Feature Selection for Support Vector Machines via Mixed Integer Linear Programming

Sebastián Maldonado[1], Juan Pérez[1], Martine Labbé[2] and Richard Weber[3]

[1]Universidad de los Andes, Av. San Carlos de Apoquindo 2200, Las Condes, Santiago, Chile.

[2]Computer Science Department, Université Libre de Bruxelles, Boulevard du Triomphe, B-1050 Brussels, Belgium.

[3]Department of Industrial Engineering, Universidad de Chile, República 701, Santiago, Chile.

August 4, 2013

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 largely studied in the literature. Most strategies propose the elimination of features independently of classifier construction by exploiting statistical properties of the variables, or via greedy search heuristics. 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 to classifier construction using optimization models. We run experiments on real-world benchmark datasets, including microarray data, comparing our approach with well-known feature selection techniques and obtaining better predictions with consistently fewer relevant features.

Keywords: Feature selection, Support Vector Machines, Mixed Integer Programming.

1 Introduction

Feature selection is one of the most important machine learning tasks. An appropriate selection of the most relevant features reduces the risk of overfitting, improving model generalization by decreasing the model's complexity (Guyon et al., 2006). 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 (Hassan et al., 2011; Maldonado et al., 2011). Additionally, a low-dimensional representation allows a better interpretation of the classifier. This is particularly important in some application fields like business analytics, since machine learning approaches are considered as "black boxes" by practitioners, and therefore they tend to be reticent to use these techniques (Carrizosa et al., 2011). The understanding of the process that generates the data is also of crucial importance in life sciences, e.g., the relevant genes that lead to a better discrimination in cancer prediction.

Support Vector Machines (SVMs) has shown to be a very powerful machine learning method. Based on the structural risk minimization principle (Vapnik, 1998), this method attempts to find the separating hyperplane which has the

largest distance to the nearest training data point 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 representation that depends on only a few parameters (Vapnik, 1998; Yu et al., 2012). In this work 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 (Bradley and Mangasarian, 1998), while the second one extends the ideas of the LP-SVM method (Zhou et al., 2002).

The paper is structured as follows. Section 2 introduces Support Vector Machines for binary classification, and its robust formulation with second-order cones. Recent developments for feature selection using SVMs are reviewed in Section 3. The proposed feature selection approaches are presented in Section 4. Section 5 provides experimental results using real-world datasets. A summary of this paper can be found in Section 6, where we provide its main conclusions and address future developments.

2 Support Vector Classification

In this section we describe the mathematical derivation of the standard l_2-SVM (Vapnik, 1998), the l_1-SVM formulation (Bradley and Mangasarian, 1998), and the LP-SVM method (Zhou et al., 2002). These linear classification methods constitute the basis for our proposed feature selection algorithms.

2.1 l_2 Support Vector Machine

Considering training examples $\mathbf{x}_i \in \Re^n$ and their respective labels $y_i \in \{-1, +1\}$, $i = 1, \ldots, m$, SVM determines an hyperplane $f(\mathbf{x}) = \mathbf{w}^\top \cdot \mathbf{x} + b$ to optimally

separate the training examples. This hyperplane minimizes the classification errors and at the same time maximizes the *margin*, which is computed as the sum of the distances to the closest positive and negative training examples. To maximize this measure, we need to correctly classify the training vectors $\mathbf{x}_i$ into the two different classes, using the smallest norm of coefficients $\mathbf{w}$ (Vapnik, 1998). 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 ξ_i, $i = 1, \ldots, m$ and a penalty parameter C that controls the trade-off between both objectives:

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

2.2 l_1 Support Vector Machine

In order to suppress features, i.e. components of the vector $\mathbf{w}$, the l_1-norm is used as feature penalty. In Bradley and Mangasarian (1998), the lasso penalty led to good feature selection and classification results.

