000	80.30 ± 6.4	80.30 ± 6.4	85.70 ± 5.6
LMA	4026	94.89 ± 2.3	94.89 ± 2.3	95.89 ± 2.2

Best results were obtained for all four data sets mentioned above using the Gaussian kernel. It is also possible to modify function (11) in order to adjust the kernel function by incorporating the componentwise product to any suitable kernel if the best kernel is not the Gaussian.

In order to study the classification performance of KP-SVM we compared the results for a given number of features (determined by the stopping criterion of our approach) with different feature selection algorithms for SVMs presented before in this paper (SVM-RFE, FSV). Furthermore, we applied the filter technique Fisher Criterion Score (Fisher). The results of the mean test error over 100 realizations using the test subset are shown in Table 3, where n is the number of features determined by KP-SVM.

The proposed method outperforms all other approaches in terms of classification error for a given number of features, as can be concluded from Table 3. The gain in terms of effectiveness is significant in the microarray data sets. For these data sets other methods fail at finding a small subset of features with good classification performance.

Then we compared the classification performance of the different ranking criteria for feature selection by plotting the mean test error for an increasing number of ranked features used for learning. Figs. 1–4 show the results for each data

Table 3
Number of selected features n, mean and standard deviation of effectiveness (in percentage) using four different feature selection methods on four data sets. We outline the best model performance in bold.

	n	Fisher + SVM	FSV	RFE-SVM	KP-SVM
DIA	5	76.42 ± 1.9	76.58 ± 1.7	76.56 ± 1.9	**76.74 ± 1.9**
WBC	15	94.70 ± 1.3	95.23 ± 1.1	95.25 ± 1.0	**97.55 ± 0.9**
CMA	20	87.46 ± 7.9	92.03 ± 7.7	92.52 ± 7.2	**96.57 ± 5.6**
LMA	8	75.23 ± 11.7	63.06 ± 6.0	63.04 ± 5.9	**99.73 ± 1.0**

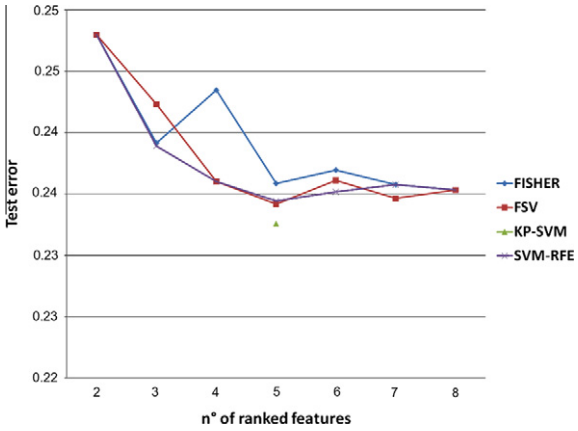

Fig. 1. Mean of test error for DIA vs. the number of ranked variables used for training.

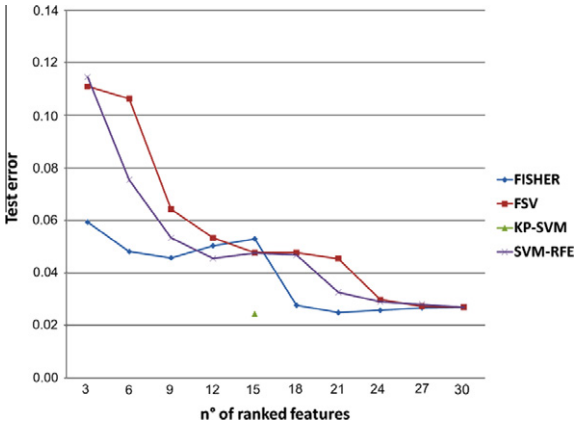

Fig. 2. Mean of test error for WBC vs. the number of ranked variables used for training.

