Information Sciences 279 (2014) 163–175

Contents lists available at ScienceDirect

Information Sciences

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

Feature selection for Support Vector Machines via Mixed Integer Linear Programming

Sebastián Maldonado [a,*], Juan Pérez [a], Richard Weber [b], Martine Labbé [c]

[a] Universidad de los Andes, Mons. Álvaro del Portillo 12455, Las Condes, Santiago, Chile
[b] Department of Industrial Engineering, Universidad de Chile, República 701, Santiago, Chile
[c] Computer Science Department, Université Libre de Bruxelles, Boulevard du Triomphe, B-1050 Brussels, Belgium

ARTICLE INFO

Article history:
Received 4 August 2013
Received in revised form 5 February 2014
Accepted 26 March 2014
Available online 2 April 2014

Keywords:
Feature selection
Support Vector Machine
Mixed Integer Linear Programming

ABSTRACT

The performance of classification methods, such as Support Vector Machines, depends heavily on the proper choice of the feature set used to construct the classifier. Feature selection is an NP-hard problem that has been studied extensively in the literature. Most strategies propose the elimination of features independently of classifier construction by exploiting statistical properties of each of the variables, or via greedy search. All such strategies are heuristic by nature. In this work we propose two different Mixed Integer Linear Programming formulations based on extensions of Support Vector Machines to overcome these shortcomings. The proposed approaches perform variable selection simultaneously with classifier construction using optimization models. We ran experiments on real-world benchmark datasets, comparing our approaches with well-known feature selection techniques and obtained better predictions with consistently fewer relevant features.

1. Introduction

Support Vector Machines (SVMs) has been shown to be a very powerful machine learning method. For classification tasks and based on the structural risk minimization principle [18], this method attempts to find the separating hyperplane which has the largest distance from the nearest training data points of any class. SVM provides several advantages such as adequate generalization to new objects, a flexible non-linear decision boundary, absence of local minima, and a representation that depends on only a few parameters [18,21].

Feature selection is one of the most important steps within classification. An appropriate selection of the most relevant features reduces the risk of overfitting, thus improving model generalization by decreasing the model's complexity [7]. This is particularly important in small-sized high-dimensional datasets, where the *curse of dimensionality* is present and a significant gain in terms of performance can be achieved with a small subset of features [9,13]. Additionally, low-dimensional representation allows better interpretation of the resulting classifier. This is particularly important in applications in fields such as business analytics, since many machine learning approaches are considered to be *black boxes* by practitioners who therefore tend to be hesitant to use these techniques [6]. A better understanding of the process that generates the data by identifying the most relevant features is also of crucial importance in life sciences, where we want to identify, for example,

* Corresponding author. Tel.: +56 2 26181874.
E-mail address: smaldonado@uandes.cl (S. Maldonado).

http://dx.doi.org/10.1016/j.ins.2014.03.110
0020-0255/

those genes that best explain the presence of a particular type of cancer, and therefore could improve cancer incidence prediction.

Since the selection of the best feature subset is considered to be an NP-hard combinatorial problem, many heuristic approaches for feature selection have been presented to date [7]. With the two Mixed Integer Linear Programs for simultaneous feature selection and classification that we introduce in this paper we show that integer programming has become a competitive approach using state-of-the-art hardware and solvers.

In particular, we propose two novel SVM-based formulations for embedded feature selection, which simultaneously select relevant features during classifier construction by introducing indicator variables and constraining their selection via a budget constraint. The first approach studies an adaptation of the l_1-SVM formulation [4], while the second one extends the LP-SVM method presented in [22]. Our experiments show that the proposed methods are capable of selecting a few relevant features in all datasets used, leading to highly accurate classifiers within reasonable computational time.

Section 2 of this paper introduces Support Vector Machines for binary classification, including recent developments for feature selection using SVMs. The proposed feature selection approaches are presented in Section 3. Section 4 provides experimental results using real-world datasets. A summary of this paper can be found in Section 5, where we provide its main conclusions and address future developments.

2. Prior work on support vector classification

The mathematical derivation of the standard l_2-SVM formulation [18], the l_1-SVM formulation [4], and the LP-SVM method [22] are described in this section. The latter two linear classification methods constitute the basis for our proposed feature selection algorithms.

2.1. l_2-Support Vector Machine

Considering training examples $\mathbf{x}_i \in \Re^n$ with their respective labels $y_i \in \{-1, +1\}, i = 1, \ldots, m$, SVM determines a hyperplane $f(\mathbf{x}, \alpha)$ to separate the training examples optimally according to their labels, where $\alpha \in \Lambda$, is the set of possible model parameters.

This optimal split is based on Statistical Learning Theory [18], which provides a general measure of complexity (the VC dimension), and estimates a bound for the expected risk $R(\alpha)$ as a function of the empirical risk $R_{emp}(\alpha)$. According to this theory, the following inequality holds with probability $1 - \eta$ [22]:

$$R(\alpha) \leqslant R_{emp}(\alpha) + \frac{\varepsilon}{2}\left(1 + \sqrt{1 + \frac{4R_{emp}(\alpha)}{\varepsilon}}\right), \tag{1}$$

where ε is a function of the VC dimension h (a measure of complexity defined by the largest number of points that can be *shattered* by members of $f(\mathbf{x}, \alpha)$), η, and the number of instances m ($\epsilon = 4(h(ln(2m/h) + 1) - ln\ \eta)/m$). We observe that minimizing the expected risk is equivalent to simultaneously minimizing the two terms on the right-hand side of Eq. (1).

Considering a linear hyperplane of the form $f(\mathbf{x}) = \mathbf{w}^\top \cdot \mathbf{x} + b$, the SVM hyperplane then minimizes the classification errors and at the same time maximizes the *margin*, which is computed as the sum of the distances to one of the closest positive and one of the closest negative training examples, and is linked to Statistical Learning Theory since maximizing the margin is similar to minimizing the VC dimension [22].