$$\begin{aligned} \min_{\mathbf{w},b,\boldsymbol{\xi}} \quad & \|\mathbf{w}\|_1 + C\sum_{i=1}^{m}\xi_i \\ \text{s.t.} \quad & y_i \cdot (\mathbf{w}^\top \cdot \mathbf{x}_i + b) \geq 1 - \xi_i, \qquad i = 1, \ldots, m, \\ & \xi_i \geq 0, \qquad i = 1, \ldots, m. \end{aligned} \tag{2}$$

which can be solved as a linear program, tackling the sums of absolute values from vector $\mathbf{w}$ with the following formulation (l_1-SVM):

$$
\begin{aligned}
\min_{\mathbf{w},\mathbf{v},b,\boldsymbol{\xi}} \quad & \sum_{j=1}^{n} v_j + C \sum_{i=1}^{m} \xi_i \\
\text{s.t.} \quad & y_i \cdot (\mathbf{w}^\top \cdot \mathbf{x}_i + b) \geq 1 - \xi_i, \qquad i = 1, \ldots, m, \\
& -v_j \leq \ w_j \leq v_j, \qquad\qquad j = 1, \ldots, n. \\
& \xi_i \geq 0, \qquad\qquad i = 1, \ldots, m.
\end{aligned}
\tag{3}
$$

2.3 Linear Programming Support Vector Machine

In linear programming SVMs, in order to improve training time, the bound of the VC dimension is loosened properly (Zhou et al., 2002) using the l_∞-norm, resulting in a linear programming formulation that controls the margin maximization directly by considering a margin variable r. This variable is then maximized while assuring - in the case of separable training examples - that each observation is on the correct side of the hyperplane, and at least at a distance r from it. For the non-separable case, the empirical risk is simultaneously minimized by penalizing a set of slack variables, similarly to standard SVM. The LP-SVM (soft-margin) formulation follows.

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

where 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 approach

was tested on simulated and real datasets in Zhou et al. (2002), leading to at least an order of magnitude improvement in training speed, making it particularly suitable for complex machine learning tasks, such as large scale problems or feature selection. Even if the VC dimension of LP-SVM is larger than that of l_2-SVM, its generalization error as obtained by the authors was smaller than for l_2-SVM in most cases, concluding that the loss in terms of structural risk is tolerable.

Formulation (4) presents some issues with noisy datasets. The following two pitfalls were identified.

- 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. In order to avoid it, a lower bound $r_{lo} > 0$ for variable r can be set.

- 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. To avoid this situation, an upper bound on variable r (r_{up}) can be set, controlling also the growth of the variables ξ_i.

Both bounds will be introduced in our model presented in Section 4.2.

3 Related Work on Feature Selection for SVMs

Guyon et al. (2006) 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 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 (Guyon et al., 2006):

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

where μ_j^+ (μ_j^-) represents the j-th feature's mean 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) (Kittler, 1978). In the first case, starting without any variable, the method tries out the feature candidates one by one and includes the most relevant one at each iteration. On the other hand, SBE starts with all candidate features and tests them one by one for statistical significance, deleting any variable that is not significant. A combination of filter methods and wrappers that focusses, however, on fuzziness in the analyzed data has been presented by (Uncu and Türksen, 2007).

Techniques from the third category (*embedded methods*) select features and construct simultaneously 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 (Guyon et al., 2006).

Recursive Feature Elimination (RFE-SVM) (Guyon et al., 2006) is one pop-

ular 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 (Algorithm 1), based on the value of the weight vector $\mathbf{w}$.

Algorithm 1 Recursive Feature Elimination SVM - linear case

1. **repeat**
2. $\mathbf{w} \leftarrow$ SVM Training (primal formulation).
3. Eliminate feature p with smallest value of $|w_p|$.
4. **until** s variables remain.

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 the respective authors usually remove half of the remaining variables in each step (Guyon et al., 2006).

Embedded feature election can also be seen as an optimization problem. This is generally done by enforcing feature selection into the model, considering a sparsity term in the objective function. One example is the minimization of the "zero norm": $\Omega(\mathbf{w}) = \|\mathbf{w}\|_0 = |\{i : w_i \neq 0\}|$. Note that $\|\cdot\|_0$ is not a norm since the triangle inequality does not hold (Bradley and Mangasarian, 1998). Weston et al. (2003) 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. This method considers the following approximation of the l_0 norm.

$$\Omega(\mathbf{w}) = \sum_{j=1}^{n} log(\epsilon + |w_i|). \tag{6}$$

A mixed-integer program (MIP) has been proposed to iteratively select features for a non-linear SVM classifier (Mangasarian and Wild, 2007). 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 variables' values of the classifier and successive updates of the binary variables indicating presence or absence of the respective features.

