Information Sciences 286 (2014) 228–246

Contents lists available at ScienceDirect

Information Sciences

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

Feature selection for high-dimensional class-imbalanced data sets using Support Vector Machines

Sebastián Maldonado [a,*], Richard Weber [b], Fazel Famili [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] National Research Council of Canada, Ottawa, Canada

ARTICLE INFO

Article history:
Received 31 May 2013
Received in revised form 27 May 2014
Accepted 6 July 2014
Available online 30 July 2014

Keywords:
Feature selection
Imbalanced data set
Dimensionality reduction
Support Vector Machine
Data mining

ABSTRACT

Feature selection and classification of imbalanced data sets are two of the most interesting machine learning challenges, attracting a growing attention from both, industry and academia. Feature selection addresses the dimensionality reduction problem by determining a subset of available features to build a good model for classification or prediction, while the class-imbalance problem arises when the class distribution is too skewed. Both issues have been independently studied in the literature, and a plethora of methods to address high dimensionality as well as class-imbalance has been proposed. The aim of this work is to simultaneously explore both issues, proposing a family of methods that select those attributes that are relevant for the identification of the target class in binary classification. We propose a backward elimination approach based on successive holdout steps, whose contribution measure is based on a balanced loss function obtained on an independent subset. Our experiments are based on six highly imbalanced microarray data sets, comparing our methods with well-known feature selection techniques, and obtaining a better prediction with consistently fewer relevant features.

1. Introduction

Feature selection is an important topic in data mining, especially in high-dimensional applications. A low-dimensional representation of the data reduces the risk of *overfitting*, which is higher in this kind of data sets [14,20], improving the model's generalization ability. Feature selection is a combinatorial problem in the number of original features [14], and finding the optimal subset of variables is considered NP-hard.[1]

Feature selection can be very helpful when facing imbalanced data sets [8]. In the context of classification, this problem occurs when there are many more examples from some classes than from others. In this paper we focus on binary classification, and we refer to the class imbalance problem when one of the two classes (the negative class) is significantly larger in terms of instances than the other, the positive class. Furthermore, we assume that misclassifying an instance from the positive class (also called target class) is more expensive than misclassifying an instances from the negative class [16]. This problem is especially challenging when both classes have a high degree of overlap as has been pointed out e.g. by Prati et al.

* Corresponding author. Tel.: +56 2 26181874.
E-mail addresses: smaldonado@uandes.cl (S. Maldonado), rweber@dii.uchile.cl (R. Weber), Fazel.Famili@nrc-cnrc.gc.ca (F. Famili).

[1] NP-hard is the class of problems that are non-deterministic polynomial-time hard; see e.g. [23].

http://dx.doi.org/10.1016/j.ins.2014.07.015
0020-0255/

[25] and Qu et al. [26]. It is prevalent in many applications, including fraud/churn detection [19], text categorization [43], medical diagnosis [35], detection of software defects [24], and many others.

Technically speaking, any data set with an unequal distribution between the two classes, can be considered imbalanced. However, class ratios of 5:1 (majority class:minority class) or higher have often been considered in experiments as imbalanced data sets ([16]).

Support Vector Machines (SVMs) [36] are effective classifiers that provide several advantages such as adequate generalization of new objects and a representation that depends on few parameters. Additionally, it can be shown that the respective objective function (see Section 3.1) is convex which assures that the method does not get stuck in local minima of the particular model [36]. However, SVMs do not determine the features' importance within the respective model [20]. In the present paper, we introduce a new feature selection approach for binary classification using SVM which is especially suited for imbalanced data sets.

Section 2 provides an overview on the class-imbalance problem. In Section 3 we briefly introduce SVM for classification as well as for feature selection lying the basis for the posterior developments. Section 4 introduces the proposed family of methods for feature selection based on SVM. Experimental results using imbalanced real-world data sets are presented in Section 5. Section 6 summarizes this paper, provides its main conclusions, and hints at future developments.