To maximize the margin, we need to classify the training vectors $\mathbf{x}_i$ correctly into the two different classes, using the smallest norm of coefficients $\mathbf{w} \in \Re^n$ [18]. The primal SVM formulation balances the minimization of $\|\mathbf{w}\|_2^2$ (structural risk) and of the misclassification errors (empirical risk) by introducing an additional set of slack variables $\xi_i, i = 1, \ldots, m$ and a penalty parameter, C, that controls the trade-off between both objectives:

$$\begin{aligned} \min_{\mathbf{w},b,\xi} \quad & \tfrac{1}{2}\|\mathbf{w}\|_2^2 + C\sum_{i=1}^{m}\xi_i \\ \text{s.t.} \quad & y_i \cdot (\mathbf{w}^\top \cdot \mathbf{x}_i + b) \geqslant 1 - \xi_i, \quad i = 1, \ldots, m, \\ & \xi_i \geqslant 0, \qquad\qquad\qquad\quad i = 1, \ldots, m. \end{aligned} \tag{2}$$

2.2. l_1-Support Vector Machine

Bradley and Mangasarian [4] proposed a variation of SVM, reducing the model's complexity by using the l_1-norm (also known as LASSO penalty) instead of the Euclidean norm. This norm provides a strategy for suppressing redundant and/or irrelevant features automatically, i.e. components of the vector $\mathbf{w}$, while converting the quadratic programming problem studied by l_2-SVM (Formulation (2)) into a linear one. This formulation follows:

$$\begin{aligned} &\min_{\mathbf{w},b,\xi} \|\mathbf{w}\|_1 + C\sum_{i=1}^{m}\xi_i \\ &\text{s.t.} \quad y_i \cdot (\mathbf{w}^\top \cdot \mathbf{x}_i + b) \geqslant 1 - \xi_i, \quad i = 1,\ldots,m, \\ &\xi_i \geqslant 0, \qquad\qquad\qquad\qquad i = 1,\ldots,m. \end{aligned} \tag{3}$$

Formulation (3) can be cast into a linear programming model, tackling the sum of absolute values from vector $\mathbf{w}$ with the following formulation (l_1-SVM):

$$\begin{aligned} &\min_{\mathbf{w},\mathbf{v},b,\xi} \quad \sum_{j=1}^{n} v_j + C\sum_{i=1}^{m}\xi_i \\ &\text{s.t.} \quad y_i \cdot (\mathbf{w}^\top \cdot \mathbf{x}_i + b) \geqslant 1 - \xi_i, \quad i = 1,\ldots,m, \\ &-v_j \leqslant w_j \leqslant v_j, \qquad\qquad j = 1,\ldots,n, \\ &\xi_i \geqslant 0, \qquad\qquad\qquad\qquad i = 1,\ldots,m, \\ &v_j \geqslant 0, \qquad\qquad\qquad\qquad j = 1,\ldots,n. \end{aligned} \tag{4}$$

2.3. Linear programming Support Vector Machine

Another strategy based on SVM was proposed by [22], where the bound of the VC dimension is loosened properly using the l_∞-norm, resulting in a linear programming formulation that directly controls the margin maximization by considering a margin variable, r. The authors define a set of m_Δ-margin separating hyperplanes of the form $f(\mathbf{x}) = \mathbf{w}^\top \cdot \mathbf{x} + b$ as follows:

$$y = \begin{cases} +1 & \mathbf{w}^\top\mathbf{x} + b \geqslant \Delta \\ -1 & \mathbf{w}^\top\mathbf{x} + b \leqslant -\Delta \end{cases}, \tag{5}$$

with $\Delta \geqslant 0$, and determining a bound for the VC dimension, h, based on this family of hyperplanes, as shown in the following theorem.

Theorem 2.1 [22]. *Given a set of training instances $\mathbf{x}_i \in \Re^n$ and their respective labels $y_i \in \{-1,+1\}, i = 1,\ldots,m$, and a set of m_Δ-margin separating hyperplanes, then there exists a constant $0 < c < \infty$ such that:*

$$h \leqslant \min\left(\left[\frac{c^2R^2\|\mathbf{w}\|_\beta^2}{\Delta^2}\right], n\right) + 1 \tag{6}$$

holds true, where $\|\cdot\|_\beta$ is any vector norm.

The authors studied $\beta = \infty$ and $c = 1$. Based on this choice, the LP-SVM (soft-margin) formulation follows:

$$\begin{aligned} &\min_{\mathbf{w},r,b,\xi} \quad -r + C\sum_{i=1}^{m}\xi_i \\ &\text{s.t.} \quad y_i \cdot (\mathbf{w}^\top \cdot \mathbf{x}_i + b) \geqslant r - \xi_i, \quad i = 1,\ldots,m, \\ &-1 \leqslant w_j \leqslant 1, \qquad\qquad j = 1,\ldots,n. \\ &\xi_i \geqslant 0, \qquad\qquad\qquad\qquad i = 1,\ldots,m. \\ &r \geqslant 0, \end{aligned} \tag{7}$$

where as in previously presented models, C is a positive parameter that can be calibrated using cross-validation. The decision function of LP-SVM is also similar to standard SVM. The variable r is maximized while assuring - in the case of separable training examples – that each observation is on the correct side of the hyperplane, and at a distance at least r from it. For the non-separable case, the empirical risk is minimized simultaneously by penalizing a set of slack variables, similarly to standard SVM. Notice that the second constraint is related to the l_∞-norm of the weight vector.

The approach was tested on simulated and real datasets in Zhou et al. [22], leading to at least one order of magnitude improvement in training speed, making it particularly suitable for complex machine learning tasks, such as large scale problems or feature selection.

2.4. Related work on feature selection for SVMs

Guyon et al. [7] identified three main categories of methods for feature selection: filter, wrapper, and embedded methods. *Filter methods* eliminate poorly informative features based on their statistical properties prior to applying any classification algorithm. A commonly used filter method is the Fisher Criterion Score (F), which computes each feature's importance independently of the other features by comparing that feature's correlation to the output labels [7]:

$$F(j) = \left| \frac{\mu_j^+ - \mu_j^-}{(\sigma_j^+)^2 + (\sigma_j^-)^2} \right| \tag{8}$$

where μ_j^+ (μ_j^-) represents the mean of the j-th feature for the positive (negative) class and σ_j^+ (σ_j^-) is the respective standard deviation.

Wrapper methods interact with the respective classification technique and explore the entire set of variables to identify good feature subsets according to their predictive power, which is computationally demanding, but often provides better results than filter methods. Common wrapper strategies are Sequential Forward Selection (SFS) and Sequential Backward Elimination (SBE) [11]. A combination of filter methods and wrappers that focuses however, on fuzziness in the analyzed data has been presented by [17].

Techniques from the third category (*embedded methods*) select features and simultaneously construct the respective classifier, which can be seen as a search in the combined space of feature subsets and hypotheses. Unlike wrapper methods, which depend on a given but separate classification algorithm, in this category it is just one technique that performs both tasks, feature selection as well as classifier construction. In general, embedded methods have the advantage of being computationally less intensive than wrapper methods [7].

Recursive Feature Elimination (RFE-SVM) [8] is a popular embedded technique which tries to find a subset of size s among n variables ($s < n$), eliminating those whose removal leads to the largest margin of class separation. This can be achieved using a linear approach, based on the value of the weight vector $\mathbf{w}$. While one could choose a single variable to remove at each iteration, this would be inefficient in many high-dimensional applications (e.g. microarray data). Such datasets are often characterized by thousands of features, and their respective authors usually remove half of the remaining variables in each step [8].