In an early work (Iannarilli and Rubin, 2003) on "Feature Selection for Multi-class Discrimination via Mixed-Integer Linear Programming" a MILP for feature selection based on the assumption of feature independence has been introduced. Later, an alternative mixed-integer programming approach has been proposed for simultaneous feature selection and multi-class classification (Carrizosa et al., 2008). 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 (1995). A biobjective optimization scheme is considered to maximize fit while minimizing the total feature costs simultaneously, leading to an approximation to the set of Pareto-optimal classifiers.

Our work differs from previous ones since we extend the idea of feature cost minimization by considering a budget constraint and solving the mixed-integer formulation directly, instead of a dealing with a multi-objective approach.

4 Proposed SVM-based MILP Formulations

In each one of the following two subsections we propose a model based on previously introduced SVM formulations. In both cases, the main idea is to perform 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 $\mathbf{c} \in \Re^{\mathbf{n}}$, where c_j is the cost of acquiring attribute $j, j = 1, ..., n$. 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 consists in the respective norm used for the SVM formulation as will be shown next.

4.1 Proposed MILP Formulations based on l_1 Support Vector Machines

The following proposal emulates the l_1-SVM formulation described in Section 2.2. Instead of minimizing the l_1 norm of $\mathbf{w}$, represented by $\sum_{j=1}^{n} v_j$, we limit the selected features using a budget constraint, and force each weight w_j to belong to a given interval $[l_j, u_j]$, if the attribute is selected ($v_j = 1$).

$$
\begin{aligned}
\min_{\mathbf{w},\mathbf{v},b,\boldsymbol{\xi}} \quad & \sum_{i=1}^{m} \xi_i \\
\text{s.t.} \quad & y_i \cdot (\mathbf{w}^\top \cdot \mathbf{x}_i + b) \geq 1 - \xi_i, \quad i = 1, \ldots, m, \\
& l_j v_j \leq w_j \leq u_j v_j, \qquad j = 1, \ldots, n. \\
& \sum_{j=1}^{n} c_j v_j \leq B. \\
& v_j \in \{0, 1\}, \qquad j = 1, \ldots, n. \\
& \xi_i \geq 0, \qquad i = 1, \ldots, m.
\end{aligned}
\tag{7}
$$

The previous formulation presents interesting properties. For instance, we explicitly define a budget B, which represents the number of features in the classifier when all c_j are equal to 1. Additionally, the budget constraint allows

to incorporate 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 (7) consists in a Branch-and-Cut method whose efficiency greatly depends 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.

4.2 Proposed MILP Formulations based on LP-Support Vector Machines

The second formulation we propose extends the ideas of LP-SVM (see 2.3) to Mixed Integer Programming.

$$
\begin{aligned}
\min_{\mathbf{w},r,b,\boldsymbol{\xi}} \quad & -r + C\sum_{i=1}^{m}\xi_i \\
\text{s.t.} \quad & y_i \cdot (\mathbf{w}^\top \cdot \mathbf{x}_i + b) \geq r - \xi_i, \qquad i = 1,\ldots,m, \\
& -v_j \leq w_j \leq v_j, \qquad j = 1,\ldots,n. \\
& \sum_{j=1}^{n} c_j v_j \leq B, \\
& \xi_i \geq 0, \qquad i = 1,\ldots,m. \\
& v_j \in \{0,1\}, \qquad j = 1,\ldots,n. \\
& r_{lo} \leq r \leq r_{up},
\end{aligned}
\tag{8}
$$

This formulation does not require the determination of lower and upper bounds (i.e. l_j, u_j) for the coefficients w_j which in model (8) only can take values from the interval $[-1, 1]$. The desired flexibility for coefficients w_j is achieved indirectly by using the non-negative variable r which explicitly considers the structural risk minimization principle. As a consequence, model (8) requires the additional regularization parameter C.