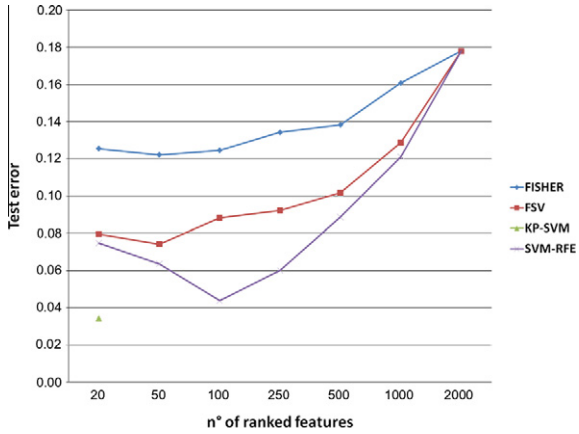

Fig. 3. Mean of test error for CMA vs. the number of ranked variables used for training.

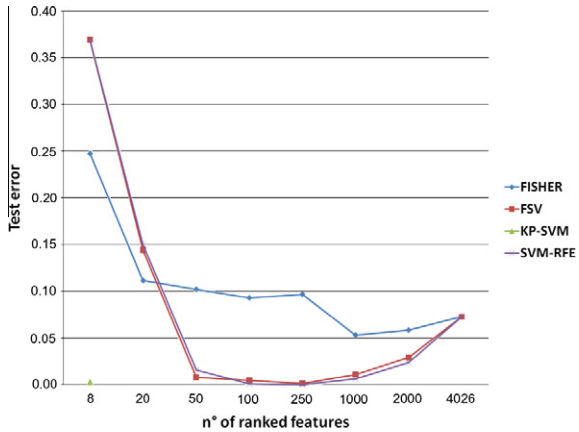

Fig. 4. Mean of test error for LMA vs. the number of ranked variables used for training.

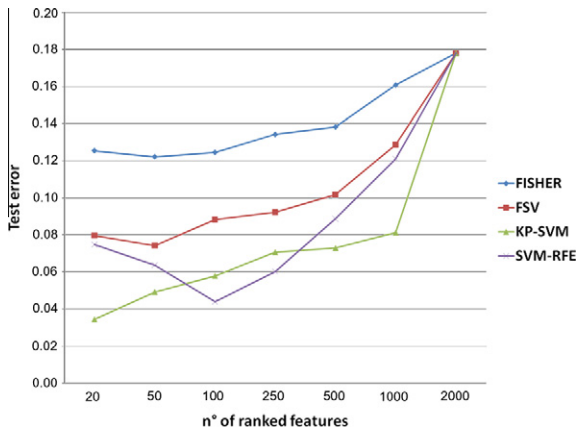

Fig. 5. Mean of test error for CMA vs. the number of ranked variables used for training.

set respectively. The proposed KP-SVM approach is represented by one single point: the mean test error obtained by its stopping criterion.

These experiments underline that the proposed approach, KP-SVM, outperforms other feature selection methods in terms of classification performance for a small number of features in all four data sets used.

5.3. Results obtained using KP-SVM as feature ranking algorithm

In order to study the performance of the modification presented in Section 4.3, we ran the algorithm for the Colorectal Microarray data set using $r \in \{20, 50, 100, 250, 500, 1000\}$ and comparing the results with other feature selection methods. Fig. 5 shows these results, illustrating that our ranking outperforms other feature selection methods for almost all the numbers of features, r. SVM-RFE performs better two out of six times but the difference is not significant. We can conclude that this modification represents a very good alternative for obtaining good classification performance for a desired number of features, even if the algorithm is designed to achieve best classification performance with few variables.

6. Discussions

The main advantage of KP-SVM in terms of computational effort is that it automatically obtains an optimal feature subset, avoiding a validation step to determine how many ranked features will be used for classification. However, several parameters should be tuned in order to obtain the final solution. In this section we study the method's performance by varying one parameter at a time, obtaining its influence on the final solution.

For the different data sets we vary the parameters C_2, β, γ, ϵ, and ν_0. In order to illustrate their influence on the classifier, the following graphs (Figs. 6–10) display the performance in terms of classification error (using 10-fold cross-validation) and number of selected features for the data set Colorectal Microarray dataset.