Embedded feature selection can also be seen as an optimization problem. This is generally done by forcing feature selection into the model, considering a sparsity term in the objective function. One example is the minimization of the "zero-norm": $\Omega(\mathbf{w}) = \|\mathbf{w}\|_0 = |\{i : w_i \neq 0\}|$.

Weston et al. [20] proposed an approach for "zero-norm" minimization (l_0-SVM) by iteratively scaling the variables, multiplying them by the absolute value of the weight vector $\mathbf{w}$, which is obtained from the SVM formulation, until convergence is reached. Variables can be ranked by removing those features whose weights become zero during the iterative algorithm and computing the order of removal.

A mixed-integer program (MIP) has been proposed to select features iteratively for a non-linear SVM classifier [14]. In this approach any suitable kernel function can be used and the MIP is solved efficiently by alternating between a linear program which determines the continuous variable values of the classifier and successive updates of the binary variables indicating presence or absence of the respective features.

In an early work [10] titled "Feature Selection for Multiclass Discrimination via Mixed-Integer Linear Programming", a MILP for feature selection based on the assumption of feature independence had been introduced. Later, an alternative mixed-integer programming approach was proposed for simultaneous feature selection and multi-class classification [5]. This method modifies the l_1 multi-class SVM formulation to include costs on features, which has also been proposed for decision trees by Turney [16]. A biobjective optimization scheme is considered to maximize fit and minimize the total feature costs simultaneously, leading to an approximation of the set of Pareto-optimal classifiers.

Our work differs from previous ones in which a multi-objective approach balances both objectives requiring additional information regarding their trade-off. Instead, we propose an additional budget constraint providing direct control over the number of selected features and solving a Mixed Integer Linear Program with a single objective, as will be shown next.

3. Proposed SVM-based MILP formulations

In each one of the following two subsections we propose a model based on previously introduced SVM formulations, namely l_1-SVM and LP-SVM. In both cases, the main idea is to perform embedded feature selection by using a binary variable linked to each attribute, and to restrict the number of attributes used in the respective classifier via a budget constraint. We assume a cost vector, $\mathbf{c} \in \Re^n$, where c_j is the cost of acquiring attribute $j, j = 1, \ldots, n$. If no such cost information is provided or equal cost among attributes is desired, all parameters c_j can be set to 1. Both proposed models use a fixed "budget" B to limit the number of selected features. The difference between them is the respective norm used for the SVM formulation.

3.1. MILP Formulation based on l_1-SVM: MILP1

The following model emulates the l_1-SVM formulation described in Section 2.2, where the l_1-norm of $\mathbf{w}$, represented by $\sum_{j=1}^{n} v_j$, is minimized simultaneously with the minimization of the sum of the classification errors ($\sum_{i=1}^{m} \xi_i$), which requires the additional parameter C.

In our proposed MILP1 formulation we select features using a budget constraint instead of minimizing the l_1-norm of $\mathbf{w}$, and force each weight w_j to belong to a given interval $[l_j, u_j]$, if the attribute is selected (i.e. if $v_j = 1$). In this way we link the

weight vector $\mathbf{w}$ and the binary variables $\mathbf{v}$ in the formulation. Additionally, we avoid the inclusion of a trade-off parameter C since the shrinkage is now performed by the second and third constraints, instead of minimizing the l_1-norm in the objective function.

$$\begin{aligned}
&\min_{\mathbf{w},\mathbf{v},b,\xi} \quad \sum_{i=1}^{m}\xi_i \\
&\text{s.t.} \quad y_i\cdot(\mathbf{w}^\top\cdot\mathbf{x}_i+b)\geqslant 1-\xi_i, && i=1,\ldots,m,\\
&l_j v_j \leqslant w_j \leqslant u_j v_j, && j=1,\ldots,n.\\
&\sum_{j=1}^{n} c_j v_j \leqslant B.\\
&v_j\in\{0,1\}, && j=1,\ldots,n.\\
&\xi_i\geqslant 0, && i=1,\ldots,m.
\end{aligned} \tag{9}$$

The previous formulation presents interesting properties. For instance, we define a budget B explicitly, which represents the number of features in the classifier when all c_j are equal to 1. Additionally, the budget constraint allows incorporating acquisition costs directly into the formulation, encouraging a cheaper solution with an adequate level of accuracy.

The formulation, however, has a higher computational cost than linear or quadratic programming approaches. An important issue is the appropriate choice of the lower and upper bounds for the components of vector $\mathbf{w}$, i.e. $\mathbf{l}$ and $\mathbf{u}$. Of course, we can always choose lower and upper bounds with arbitrarily high values (positive or negative). However, the generic approach for solving MILP formulations like (9) consists of a Branch-and-Cut method whose efficiency depends heavily on the tightness of the model's LP-relaxation, i.e. how close the optimal values of the MILP and its LP-relaxation are, which in turn strongly depends on the tightness of the lower and upper bounds. In Section 4.4 we study the influence of these parameters.

3.2. MILP formulation based on LP-SVM: MILP2

The second formulation we propose extends the LP-SVM model to Mixed Integer Linear Programming. Similar to MILP1, MILP2 includes binary variables to activate the usage of the different attributes, limiting the number of selected variables in the classifier via a budget constraint. The main difference between the two models is that the weight vectors are now bounded by the interval $[-1,1]$, only if a particular variable is selected (i.e. if $v_j=1$), instead of a predefined interval $[l_j,u_j]$. These constraints are directly linked to the l_∞-norm of the weight vector used for the bound of the VC dimension described by Zhou et al. [22]. The second difference is that, for MILP2, a margin variable r is now maximized, in the attempt to find the m_Δ-margin separating hyperplane with maximum margin, instead of using the classical constraint based on the construction of the canonical hyperplanes. The proposed formulation follows:

$$\begin{aligned}
&\min_{\mathbf{w},r,b,\xi,\mathbf{v}} \quad -r+C\sum_{i=1}^{m}\xi_i \\
&\text{s.t.} \quad y_i\cdot(\mathbf{w}^\top\cdot\mathbf{x}_i+b)\geqslant r-\xi_i, && i=1,\ldots,m,\\
&-v_j\leqslant w_j\leqslant v_j, && j=1,\ldots,n.\\
&\sum_{j=1}^{n} c_j v_j \leqslant B,\\
&\xi_i\geqslant 0, && i=1,\ldots,m.\\
&v_j\in\{0,1\}, && j=1,\ldots,n.
\end{aligned} \tag{10}$$

Additionally, we identified the following two practical pitfalls in the LP-SVM formulation presented in Section 2.3, which it is necessary for us to address in our proposed formulation, given the higher complexity of our approach.