As we will show next, the final constraint in model (8) is required in order to avoid that r becomes zero or diverges. Similar to model LP-SVM (see Section 2.3), for model (8) the solution with all variables equal to zero is feasible. As has been mentioned already, this can be avoided by introducing a lower bound for variable r. In our experiments we use the value $r_{lo} = 0.001$.

On the other hand, we introduce an upper bound (r_{up}) for variable r to avoid divergence. Different values for r_{up} are studied using line search.

5 Experimental Results

In this section we apply the classification models l_2-Support Vector Machines (Formulation (1)) and LP-Support Vector Machines (Formulation (4)) as well as the feature selection approaches l_1-Support Vector Machines (Formulation (3)), l_0-Support Vector Machines, and the two benchmark techniques for feature selection (Fischer+SVM and RFE-SVM) in comparison with the proposed formulations for simultaneous feature selection and classification via Mixed-Integer Linear Programming (Eq. (7) -MILP1- and Eq. (8) -MILP2-) to different datasets. These datasets will be presented in Section 5.1. Then we will describe our model selection procedure. Section 5.3 shows the results we obtained. The different parameters' influence on robustness and stability will be studied in Section 5.4. Finally, we analyze running times for all methods used in our experiments in Section 5.5.

5.1 Datasets

We applied the proposed approaches on six well-known datasets from the UCI Repository (Asuncion and Newman, 2007). These datasets have already been used for benchmark studies regarding the performance of Support Vector Machines (see e.g. (Ali and Smith-Miles, 2006; Song et al., 2012)).

- **Australian Credit (AUS)**: This dataset contains 690 granted loans; 383 good payers and 307 bad payers in terms of repayment, described by 14 variables.
- **Wisconsin Breast Cancer (WBC)**: This dataset contains 569 observations from 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 1,000 granted loans; 700 good payers and 300 bad payers in terms of repayment, described by 8 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 1,000 randomly selected examples (from the complete set of 3,190 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) (Alon et al., 1999): This dataset contains the expression of the 2,000 genes with highest minimal intensity across 62 samples (40 tumor and 22 normal). The following budget values were studied:
  $B \in \{10, 20, 50, 100, 250, 500, 1000, 2000\}$.

5.2 Model selection

The following model selection procedure was performed: training and test subsets were constructed using 10-fold cross-validation and the average accuracy (proportion of true results, Eq. (9)) and AUC were computed. For the microarray dataset we follow the procedure presented in (Victo Sudha George and Cyril Raj, 2011): training and test subsets are obtained using a leave-one-out procedure. Feature selection and classification is then performed on the training set and the classification performance is finally computed from test results.

For this work we studied the performance metrics "Accuracy" and "Area Under the Curve (AUC)" defined by one run (Eq. (10)), namely AUC, which is widely known as "Balanced Accuracy" (Sokolova et al., 2006). AUC can be described as the tradeoff between the benefits (TP_{rate}, or true positive rate) and costs (FP_{rate}, or false positive rate).

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}, \tag{9}$$

$$AUC = \frac{sensitivity + specificity}{2} \tag{10}$$

where TP=true positives, TN= true negatives, FP=false positives, FN= false negatives, sensitivity= $\frac{TP}{TP+FN}$ and specificity= $\frac{TN}{TN+FP}$. We performed a grid search to study the influence of parameter C for soft-margin models, and other model-specific parameters; see Section 5.4.

The intervals were further divided in homogenous values in order to characterize the relevant area for this parameter (where maximum predictive performance is reached). Additionally, the parameter B for the budget constraint was varied along all possible number of attributes. The values of the feature cost vector $\mathbf{c}$ are set to one. The optimization was performed using LIBSVM in the case of l_2-Support Vector Machines, LINPROG solver for Matlab in the case of l_1-Support Vector Machines (Formulation (3)), and CPLEX solver for the approaches LP-Support Vector Machines, MILP1, and MILP2.

5.3 Results

Tables 1 to 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 data set are presented in bold. In case of identical AUC, the solution with less variables is considered the best.