2. The class imbalance problem

Several solutions to handle the problem of classifying imbalanced data sets have been proposed. These are mainly in four directions: resampling, cost-sensitive learning, one class learning, and feature selection. In the following subsections we briefly describe the first three of these directions. We then refer to the assessment metrics used to evaluate the respective classifiers. Related work on feature selection to classify imbalanced data sets constitutes the basis for our present work and will be detailed in Section 3.2.

2.1. Resampling

There are several techniques regarding data resampling, which differ mainly in their nature (random or informed resampling). The two most common data resampling techniques are *random oversampling* and *random undersampling*. Random oversampling duplicates randomly selected examples of the minority class. While this does help to balance the class distribution, no new information is added to the data set thus leading potentially to overfitting [35]. Also, this procedure increases the size of the training set, causing longer model training times. Random undersampling randomly discards instances from the majority class, which may lead to an important loss of information [35].

Chawla et al. [9] proposed SMOTE, a "Synthetic Minority Over-sampling TEchnique", which generates new examples for the minority class. These are created artificially by interpolating the preexisting minority instances, which may help to improve the classification performance on imbalanced data sets [19].

2.2. Cost-sensitive learning

Cost-sensitive techniques are based on the concept of a cost matrix, which can be considered as a numerical representation of the penalty when classifying instances to the wrong class. For example, we define C_- as the cost of misclassifying a majority class instance as a minority class instance and let C_+ represent the cost of the contrary case. Typically, there is no cost associated with correct classification and the cost of misclassification in the target class is higher than the contrary case, i.e., $C_+ > C_-$. The objective of cost-sensitive learning then is to develop a classifier that minimizes the overall cost on the training data set.

There are different ways of implementing cost-sensitive learning. A detailed analysis would go beyond the scope of this paper since we concentrate on a methodological development independent of a particular application where the respective misclassification costs could be specified. The interested reader is referred to [16,33].

2.3. One-class learning

When negative examples greatly outnumber the positive ones, most classifiers tend to overfit [8]. This is particulary true when facing high-dimensional data sets [35]. In this case, one-class SVM trained only with the target class may lead to a better predictive performance [34]. Here, the method attempts to measure the similarity between a query object and the target class, where classification is accomplished by imposing a threshold on the similarity value [8]. It has been shown [34] that the one-class approach to classify high-dimensional imbalanced data sets can be an interesting alternative to feature selection in such situations.

2.4. Assessment metrics for imbalanced classification

Traditionally, the most frequently used metric for binary classification is the *accuracy*, which is the proportion of true results to all results. This metric provides a simple way of describing the classification performance in a given data set.

However, it is not appropriate for classification of imbalanced data sets [16]. For example, if a given data set includes 5% of target class instances and 95% of majority examples, a naïve approach of classifying every instance to the majority class would provide an accuracy of 95%, which could be considered very good. This measure, however, fails to reflect the fact that 0% of the target instances are identified, which we assume with higher misclassification cost [16].

Alternatively, several assessment metrics are frequently adopted in the research community for learning problems with imbalanced data sets. In this work we use the *G-mean* as the main performance metric. This measure is computed as the geometric mean of the true positive and true negative rates:

$$G-mean=\sqrt{\frac{TP}{TP+FN}*\frac{TN}{TN+FP}} \tag{1}$$

where *TP* = true positives, *TN* = true negatives, *FP* = false positives, and *FN* = false negatives. Another metric commonly used to assess classification models is the Area under the curve (AUC), see [30].

3. Related work on classification and feature selection with Support Vector Machines

This section lays the foundation for the posterior developments by first recalling the SVM approach for binary classification. The subsequent subsection presents related work for feature selection with SVM.