- One possible solution of the optimization problem is that all variables become zero. In that extreme case, all object labels will be predicted as zero, resulting in an accuracy of 0%. High values of C in noisy data may trigger that issue. This can be avoided by using a lower bound $r_{lo}>0$ for variable r. In our experiments we use the value $r_{lo}=0.001$.
- Another issue is that the variables r and ξ_i may grow unboundedly, given their relationship in the objective function. When this happens, results are very inaccurate. Setting an upper bound on variable r (r_{up}) avoids this effect, also controlling the growth of the variables ξ_i. Different values for r_{up} are studied using line search; see Section 4.4.

To address both issues, we included an additional constraint to bound the margin variable r between both lower and upper values.

$$
\begin{aligned}
&\min_{\mathbf{w},r,b,\xi,\mathbf{v}} \quad -r + C\sum_{i=1}^{m}\xi_i \\
&\text{s.t.} \quad y_i \cdot (\mathbf{w}^\top \cdot \mathbf{x}_i + b) \geqslant r - \xi_i, \quad i = 1,\ldots,m, \\
&-v_j \leqslant w_j \leqslant v_j, \qquad\qquad j = 1,\ldots,n. \\
&\sum_{j=1}^{n} c_j v_j \leqslant B, \\
&\xi_i \geqslant 0, \qquad\qquad i = 1,\ldots,m. \\
&v_j \in \{0,1\}, \qquad\qquad j = 1,\ldots,n. \\
&r_{lo} \leqslant r \leqslant r_{up},
\end{aligned}
\tag{11}
$$

This formulation does not require the determination of lower and upper bounds (i.e. l_j, u_j) for the variables w_j, which in model (11) can take values only from the interval $[-1,1]$. The desired flexibility for variables w_j is achieved indirectly by using the non-negative variable r which explicitly considers the structural risk minimization principle. As a consequence, model (11) requires the additional regularization parameter C. The final constraint in model (11) prohibits r becoming zero or diverging.

4. Experimental results

In this section we apply the classification models l_2-Support Vector Machines (Formulation (2)) and LP-Support Vector Machines (Formulation (7)) as well as the feature selection approaches l_1-Support Vector Machines (Formulation (4)), l_0-Support Vector Machines, and the two benchmark techniques for feature selection (Fischer + SVM and RFE-SVM) to various datasets, in comparison with the newly proposed formulations for simultaneous feature selection and classification via Mixed-Integer Linear Programming (MILP1: Model (9) and MILP2: Model (11)).

These datasets are presented in Section 4.1. Subsequently, we describe our model selection procedure. Section 4.3 presents the results we obtained. The influence of the different parameters on robustness and stability are studied in Section 4.4. Finally, we analyze running times for all methods used in our experiments in Section 4.5.

4.1. Datasets

We applied the proposed approaches on six well-known datasets from the UCI Repository [3]. These datasets have already been used for benchmark studies regarding the performance of Support Vector Machines (see e.g. [1,15]).

- **Australian Credit (AUS)**: This dataset contains 690 granted loans; 383 good repayers and 307 bad repayers, described by 14 variables.
- **Wisconsin Breast Cancer (WBC)**: This dataset contains 569 observations of tissues (212 malignant and 357 benign tumors) described by 30 continuous features.
- **Pima Indians Diabetes (PIMA)**: The Pima Indians Diabetes dataset presents 8 features and 768 examples (500 tested negative for diabetes and 268 tested positive).
- **German Credit (GC)**: This dataset presents 1000 granted loans; 700 good repayers and 300 bad repayers, described by 24 attributes.
- **Ionosphere (IONO)**: This dataset presents 351 data points; 225 labeled as *good* radar returns (evidence of some type of structure in the Ionosphere) and 126 labeled as *bad* radar returns (no evidence of structure), described by 34 attributes.
- **Splice**: This dataset contains 1000 randomly selected examples (from the complete set of 3190 splice junctions), where 517 are labeled as IE borders and 483 as EI borders, described by 60 categorical variables (the gene sequence). Given a DNA sequence, the problem posed in this dataset is to recognize the boundaries between exons and introns (the parts of the sequence retained after splicing and the parts that are spliced out, respectively).

Additionally, we analyzed a microarray dataset in order to study the performance of the proposed methods under conditions of high dimensionality with a small number of examples.

- Colorectal Microarray (CoMA) [2]: This dataset contains the expression of the 2000 genes with highest minimal intensity across 62 samples (40 tumor and 22 normal).

4.2. Model selection

The following model selection procedure was performed: training and test subsets were constructed using 10-fold cross-validation and the average AUC (Area Under Curve) was computed. For the microarray dataset we followed the procedure presented in [19]: training and test subsets were obtained using a leave-one-out procedure. Feature selection and classifi-

cation were then performed on the training set, and the classification performance was finally computed from test results. We performed a grid search to study the influence of parameter C for soft-margin models, and other model-specific parameters; (see Section 4.4).

The intervals were further divided into homogenous values in order to characterize the relevant area for this parameter (where maximum predictive performance is reached). The parameter B for the budget constraint was varied along all possible numbers of attributes for the first six datasets. For the Colorectal Microarray dataset the following budget values were studied:

$B \in \{10, 20, 50, 100, 250, 500, 1000, 2000\}$.

All values of the feature cost vector $\mathbf{c}$ were set to 1 since we do not have specific cost information in our experiments. The optimization was performed using LIBSVM in the case of l_2-Support Vector Machines, LINPROG solver for Matlab 7.8 in the case of l_1-Support Vector Machines (Formulation (4)), and CPLEX 9.1 solver for the LP-Support Vector Machines, MILP1, and MILP2 approaches. The experiments were performed in a Lenovo x220t with 16 GB RAM, 750 GB SSD, a i7-2620M processor with 2.70 GHz, and Microsoft Windows 8.1 Operating System (64-bit).

4.3. Results

Tables 1–3 summarize the predictive performance for all methods along all datasets for the best combinations of parameters, and for the best subset of features, in terms of AUC. Best results for each dataset are presented in bold. In case of identical AUC, the solution with fewer variables is considered best. We also report Accuracy (ACC) using 0.5 as the cut-off value.

From previous tables we observe that the best predictive performance is achieved with the proposed models MILP1 and MILP2 in all seven cases. For Australian Credit, the best AUC is achieved using MILP2 with 10 attributes. A similar performance results with standard SVM and LP-SVM using all variables, but in this study, a solution with fewer features is preferred in case of similar AUC. However, the difference in terms of classification performance among all approaches is not significant. For the WBC dataset, best results are obtained with MILP1 using 26 out of 30 attributes, representing a significant improvement compared with all other methods. For the PIMA dataset, similar results are obtained with all approaches.