	Aus. Credit			W. 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 1: Best accuracy and AUC, in percentage, and number of selected features (k) for AUS, WBC, and PIMA 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 2: Best accuracy and AUC, in percentage, and number of selected features (k) for GC, IONO, and Splice datasets.

	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

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

From previous tables we observe that best predictive performance is achieved with the proposed models MILP1 and MILP2 in all seven cases. For Australian Credit, best AUC is achieved using MILP2 using 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 between 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 to all other methods. For the PIMA dataset, similar results are obtained with all approaches, and MILP2 performs slightly better with $k = 8$ (no feature selection). For the German Credit dataset, 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 considering 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 a remarkably better performance compared to alternative feature selection approaches, and MILP1 achieved a 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 Figures 1 to 7. The proposed approaches MILP1 and MILP2 are presented, together with the best alternative approach in terms of predictive AUC.

From Figure 1 we observe that, for Australian Credit data, all approaches behave very similar along all different subsets where the first attributes are the only relevant ones in this case. MILP2 slightly better for $k = 10$ and $k = 12$,

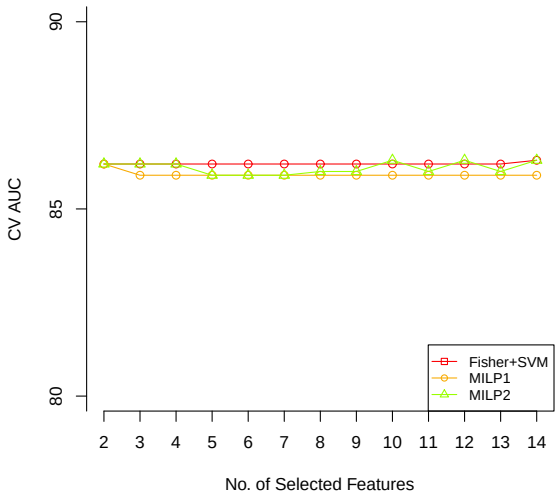

Figure 1: AUC versus the number of ranked variables for different feature selection approaches. Aus. Credit dataset.

while MILP1 has a slightly worse behavior overall.

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

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

For the German Credit dataset (Figure 4), 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 (Figure 5), best results are obtained with MILP1 using about half of the attributes, improving significantly the solution obtained with all features.

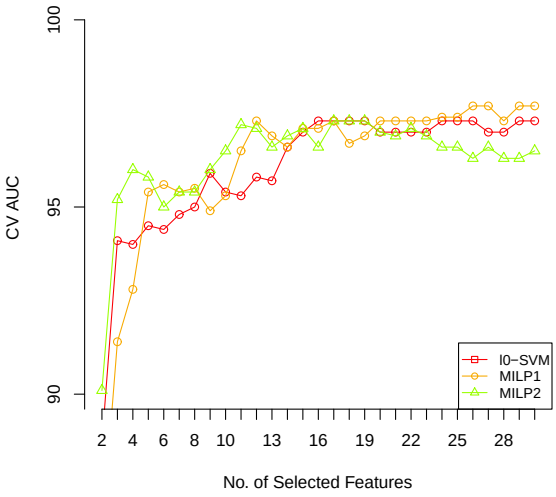

Figure 2: AUC versus the number of ranked variables for different feature selection approaches. W. Breast Cancer dataset.

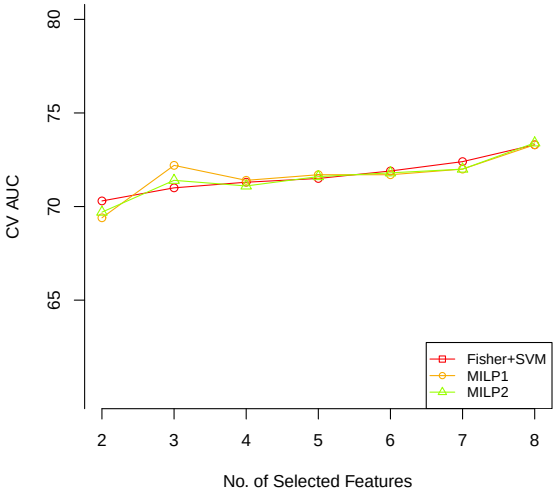

Figure 3: AUC versus the number of ranked variables for different feature selection approaches. PIMA Diabetes dataset.