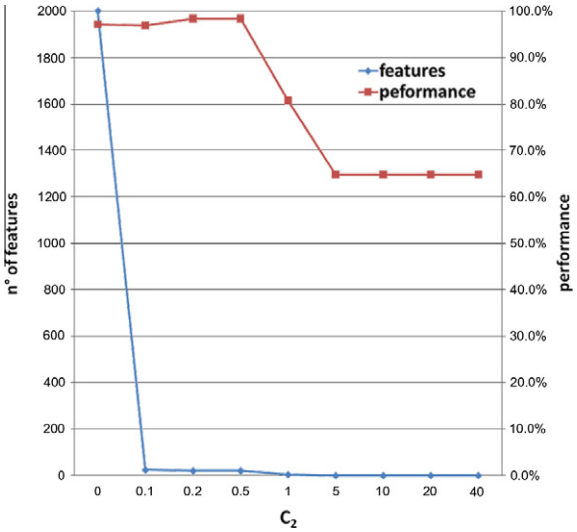

Fig. 6. Classification performance for CMA by varying parameter C_2.

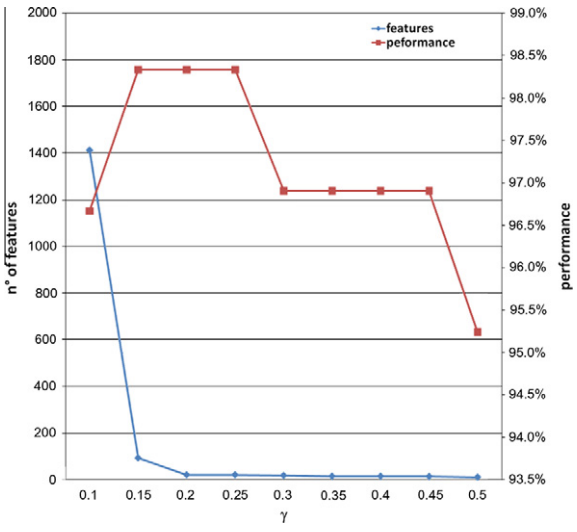

Fig. 7. Classification performance for CMA by varying parameter γ.

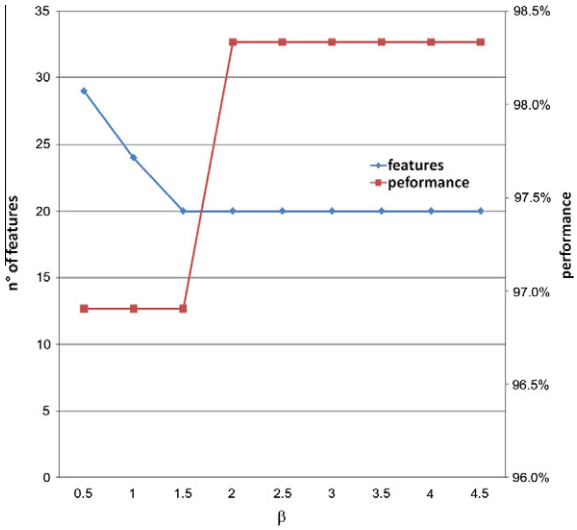

Fig. 8. Classification performance for CMA by varying parameter β.

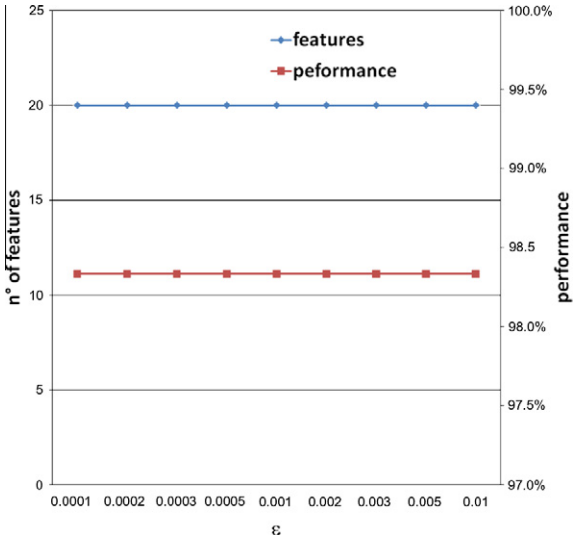

Fig. 9. Classification performance for CMA by varying parameter ϵ.