For the German Credit dataset, the best performance is achieved with RFE-SVM and MILP1, where MILP1 is preferred since the best solution is obtained with fewer attributes. For the Ionosphere dataset, results are better in terms of AUC with MILP1 including 19 out of 34 variables. MILP2 performed better for the Splice dataset (although not statistically significant) using 35 out of 60 attributes. For Colorectal Microarray data, both proposed approaches achieved remarkably better performance compared to alternative feature selection approaches, and MILP1 achieved slightly better AUC.

In order to compare the predictive performance of feature selection approaches along different subsets of attributes, a comparison in terms of AUC is presented for all datasets in Figs. 1–7. The proposed approaches, MILP1 and MILP2, are presented, together with the best alternative approach in terms of predictive AUC.

In Fig. 1 we observe that, for Australian Credit data, all approaches behave very similarly along all different subsets where the first attributes are the only relevant ones in this case. MILP2 performed slightly better for $k = 10$ and $k = 12$, while MILP1 had slightly worse behavior overall.

For the WBC dataset (Fig. 2), the proposed approaches are consistently better along all different feature subsets, and the best performance is achieved using MILP1.

The PIMA Diabetes dataset (Fig. 3) has few attributes and experiments proved that all of them seem to be relevant. A slightly better performance is achieved with MILP2 using all attributes. All feature selection methods behave relatively similarly.

For the German Credit dataset (Fig. 4), the best performance is obtained with MILP1 and l_0-SVM, and the gain of using fewer attributes is significant compared to the selection of all features.

For the Ionosphere dataset (Fig. 5), the best results are obtained with MILP1 using about half of the attributes, significantly improving the solution obtained with all features.

MILP2 obtained the best results for the SPLICE dataset (Fig. 6), achieving similar AUC compared to standard SVM but with about half the number of features.

Table 1
Best accuracy and AUC, in percentage, and number of selected features (k) for AUS, WBC, and PIMA datasets.

	Aus. Credit			Wisconsin Breast Cancer			PIMA Diabetes		
	ACC	AUC	k	ACC	AUC	k	ACC	AUC	k
l_2-SVM	85.7	86.3	14	97.9	97.3	34	77.9	73.3	8
LP-SVM	85.7	86.3	14	97.2	96.5	34	77.9	73.3	8
l_1-SVM	85.5	86.2	12	97.5	97.2	10	77.9	73.3	8
Fisher + SVM	85.5	86.2	2	97.9	97.3	20	77.5	72.4	7
RFE-SVM	85.5	86.2	2	97.9	97.3	23	77.1	71.8	3
l_0-SVM	85.5	86.2	2	97.9	97.3	16	77.0	71.8	5
MILP1	85.5	86.2	2	98.1	**97.7**	**26**	77.9	73.3	8
MILP2	85.7	**86.3**	**10**	97.9	97.3	17	78.0	**73.4**	**8**

Table 2
Best accuracy and AUC, in percentage, and number of selected features (k) for GC, IONO, and Splice datasets.

	German Credit			Ionosphere			Splice		
	ACC	AUC	k	ACC	AUC	k	ACC	AUC	k
l_2-SVM	76.7	69.1	30	88.6	85.2	34	81.5	81.6	60
LP-SVM	77.1	69.4	30	87.2	84.1	34	80.6	80.7	60
l_1-SVM	76.8	69.0	24	88.3	85.0	29	81.0	81.1	44
Fisher + SVM	77.3	69.5	22	87.7	84.3	30	81.4	81.5	43
RFE-SVM	77.5	69.8	22	88.3	84.7	8	81.1	81.2	16
l_0-SVM	76.5	68.5	22	88.9	85.7	16	81.4	81.5	24
MILP1	77.5	**69.8**	**20**	88.6	**86.0**	**19**	80.9	81.0	18
MILP2	77.0	69.3	21	88.1	85.0	19	81.5	**81.6**	**35**

Table 3
Best accuracy and AUC, in percentage, and number of selected features (k) for Colorectal Microarray dataset.

	Colorectal Microarray		
	ACC	AUC	k
l_2-SVM	83.9	83.4	2000
LP-SVM	87.1	86.9	2000
l_1-SVM	87.1	85.9	217
Fisher + SVM	83.9	83.4	50
RFE-SVM	85.5	84.7	500
l_0-SVM	83.9	82.4	100
MILP1	85.5	**90.3**	**100**
MILP2	90.3	89.4	50

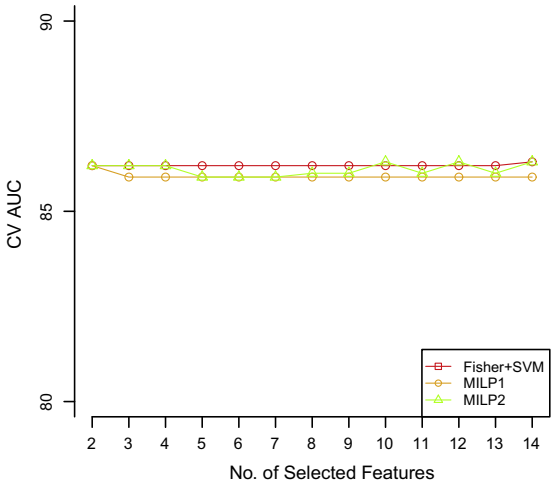

Fig. 1. AUC versus the number of ranked variables for different feature selection approaches. Aus. Credit dataset.

Finally, for the Colorectal Microarray data (Fig. 7), which is the highest-dimensional set of our experiments, results are remarkably better using the proposed approaches with 50 and 100 variables respectively instead of the entire set of 2000 attributes, also outperforming the alternative methods in this application. Fisher + SVM behaved unstably for this dataset, which is somehow expected given the size of the dataset (62 instances). Since the Fisher Criterion Score does not take the correlation between variables into account, which is a major issue in microarray data (see [8]), it is possible that the feature removal led to poor performance at the beginning, while its performance improved when reaching 50 variables. In any case, similar behavior was reported in Guyon et al. [8] for the same dataset (Fig. 4).

In summary, we can classify our results according to the dimensionality of the dataset as follows:

- For low-dimensional datasets, performance either remains stable (Australian Credit) or decreases smoothly (PIMA Diabetes), with very low variance between feature selection methods, achieving no gain in performance compared to classification methods without feature selection.