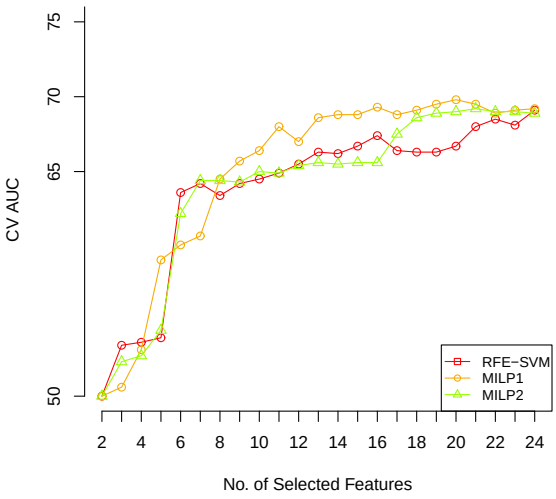

Figure 4: AUC versus the number of ranked variables for different feature selection approaches. German Credit dataset.

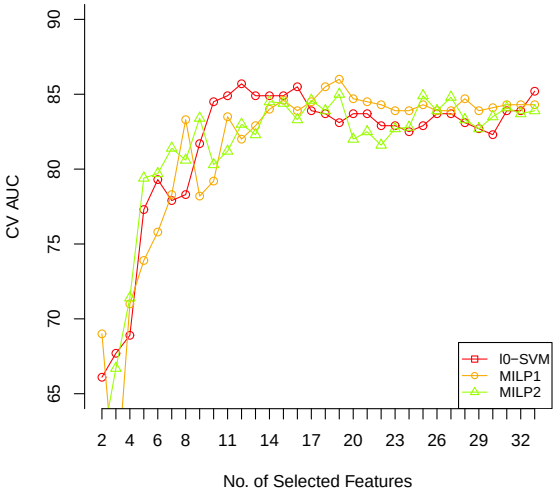

Figure 5: AUC versus the number of ranked variables for different feature selection approaches. Ionosphere dataset.

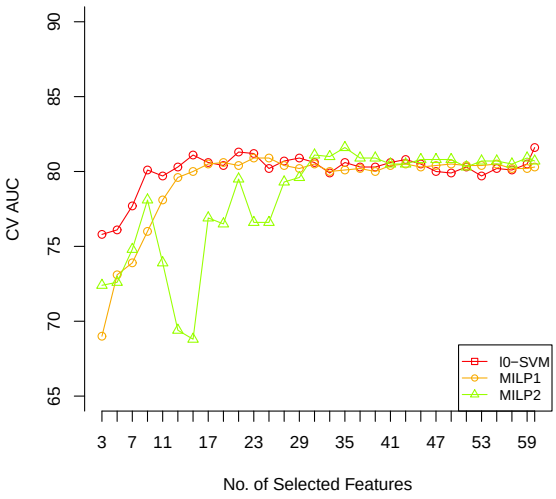

Figure 6: AUC versus the number of ranked variables for different feature selection approaches. Splice dataset.

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

Finally, for Colorectal microarray dataset (Figure 7), results are remarkably better using the proposed approaches with 50-100 variables instead of the entire set of 2,000 attributes, performing also better than the alternative approaches in this segment.

5.4 Influence of parameters

The proposed approaches consider different parameters that need to be studied in order to understand the robustness and stability of the respective methods. We vary parameter C and the upper bound for the margin variable r (r_{up}) for model MILP2 (Formulation (8)). For model MILP1 (Formulation (7)) we analyze the weight vectors' bounds (l_j and u_j).