3.1. Binary classification with Support Vector Machines

Given training vectors $\mathbf{x}_i \in \Re^n, i=1,\ldots,m$ and a vector of binary class labels $\mathbf{y} \in \Re^m, y_i \in \{-1,+1\}, i=1,\ldots,m$, linear SVM determine the optimal hyperplane $f(\mathbf{x})=\mathbf{w}^T\cdot\mathbf{x}+b$ that aims to separate the training patterns according to their classes while achieving best generalization performance for new instances. In order to obtain this optimal hyperplane, SVM maximize its *margin*, which is the sum of the distances to the closest positive and negative training patterns, and is equivalent to minimizing the norm of $\mathbf{w}$ [36]. Since a perfect separation of all training patterns is not always possible, a slack variable ξ_i is introduced for each training vector $\mathbf{x}_i, i=1,\ldots,m$ and C is a parameter that penalizes the overall training error [36] as shown in (2).

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

In binary classification of imbalanced data sets the case of primary interest is the one where the target (positive) class, labeled $+1$, is much smaller than the background (negative) class, labeled -1.

To achieve a nonlinear classifier based on SVM, the solution will be given by a kernel machine

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

where training data are mapped to the higher dimensional space $\mathscr{H}$ by the function $\phi:\mathbf{x}\rightarrow\phi(\mathbf{x})\in\mathscr{H}$. The mapping can be represented by a kernel function $K(\mathbf{x},\mathbf{y})=\phi(\mathbf{x})\cdot\phi(\mathbf{y})$ which defines an inner product in $\mathscr{H}$ and should satisfy Mercer's condition [36]. The optimal separating hyperplane in $\mathscr{H}$ is the hyperplane with maximal distance to the closest image $\phi(\mathbf{x}_i)$ from the respective training pattern. The corresponding dual formulation can be stated as follows:

$$\begin{aligned}\max_{\boldsymbol{\alpha}}\quad & \sum_{i=1}^{m}\alpha_i-\frac{1}{2}\sum_{i,s=1}^{m}\alpha_i\alpha_s y_i y_s K(\mathbf{x}_i,\mathbf{x}_s)\\ \text{s.t.}\quad & \sum_{i=1}^{m}\alpha_i y_i=0,\\ & 0\leqslant\alpha_i\leqslant C,\quad i=1,\ldots,m.\end{aligned} \tag{4}$$

We base our analysis on the Gaussian kernel, which has led to the best results in many applications and is a common choice in the literature [20]. It has the following form.

$$K(\mathbf{x}_i,\mathbf{x}_s)=\exp\left(-\frac{||\mathbf{x}_i-\mathbf{x}_s||^2}{2\sigma^2}\right) \tag{5}$$

where $\sigma>0$ is the parameter controlling the kernel width.

3.2. Related work on feature selection with SVM

According to [14], three main approaches have been developed for feature selection: filter, wrapper, and embedded methods. The first approach (*filter methods*) uses statistical properties of the features in order to filter out poorly informative

ones, before applying any classification algorithm. Commonly used filter methods are the χ^2 statistic, which measures the independence between the distribution of values and classes [35]; the Information Gain, which is a measure of informational entropy, to the problem of deciding how important a given feature is [35], and the Fisher Criterion Score (F), which computes each feature's importance independently of the other features by calculating the difference of that feature's mean values for the two classes [14], as shown in (6).

$$F(j) = \left| \frac{\mu_j^+ - \mu_j^-}{(\sigma_j^+)^2 + (\sigma_j^-)^2} \right| \tag{6}$$

where μ_j^+ (μ_j^-) represent the mean for the j-th feature in the positive (negative) class and σ_j^+ (σ_j^-) is the respective standard deviation ($j = 1, \ldots, n$).

Many successful applications of this measure have been reported in literature; see e.g. [11] where based on the Fisher score a multi-class SVM has been proposed to classify face images. In [39] this score has been used successfully for text classification.

Wrapper methods explore the whole set of features to score feature subsets according to their predictive power, which is computationally demanding, but often provides more accurate results than filter methods. Common wrapper strategies are Sequential Forward Selection (SFS) and Sequential Backward Elimination (SBE) [14] which evaluate the selected feature sets based on a particular classification method. In the first case, starting without any variable, the method tries out the variables one by one and includes in each iteration the most relevant of the remaining ones. On the other hand, SBE starts with all candidate features and tests them one by one for statistical significance. In each iteration the least significant feature is eliminated.