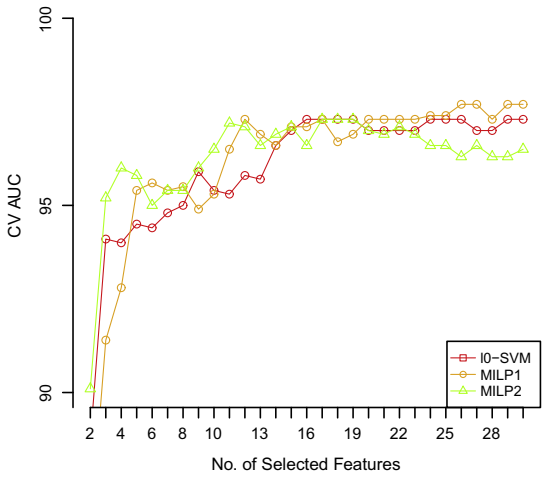

Fig. 2. AUC versus the number of ranked variables for different feature selection approaches. Wisconsin Breast Cancer dataset.

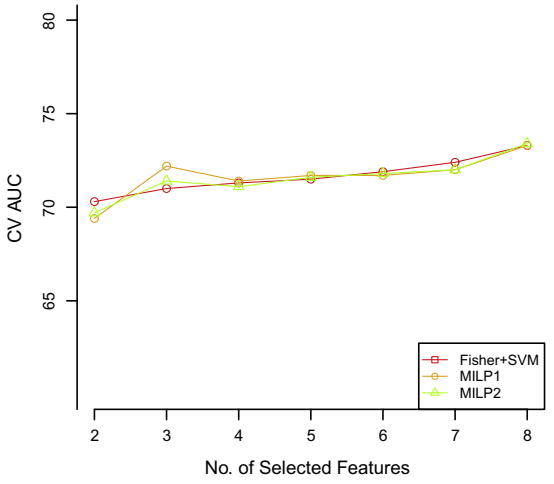

Fig. 3. AUC versus the number of ranked variables for different feature selection approaches. PIMA Diabetes dataset.

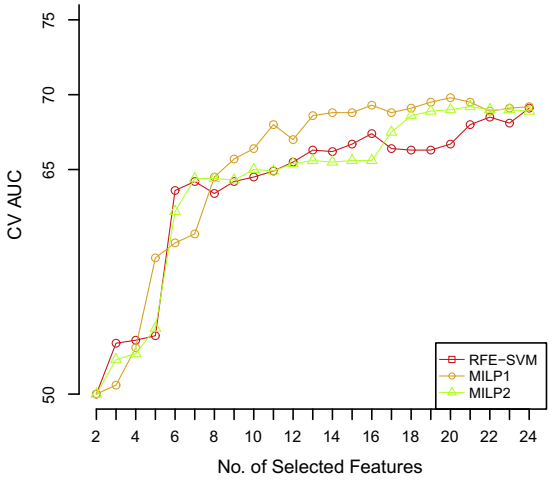

Fig. 4. AUC versus the number of ranked variables for different feature selection approaches. German Credit dataset.

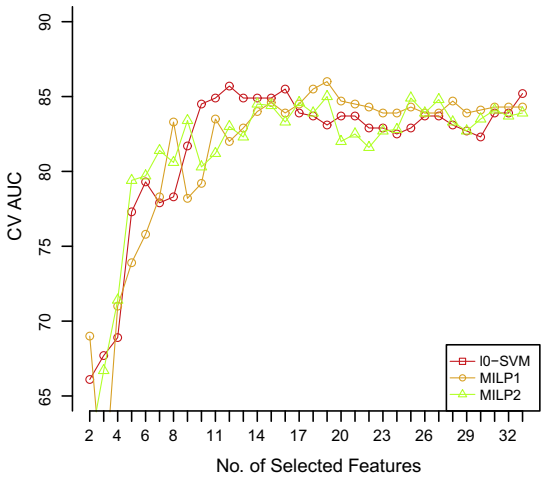

Fig. 5. AUC versus the number of ranked variables for different feature selection approaches. Ionosphere dataset.

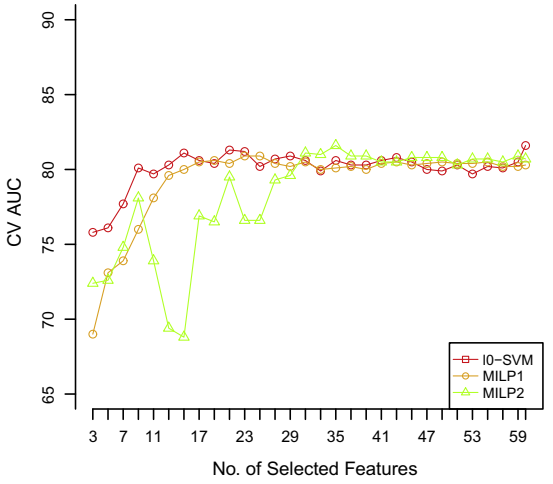

Fig. 6. AUC versus the number of ranked variables for different feature selection approaches. Splice dataset.

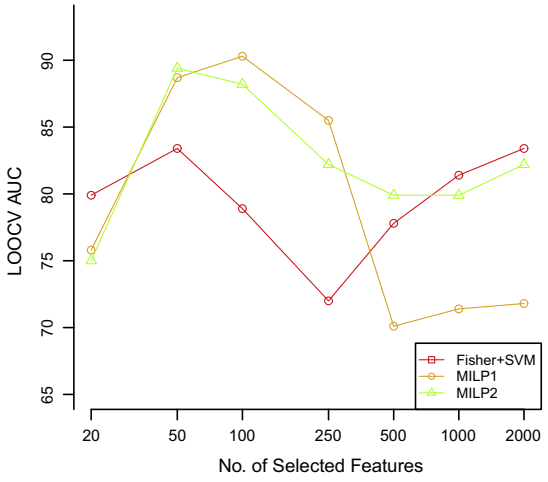

Fig. 7. AUC versus the number of ranked variables for different feature selection approaches. Colorectal Microarray dataset.

- For medium-sized datasets, we observed relatively similar behavior for most methods, achieving slight improvements in terms of AUC while decreasing the number of selected attributes, and finally decreasing their performance with below 10–15 variables.
- For a high-dimensional dataset such as Colorectal Microarray, feature selection provides significant improvements in terms of AUC (up to 6%) while reducing the number of variables used for classification to 10% or less compared to the original size.

4.4. Influence of parameters

The proposed approaches consider various parameters that need to be studied in order to understand the robustness and stability of the respective methods. We varied parameter C and the upper bound for the margin variable r (r_{up}) for model MILP2 (Formulation (11)).

For model MILP1 (Formulation (9)) we analyzed the weight vectors' bounds (l_j and u_j).