We draw the following conclusions from these experiments:

- C_2 is the most influential parameter. This parameter should be tuned carefully because of the wide range of possible values. It can be smaller or greater than C. In our experiments we found that the solution $C_2 = 0$ (no feature penalization) performs worse than a penalized solution, proving that feature selection helps to obtain better classification performance.
- Parameter γ is related to the necessary iterations for convergence: a larger γ leads to a smaller number of iterations.
- The parameters β, ϵ and v_0 have less influence in the final solution.
- In order to avoid the elimination of useful features in the first iterations of the algorithm, parameters γ and ϵ should be sufficiently small.

Another topic of discussion is the convergence of algorithms. Empirically convergence is reached in all four data sets in less than 200 iterations. Fig. 11 illustrates the variation of $\boldsymbol{v}$ for an increasing number of iterations, t, for the CMA data set ($\|\boldsymbol{v}^{t+1} - \boldsymbol{v}^{t}\|_1$).

For this data set convergence is reached in less than 75 iterations. The required number of iterations depends on the number of original features and parameters γ and C_2. Fig. 12 shows the influence of the number of iterations on the classifier for the CMA data set, in terms of selected features and classification performance.

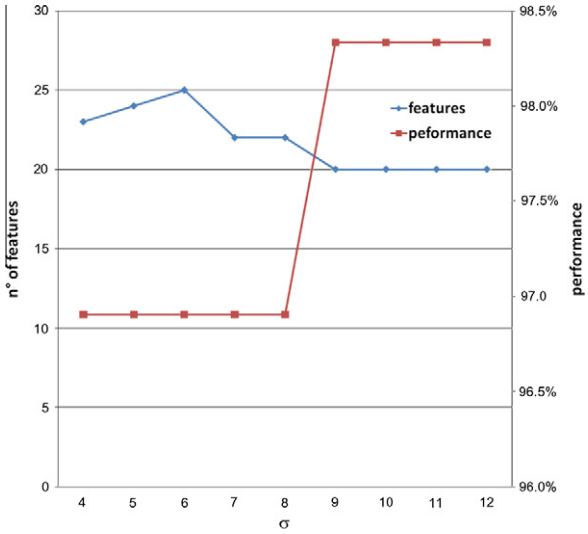

Fig. 10. Classification performance for CMA by varying parameter $\sigma = \frac{1}{\nu_0}$.

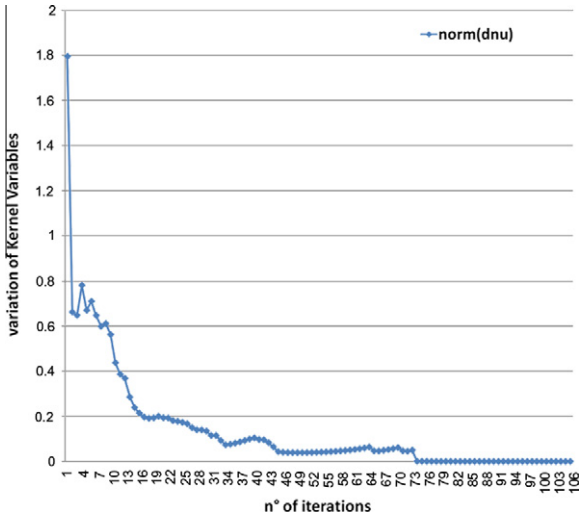

Fig. 11. $\|\boldsymbol{v}^{t+1} - \boldsymbol{v}^{t}\|_1$ for CMA by increasing the number of iterations t.

From this graph we observe that the classification performance decreases after 26 iterations, and the features selected decrease smoothly after 13 iterations.

Empirically we observe that some kernel variables $\boldsymbol{v}$ may grow unboundedly, affecting the classification performance. Therefore it is important to monitor the process and select the model by maximizing classification performance. Upper bounding the vector $\boldsymbol{v}$ is also recommended.