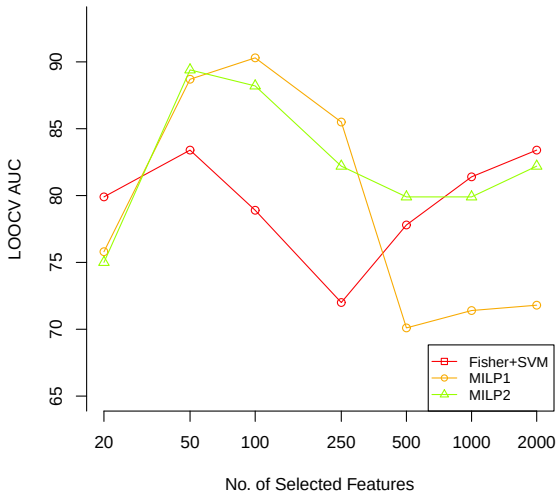

Figure 7: AUC versus the number of ranked variables for different feature selection approaches. Colorectal Microarray dataset.

Assuming $l_j = -u_j$ and $u_j = u$, $\forall j$, we use the following set of values for parameters u, C, and r_{up}, respectively:

$u \in \{1, 2, 4, 8, 16, 32, 64, 128, 256\}$

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

$r_{up} \in \{1, 2, 4, 8, 16, 32, 64, 128, 256\}$

Tables 5 to 4 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, respectively.

	AUS	WBC	PIMA	GC	IONO	SPLICE	CoMA
1	85,9	95,4	65,9	69,2	81,3	80,3	82,6
2	85,7	97,7	72,4	69,1	82,5	79,8	85,7
4	85,7	96,7	73,3	69,1	83,5	79,8	76,1
8	85,7	96,6	73,3	69,1	84,2	79,8	70,6
16	85,7	95,9	73,3	69,1	84,3	79,8	67,3
32	85,7	95,6	73,3	69,1	83,9	79,8	62,3
64	85,7	95,1	73,3	69,1	83,9	79,8	70,3
128	85,7	94,5	73,3	69,1	83,9	79,8	68,3
256	85,7	94,6	73,3	69,1	83,9	79,8	75,1

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

	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 C.

	AUS	WBC	PIMA	GC	IONO	SPLICE	CoMA
1	85,9	96,5	73,3	68,9	83,9	80,3	76,4
2	86,3	96,5	73,3	68,9	83,9	80,7	81,1
4	86,3	96,5	73,3	68,9	83,9	80,7	76,4
8	86,3	96,5	73,3	68,9	83,9	80,7	82,2
16	86,3	96,5	73,3	68,9	83,9	80,7	79,7
32	86,3	96,5	73,3	68,9	83,9	80,7	82,2
64	86,3	96,5	73,3	68,9	83,9	80,7	86,9
128	86,3	96,5	73,3	68,9	83,9	80,7	86,9
256	86,3	96,5	73,3	68,9	83,9	80,7	86,9

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

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 datasets. All parameters have similar influence on the methods' predictive performance.

5.5 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 the proposed method (one fold using 10-fold cross-validation or Leave-One-Out in the case of Microarray datasets). The mean running time (in seconds) is obtained by averaging all running times for different folds (and for different budgets for the proposed approaches).

	AUS	WBC	PIMA	GC	IONO	SPLICE	CoMA
l_2-SVM	0,5	0,3	0,3	0,5	0,3	0,7	0,8
LP-SVM	0,2	0,1	0,1	0,3	0,1	0,4	0,4
l_1-SVM	0,4	0,4	0,3	0,4	0,3	4,8	0,6
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
MILP1-FS	0,2	0,2	0,2	0,3	0,2	1,7	2,2
MILP2-FS	0,2	29,0	0,3	0,7	8,7	38,3	1,2

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

It is important to notice that all running times are tractable and reasonable. There are some cases when the proposed approaches become very slow, affecting the average times and as a consequence the comparison. In particular, when the budget is about half of the number of original variables, running times are usually higher, while the fastest experiments are obtained when the budget constraint is not activated and all features are used. In this last case, average running times are similar to those obtained by LP-SVM.

6 Conclusions

In this work we presented two embedded approaches for simultaneous feature selection and classification based on Mixed Integer 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 outperform other feature ranking techniques in terms of predictive performance for different SVM-based feature selection techniques, based on their ability to identify irrelevant attributes using the classifier.
- 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 is maintained.