The third class of feature selection techniques (*embedded methods*) determines the respective feature subset during classifier construction, which can be seen as a search in the combined space of feature subsets and hypotheses. Similar to wrapper methods, embedded approaches are specific to a given classification method and therefore have the advantage to include the interaction with the classifier when modeling feature dependencies. They are, however, computationally less intensive than wrapper methods [14].

One popular method, which is relevant for the remainder of this paper, is known as Recursive Feature Elimination (RFE-SVM) [15]. The goal of the linear version of this approach is to find a subset of size r among n variables ($r < n$), eliminating those features whose removal lead to the largest margin of class separation. This can be achieved using backward elimination, based on the components of the weight vector $\mathbf{w}$.

The linear RFE-SVM approach presented in [15] can be also generalized to the nonlinear case which will be presented next and is the basis for our subsequent development. Since the margin of the separating hyperplane is inversely proportional to the Euclidean norm of the weight vector $\mathbf{w}$, this norm can be rewritten in terms of the dual variables of the SVM model:

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

The feature to be removed in each iteration is the one whose removal minimizes the variation of $W^2(\boldsymbol{\alpha})$. The nonlinear RFE-SVM algorithm follows, where $W^2_{(-p)}(\boldsymbol{\alpha})$ represents the training object i with feature p removed.

Algorithm 1. Recursive Feature Elimination SVM - nonlinear case

1. **repeat**
2. $\quad \mathbf{w} \leftarrow$ SVM Training (dual formulation).
3. $\quad$ Eliminate feature p with smallest value of $|W^2(\boldsymbol{\alpha}) - W^2_{(-p)}(\boldsymbol{\alpha})|$.
4. **until** r variables remain.

While one could choose a single variable to be removed in each iteration, this would be inefficient in many high-dimensional applications (e.g. classifying microarray data) where data sets are often characterized by thousands of features, and the authors usually remove half of the variables in each step [15]. Successful applications of RFE-SVM e.g. for analyzing EEG-signals have been reported in [17] where the authors confirm that this method could provide accurate and useful information to aid medical diagnosis.

Embedded feature selection can also be seen as an optimization problem. This is generally done through enforcing feature selection on the model parameters directly, considering a sparsity term in the objective function. For instance, the Euclidean norm from the classical SVM (Formulation (2)) can be replaced by the l_1 norm $\Omega(\mathbf{w}) = \sum_i |w_i|$, as presented in the approach l_1 Support Vector Machine (l_1-SVM) by Bradley and Mangasarian [6]:

$$\begin{aligned} \min_{\mathbf{w},b,\xi} \quad & \sum_{j=1}^{n} |w_j| + C \sum_{i=1}^{m} \xi_i \\ \text{s.t.} \quad & y_i \cdot (\mathbf{w}^T \cdot \mathbf{x}_i + b) \geqslant 1 - \xi_i, \quad i = 1, \ldots, m, \\ & \xi_i \geqslant 0, \quad i = 1, \ldots, m. \end{aligned} \tag{8}$$

An alternative sparsity term is the minimization of the "zero norm": $\Omega(\mathbf{w}) = \|\mathbf{w}\|_0 = |\{i : w_i \neq 0\}|$. Note that, unlike the l_1 norm, $\|\cdot\|_0$ is not a norm because the triangle inequality does not hold [6]. Weston [41] 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 $\mathbf{w}$ obtained from the SVM formulation, until convergence. 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:

$$\Omega(\mathbf{w}) = \sum_{j=1}^{n} log(\epsilon + |w_j|). \tag{9}$$

where ϵ is a small positive constant.

Filter and resampling methods have been used together with positive results in imbalanced data sets [5,29,35,40,43]. Filter techniques do not require further adaptations when facing imbalanced data sets and are compatible to data resampling since the process of feature filtering is independent of the learning process. Some specific filter approaches have been proposed to address the class-imbalance problem directly: FAST [10] performs feature filtering based on the Area under the curve (AUC), and DBFS [1], which considers Information Gain as an alternative for AUC.