We propose the following suggestions for model selection:

- Perform model selection by finding the best model for the isotropic Gaussian kernel, in order to obtain the best solution $\boldsymbol{\alpha}$ and parameters C and σ.
- Set the parameters C and ν_0 according to the first model selection step.
- The value for ϵ should be sufficiently small, for example $\epsilon = \nu_0/4$. Other good initial values are $\gamma = 0.25$ and $\beta = 5$, as suggested in the literature [2].
- Vary C_2 studying the classification performance and number of selected features using cross-validation for an increasing number of iterations, t.

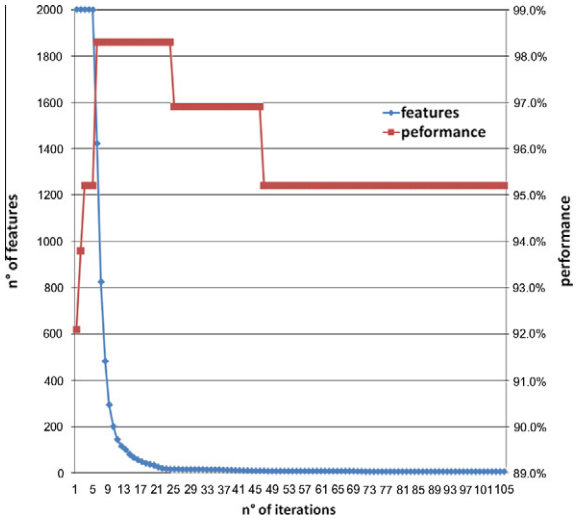

Fig. 12. classification performance for CMA by increasing the number of iterations t.

- Try different values of γ and β in order to improve the classification performance.

7. Conclusions

In this paper we present a novel embedded method for feature selection using SVMs. A comparison with other feature selection techniques shows the advantages of our approach:

- Empirically, KP-SVM outperforms other filter and wrapper techniques, based on its ability to adjust better to the data by optimizing the kernel function and simultaneously selecting an optimal feature subset for classification.
- Unlike most feature selection methods, it is not necessary to set the feature number to be selected a priori: KP-SVM determines the optimal number of features according to the regularization parameter, C_2.
- Any suitable kernel function can be used instead of the Gaussian.
- It can easily be generalized to variations of SVM, such as SV Regression and Multi-class SVM.

Even if several parameters should be tuned to obtain the final solution, the computational effort can be reduced since the feature subset is obtained automatically, reducing computational time by avoiding a further validation step in order to find the adequate number of ranked features. The proposed model selection methodology also reduces computational effort for finding the parameters.

KP-SVM attempts to find an optimal subset of features for classification. If, however, the goal of feature selection is to find a subset of a fixed size r among all n features, KP-SVM can be modified to accomplish this goal as well. The main idea is to construct a feature ranking with the removed features. Earlier removed variables have a lower rank than later removed variables. For features that have been eliminated as a batch in the same iteration, the ones with higher last value of v_j have a better rank.

Our algorithm relies on a non-linear optimization problem, which is computationally treatable but expensive if the number of input features is large. We could improve its performance by applying filter methods for feature selection before running KP-SVM [9,19] or by developing hybrid models [20]. This way we can identify and remove irrelevant features at low cost. In several Credit Scoring projects we have performed for Chilean financial institutions we used univariate analysis (Chi-Square Test for categorical features and the Kolmogorov–Smirnov Test for continuous ones) as a first filter for features selection with excellent results [10].

Future work can be done in several directions. First, it would be interesting to use the proposed method in combination with variations of SVM, such as Regression [11] or Multi-class. Also interesting would be the application of this approach with other kernel functions like polynomial kernel or with weighted support vector machines to compensate for the undesirable effects caused by unbalanced data sets in model construction; an issue which occurs for example in the domains of credit scoring and fraud detection.

Acknowledgements

Support from the Chilean Instituto Sistemas Complejos de Ingeniera (ICM: P-05-004-F, CONICYT: FBO16) is greatly acknowledged (www.sistemasdeingenieria.cl). The first author also acknowledges a grant provided by CONICYT for his Ph.D. studies in Engineering Systems at Universidad de Chile.