Assuming $l_j = -u_j$ and $u_j = u,\ \forall j$, we used the following set of values for parameters C (previously used in [12,13]), u, and r_{up}:

$$\{2^{-7}, 2^{-6}, 2^{-5}, 2^{-4}, 2^{-3}, 2^{-2}, 2^{-1}, 2^{0}, 2^{1}, 2^{2}, 2^{3}, 2^{4}, 2^{5}, 2^{6}, 2^{7}\}$$

Tables 4–6 present the results of the mean cross-validation AUC performance (leave-one-out AUC performance in the case of microarray datasets) obtained by varying parameters u for MILP1 as well as C and r_{up} for MILP2. Since the best result in terms of AUC is always found for u and r_{up} parameter values above 1, and to reduce the size of Tables 5 and 6, only their lower parts are displayed.

From previous tables we observe that the proposed methods, MILP1 and MILP2, are very robust and stable along the different values of the analyzed parameters. Relevant differences arise only for microarray data. Parameters should be tuned carefully in this type of dataset. All parameters have similar influence on the predictive performance of the methods.

4.5. Running times

The proposed approaches are based on Mixed-Integer Programming formulations, which are known to be very time-consuming and therefore, in general, less suitable for machine learning where huge datasets are to be analyzed. Table 7 provides a comparison for one run of each method (the average running time for one fold using 10-fold cross-validation or leave-one-out in the case of Microarray datasets, considering the best set of parameters obtained using the model selection procedure). The first group of models does not involve feature selection, which is the case for standard SVM (l_2-SVM), LP-SVM, and both proposed methods when the budget constraint is not included, i.e. MILP1-NFS and MILP2-NFS. The second group of models has feature selection approaches, and the displayed values in Table 7 are the running times until each method reaches its optimal number of features.

It is important to notice that all running times are tractable and reasonable (all running times below one minute, and between 1 and 2 s in most cases). For the first group of methods (no feature selection), running times are very short and very similar. For the second group of methods, we observe that the fastest approach is the Fisher Score with SVM classification, which is almost as fast as using SVM without feature selection. MILP1 is, on average, faster than the other methods, and all approaches achieve relatively similar results in most cases. In particular, for the first three datasets (the simplest ones) our

Table 4
Predictive performance (AUC) obtained by varying parameter C.

	AUS	WBC	PIMA	GC	IONO	SPLICE	CoMA
2^{-7}	86.3	96.5	73.3	69.2	81.1	80.7	74.1
2^{-6}	85.7	94.3	72.6	69.2	83.9	79.8	75.1
2^{-5}	85.7	95.3	73	69.1	84.1	79.8	80.9
2^{-4}	85.7	94.6	72.6	69.2	83.9	80.3	72.8
2^{-3}	85.7	95.6	73	69.2	83.9	79.6	76.4
2^{-2}	85.7	95.2	73.1	68.9	84.1	79.9	72.8
2^{-1}	85.7	94.8	72.8	68.7	84.1	79.8	77.4
2^{0}	85.7	94.8	73.4	69.1	83.9	79.6	64.8
2^{1}	85.7	94.8	73.3	69.3	83.9	79.7	70.6
2^{2}	85.7	94.9	73.2	69	83.9	79.8	73.9
2^{3}	85.7	95.2	73.1	68.6	84.1	79.5	71.8
2^{4}	85.7	95.3	73.1	69.1	83.9	80.1	70.3
2^{5}	85.7	94.9	73.1	68.9	83.9	80	63.5
2^{6}	85.7	94.9	73.1	69.1	83.9	79.9	70.1
2^{7}	85.7	94.9	73.1	69.1	83.9	80	71.4

Table 5
Predictive performance (AUC) obtained by varying parameter u.

	AUS	WBC	PIMA	GC	IONO	SPLICE	CoMA
2^0	85.9	95.4	65.9	69.2	81.3	80.3	82.6
2^1	85.7	97.7	72.4	69.1	82.5	79.8	85.7
2^2	85.7	96.7	73.3	69.1	83.5	79.8	76.1
2^3	85.7	96.6	73.3	69.1	84.2	79.8	70.6
2^4	85.7	95.9	73.3	69.1	84.3	79.8	67.3
2^5	85.7	95.6	73.3	69.1	83.9	79.8	62.3
2^6	85.7	95.1	73.3	69.1	83.9	79.8	70.3
2^7	85.7	94.5	73.3	69.1	83.9	79.8	68.3

Table 6
Predictive performance (AUC) obtained by varying parameter r_{up}.

	AUS	WBC	PIMA	GC	IONO	SPLICE	CoMA
2^0	85.9	96.5	73.3	68.9	83.9	80.3	76.4
2^1	86.3	96.5	73.3	68.9	83.9	80.7	81.1
2^2	86.3	96.5	73.3	68.9	83.9	80.7	76.4
2^3	86.3	96.5	73.3	68.9	83.9	80.7	82.2
2^4	86.3	96.5	73.3	68.9	83.9	80.7	79.7
2^5	86.3	96.5	73.3	68.9	83.9	80.7	82.2
2^6	86.3	96.5	73.3	68.9	83.9	80.7	86.9
2^7	86.3	96.5	73.3	68.9	83.9	80.7	86.9

Table 7
Average running times, in seconds, for all datasets.

	AUS	WBC	PIMA	GC	IONO	SPLICE	CoMA
l_2-SVM	0.5	0.2	0.3	0.7	0.2	1.5	0.2
LP-SVM	0.2	0.1	0.1	0.3	0.1	0.4	0.4
MILP1-NFS	0.2	0.2	0.2	0.4	0.2	0.7	0.4
MILP2-NFS	0.2	0.3	0.2	0.4	0.2	1.4	0.6
Fisher + SVM	0.5	0.2	0.4	0.7	0.2	1.5	0.2
l_1-SVM	0.4	0.4	0.3	0.4	0.3	4.8	0.6
RFE-SVM	0.8	0.4	0.6	0.7	0.6	2.3	1.7
l_0-SVM	0.8	0.5	1.3	1.7	0.4	2.4	1.8
MILP1-FS	0.2	0.2	0.2	0.3	0.2	1.7	2.2
MILP2-FS	0.3	0.2	0.3	0.9	2.3	51.6	3.2

approaches are faster than the alternative feature selection methods. MILP2, however, presents significantly higher values for Ionosphere and Splice datasets, especially in the latter. This phenomenon could be due to the complexity of the datasets (highest number of attributes except for the Colorectal Microarray data, with a higher number of instances). Another reason is that the highest computational times for the proposed methods occur when the budget is about half of the number of original variables, while the fastest experiments are obtained when the budget constraint is not activated and all features are used, i.e. no feature selection is performed. For the Splice dataset, the optimal solution for MILP2 is found for a subset of 35 out of 60 attributes, which is also the best performance among all methods. For MILP2 we also observed an important dependence between performance (AUC and running times) and the bounds for the margin variable r, which causes higher average times compared to MILP1.