In contrast, for microarray data, and in particular for colorectal microarray, our approaches led to an important 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 proposed models resulted 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 to the weight vector into a kernel-based formulation. Secondly, although in this work all attributes are treated equally, the proposed approach has the potential to incorporate 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 constraints or minimizes acquisition costs while classifying adequately.

Acknowledgments

Support from the Institute of Complex Engineering Systems (ICM: P-05-004-F, CONICYT: FBO16) (`www.isci.cl`) is greatly acknowledged. The first author was supported by FONDECYT project 11121196.

References

S. Ali and K. A. Smith-Miles. On learning algorithm selection for classification. *Applied Soft Computing*, 6:119–138, 2006.

U. Alon, N. Barkai, D. A. Notterman, K. Gish, S. Ybarra, D. Mack, and Levine A. J. 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, 6745-6750*, 1999.

A. Asuncion and D.J. Newman. UCI machine learning repository, 2007. URL `http://archive.ics.uci.edu/ml/`.

P. Bradley and 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*, 1998.

E. Carrizosa, B. Martín-Barragán, and Romero-Morales D. Multi-group support vector machines with measurement costs: A biobjective approach. *Discrete Applied Mathemathics*, 156:950–966, 2008.

E. Carrizosa, B. Martín-Barragán, and Romero-Morales D. Detecting relevant variables and interactions in supervised classification. *European Journal of Operational Research*, 213(1):260–269, 2011.

I. Guyon, S. Gunn, M. Nikravesh, and L. A. Zadeh. *Feature extraction, foundations and applications*. Springer, Berlin, 2006.

Rohayanti Hassan, Razib M. Othman, Puteh Saad, and Shahreen Kasim. A compact hybrid feature vector for an accurate secondary structure prediction. *Information Sciences*, 181(23):5267–5277, 2011.

Frank J. Iannarilli and Paul A. Rubin. Feature selection for multiclass discrimination via mixed-integer linear programming. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(6):779–783, 2003.

J. Kittler. *Pattern Recognition and Signal Processing*, chapter Feature Set Search Algorithms, pages 41–60. Sijthoff and Noordhoff, Netherlands, 1978.

S. Maldonado, R. Weber, and J. Basak. Kernel-penalized SVM for feature selection. *Information Sciences*, 181(1):115–128, 2011.

Olvi L. Mangasarian and Edward W. Wild. Feature selection for nonlinear kernel support vector machines. In *Seventh IEEE International Conference on Data Mining*, pages 231–236, Omaha, NE, October 28-31 2007. IEEE.

M. Sokolova, N. Japkowicz, and Szpakowicz S. Beyond accuracy, f-score and roc: A family of discriminant measures for performance evaluation. In *Advances in Artificial Intelligence*. Springer, Berlin Heidelberg, 1015-1021, 2006.

L. Song, A. Smola, A. Gretton, J. Bedo, and K. Borgwardt. Feature selection via dependence maximization. *Journal of Machine Learning Research*, 13: 1393–1434, 2012.

P.D. Turney. Cost-sensitive classification: Empirical evaluation of a hybrid genetic decision tree induction algorithm. *Journal of Artificial Intelligence Research*, 2:369–409, 1995.

Ö. Uncu and I. B. Türksen. A novel feature selection approach: Combining feature wrappers and filters. *Information Sciences*, 177(2):449–466, 2007.

V. Vapnik. *Statistical Learning Theory*. John Wiley and Sons, 1998.

G. Victo Sudha George and V. Cyril Raj. Review on feature selection techniques and the impact of svm for cancer classification using gene expression profile. *International Journal of Computer Science and Engineering Survey*, 2(3):16–27, 2011.

J. Weston, A.. Elisseeff, B. Schölkopf, and M. Tipping. The use of zero-norm with linear models and kernel methods. *Journal of Machine Learning Research*, 3:1439–1461, 2003.

Hwanjo Yu, Jinha Kim, Youngdae Kim, Seungwon Hwang, and Young Ho Lee. An efficient method for learning nonlinear ranking SVM functions. *Information Sciences*, 209:37–48, 2012.

W. Zhou, L. Zhang, and L. Jiao. Linear programming support vector machines. *Pattern Recognition*, 35:2927–2936, 2002.