References

[1] A.P. Blum, P. Langley, Selection of relevant features and examples in machine learning, Artificial Intelligence 97 (1997) 245–271.
[2] P. Bradley, O. Mangasarian, Feature selection vía concave minimization and SVMs, in: Machine Learning Proceedings of the Fifteenth International Conference, Morgan Kaufmann, San Francisco, 1998, pp. 82–90.
[3] S. Canu, Y. Grandvalet, Adaptive scaling for feature selection in SVMs, Advances in Neural Information Processing Systems, vol. 15, MIT Press, Cambridge, MA, USA, 2002. pp. 553–560.
[4] O. Chapelle, V. Vapnik, O. Bousquet, S. Mukherjee, Choosing multiple parameters for support vector machines, Machine Learning 46 (1) (2002) 131–159.
[5] I. Guyon, A. Elisseeff, An introduction to variable and feature selection, Journal of Machine Learning Research 3 (2003) 1157–1182.
[6] I. Guyon, S. Gunn, M. Nikravesh, L.A. Zadeh, Feature Extraction, Foundations and Applications, Springer, Berlin, 2006.
[7] I. Guyon, J. Weston, S. Barnhill, V. Vapnik, Gene selection for cancer classification using support vector machines, Machine Learning 46 (1–3) (2002) 389–422.
[8] S. Hettich, S.D. Bay, The UCI KDD Archive, University of California, Department of Information and Computer Science, Irvine, CA, 1999. <http://kdd.ics.uci.edu>.
[9] Y. Liu, Y.F. Zheng, FS-SFS: A novel feature selection method for support vector machines, Pattern Recognition 39 (2006) 1333–1345.
[10] S. Maldonado, R. Weber, A wrapper method for feature selection using support vector machines, Information Sciences 179 (13) (2009) 2208–2217.
[11] S. Maldonado, R. Weber, Feature selection for support vector regression via kernel penalization, in: Proceedings of the 2010 International Joint Conference on Neural Networks, Barcelona, Spain, 2010, pp. 1973–1979.
[12] J. Miranda, R. Montoya, R. Weber, Linear penalization support vector machines for feature selection, in: S.K. Pal et al. (Eds.), PReMI 2005, LNCS, vol. 3776, Springer-Verlag, Berlin, Heidelberg, 2005, pp. 188–192.
[13] J. Neumann, C. Schnörr, G. Steidl, Combined SVM-based feature selection and classification, Machine Learning 61 (1–3) (2005) 129–150.
[14] A. Rakotomamonjy, Variable selection using SVM-based criteria, Journal of Machine Learning Research 3 (2003) 1357–1370.
[15] G. Rätsch, T. Onoda, K-R Müller, Soft margins for AdaBoost, Machine Learning 42 (3) (2001) 287–320.
[16] J. Reunanen, I. Guyon, A. Elisseeff, Overfitting in making comparisons between variable selection methods, Journal of Machine Learning Research 3 (2003) 1371–1382.
[17] B. Schölkopf, A.J. Smola, Learning with Kernels, MIT Press, Cambridge, MA, USA, 2002.
[18] M.-D. Shieh, C.-C. Yang, Multiclass SVM-RFE for product form feature selection, Expert Systems with Applications 35 (1–2) (2008) 531–541.
[19] Ö. Uncu, I.B. Türksen, A novel feature selection approach: combining feature wrappers and filters, Information Sciences 177 (2007) 449–466.
[20] A. Unler, A. Murat, R.B. Chinnam, *mr²PSO*: A maximum relevance minimum redundancy feature selection method based on swarm intelligence for support vector machine classification, Information Sciences, in press, doi:10.1016/j.ins.2010.05.037.
[21] V. Vapnik, Statistical Learning Theory, John Wiley and Sons, New York, 1998.
[22] J. Weston, S. Mukherjee, O. Chapelle, M. Ponntil, T. Poggio, V. Vapnik, Feature selection for SVMs, Advances in Neural Information Processing Systems, vol. 13, MIT Press, Cambridge, MA, 2001.
[23] J. Weston, A. Elisseeff, B. Schölkopf, M. Tipping, The use of zero-norm with linear models and kernel methods, Journal of Machine Learning Research 3 (2003) 1439–1461.
[24] M.L. Zhang, J.M. Pena, V. Robles, Feature selection for multi-label naive Bayes classification, Information Sciences 179 (19) (2009) 3218–3229.