5. Conclusions

In this work we presented two embedded approaches for simultaneous feature selection and classification based on Mixed Integer Linear Programming and Support Vector Machines. The main idea is to perform attribute selection by introducing binary variables, obtaining a low-dimensional SVM classifier. Two different SVM-based linear programming formulations, namely l_1-SVM and LP-SVM, were adapted to Mixed-Integer Programming formulations. A comparison with other feature selection approaches for SVM in low- and high-dimensional datasets showed the advantages of the proposed methods:

- They allow the construction of a classifier for a desired number of attributes without the need of two-step methodologies that perform feature selection and classification independently.

- They provide better predictive performance compared to feature ranking techniques, based on their ability to discard irrelevant attributes by limiting the number of features in the model via a budget constraint.
- Our approaches find the optimal subset of features in one run, given a predefined number of attributes (the budget parameter), while sequential approaches require multiple runs and successive training steps to reach a desired number of features.
- They determine an optimal solution for the feature selection problem in reasonable running times given a predefined number of features.

From the experimental section of this work, several conclusions can be drawn. Predictive performance (in terms of AUC) can be improved with fewer variables, demonstrating the relevance of feature selection. In our experiments, in all seven datasets a gain in terms of performance was achieved using feature selection, or at least performance was maintained.

By contrast, for microarray data, and in particular for the Colorectal Microarray, our approaches led to a significant improvement in terms of performance (a gain of almost 7% in terms of AUC using MILP1). In general, our models performed consistently better than alternative feature selection approaches. Additionally, the results of the proposed models proved to be robust and stable for different values of the parameters used for calibration. Finally, the algorithms' running times are adequate for most machine learning tasks, such as classification of microarray data.

There are several opportunities for future work. First, the extension of the proposed methods to kernel approaches may lead to better performance thanks to the ability of constructing non-linear classifiers, while selecting the relevant attributes in the original space. The main challenge is to incorporate binary variables associated with the weight vector into a kernel-based formulation.

Secondly, although in this work all attributes are treated equally, the proposed approach has the potential of incorporating different costs of different features in the budget constraint. Credit scoring, fraud detection, and churn prediction are some interesting application areas where the acquisition costs of each attribute may differ, and those costs can be estimated in order to construct a classifier that constrains or minimizes acquisition costs while classifying adequately.

Acknowledgments

Support from the Institute of Complex Engineering Systems (ICM: P-05-004-F, CONICYT: FBO16) (http://www.isci.cl) is greatly acknowledged. The first author was supported by FONDECYT project 11121196. The research of the last author is supported by the Interuniversity Attraction Poles Programme initiated by the Belgian Science Policy Office.

References

[1] S. Ali, K.A. Smith-Miles, On learning algorithm selection for classification, Appl. Soft Comput. 6 (2006) 119–138.
[2] U. Alon, N. Barkai, D.A. Notterman, K. Gish, S. Ybarra, D. Mack, A.J. Levine, Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligo-nucleotide arrays, in: Proceedings of the National Academy of Sciences, pp. 6745–6750.
[3] A. Asuncion, D. Newman, UCI Machine Learning Repository, 2007.
[4] P. Bradley, O. Mangasarian, Feature selection vía concave minimization and support vector machines, in: Machine Learning proceedings of the Fifteenth International Conference (ICML'98) 82–90, San Francisco, California, Morgan Kaufmann.
[5] E. Carrizosa, B. Martín-Barragán, D. Romero-Morales, Multi-group support vector machines with measurement costs: a biobjective approach, Discrete Appl. Math. 156 (2008) 950–966.
[6] E. Carrizosa, B. Martín-Barragán, D. Romero-Morales, Detecting relevant variables and interactions in supervised classification, Euro. J. Oper. Res. 213 (2011) 260–269.
[7] I. Guyon, S. Gunn, M. Nikravesh, L.A. Zadeh, Feature Extraction, Foundations and Applications, Springer, Berlin, 2006.
[8] I. Guyon, J. Weston, S. Barnhill, V. Vapnik, Gene selection for cancer classification using support vector machines, Mach. Learn. 46 (2002) 389–422.
[9] R. Hassan, R.M. Othman, P. Saad, S. Kasim, A compact hybrid feature vector for an accurate secondary structure prediction, Inform. Sci. 181 (2011) 5267–5277.
[10] F.J. Iannarilli, P.A. Rubin, Feature selection for multiclass discrimination via mixed-integer linear programming, IEEE Trans. Pattern Anal. Mach. Intell. 25 (2003) 779–783.
[11] J. Kittler, Pattern Recognition and Signal Processing, Pattern Recognition and Signal Processing, Sijthoff and Noordhoff, Netherlands, 1978. pp. 41–60.
[12] S. Maldonado, J. López, Imbalanced data classification using second-order cone programming support vector machines, Pattern Recogn. 47 (2014) 2070–2079.
[13] S. Maldonado, R. Weber, J. Basak, Kernel-penalized SVM for feature selection, Inform. Sci. 181 (2011) 115–128.
[14] O.L. Mangasarian, E.W. Wild, Feature selection for nonlinear kernel support vector machines, in: Seventh IEEE International Conference on Data Mining, IEEE, Omaha, NE, 2007, pp. 231–236.
[15] L. Song, A. Smola, A. Gretton, J. Bedo, K. Borgwardt, Feature selection via dependence maximization, J. Mach. Learn. Res. 13 (2012) 1393–1434.
[16] P. Turney, Cost-sensitive classification: empirical evaluation of a hybrid genetic decision tree induction algorithm, J. Artif. Intell. Res. 2 (1995) 369–409.
[17] O. Uncu, I.B. Türksen, A novel feature selection approach: combining feature wrappers and filters, Inform. Sci. 177 (2007) 449–466.
[18] V. Vapnik, Statistical Learning Theory, John Wiley and Sons, 1998.
[19] G. Victo Sudha George, V. Cyril Raj, Review on feature selection techniques and the impact of svm for cancer classification using gene expression profile, Int. J. Comput. Sci. Eng. Surv. 2 (3) (2011) 16–27.
[20] J. Weston, A. Elisseeff, B. Schölkopf, M. Tipping, The use of zero-norm with linear models and kernel methods, J. Mach. Learn. Res. 3 (2003) 1439–1461.
[21] H. Yu, J. Kim, Y. Kim, S. Hwang, Y.H. Lee, An efficient method for learning nonlinear ranking SVM functions, Inform. Sci. 209 (2012) 37–48.
[22] W. Zhou, L. Zhang, L. Jiao, Linear programming support vector machines, Pattern Recogn. 35 (2002) 2927–2936.