To the best of our knowledge, neither wrapper nor embedded methods have been proposed to treat the class imbalance problem in high-dimensional applications using Support Vector Machines. Fernandez et al. [13] applied a genetic algorithm to improve the performance of a fuzzy rule-based classification system for imbalanced data sets, but without selecting relevant features. Villar et al. [37] also proposed a genetic algorithm that, in their case, simultaneously performs feature selection and granularity learning for fuzzy rule-based classification systems for imbalanced data sets. This approach removes irrelevant variables through a backward elimination process using an AUC-based fitness function. This method achieved good results under imbalanced class conditions, but only for low-dimensional data sets.

4. Proposed feature selection methods for imbalanced data sets

We propose a family of embedded methods for backward feature selection using Support Vector Machines which are inspired by the backward elimination procedure of SVM-RFE [15], as presented in Section 3.2. The rationale behind our approach is that we eliminate those features whose removal has less impact on the final solution, considering a class-imbalanced problem. To perform this, we attempt at recreating the main goal for this task: to achieve the best predictive performance in an unseen subset, using a cost-sensitive metric.

Section 4.1 presents our base algorithm for backward feature elimination. The holdout scheme proposed in Section 4.2 shows how an advanced split of training sets can improve the respective results derived from the base algorithm. Section 4.3 illustrates how SMOTE can be used within the two previously presented methods in order to treat very highly imbalanced data sets.

4.1. Base algorithm for backward elimination

In this subsection we introduce our base algorithm and study different variations, for example by using different loss functions. Following the notation used by Song et al. [31], we denote by $\mathcal{S}$ the full set of features. We want to find a subset $\mathcal{K}$ ($\mathcal{K} \subseteq \mathcal{S}$) of features, such that the performance of the SVM classifier using this subset's features is maximized, considering a training set $\mathcal{T}$.

In order to compare the performance of different feature selection strategies, we will generate a vector of features $\mathcal{S}^\dagger$ sorted in increasing degree of relevance. This is achieved by the following iterative algorithm:

Algorithm 2. Backward Algorithm for Feature Elimination (BFE-SVM)

Input: The original set of features $\mathcal{S}$
Output: An ordered vector of features $\mathcal{S}^\dagger$
1. $\mathcal{S}^\dagger \leftarrow \emptyset$
2. **repeat**
3. $\quad \boldsymbol{\alpha} \leftarrow$ SVM Training using$\mathcal{T}$
4. $\quad \mathcal{I} \leftarrow \text{argmin}_{\mathcal{I}} \sum_{j \in \mathcal{I}} \text{LOSS}(\boldsymbol{\alpha}, \mathcal{S} \setminus \{j\}),\ \mathcal{I} \subset \mathcal{S}$
5. $\quad \mathcal{S} \leftarrow \mathcal{S} \setminus \mathcal{I}$
6. $\quad \mathcal{S}^\dagger \leftarrow (\mathcal{S}^\dagger, \mathcal{I})$
7. **until** $\mathcal{S} = \emptyset$

In Step 4 the algorithm determines a set $\mathcal{I}$ of features to be eliminated. While one could choose a single element of $\mathcal{S}$, this would be inefficient when there is a large number of irrelevant features. On the other hand, removing too many features at once increases the risk to loose relevant features [15]. In our experiments, we found a good compromise between speed and feature quality was to remove 10% of the current features at every iteration.

Since our proposed embedded approach takes explicitly into account feature performance within a set of remaining features at any iteration, the problems typically related to model-free feature screening, do not occur. These problems are (1) any irrelevant feature that is highly correlated with the set of relevant features could be selected and (2) a marginally uncorrelated feature that is jointly correlated with the response might not be selected; see [2,12] for a more general discussion of feature screening.

Different loss functions can be considered within Algorithm 2. In our experiments we used the following ones:

- **Standard 0–1 loss function:** This measure is based on the total number of errors in the training set. It assumes equal cost for both errors and therefore it is not suitable for imbalanced data sets. Using this loss function we want to prove that feature relevance should be measured considering different costs for different types of errors. Formally, given a solution $(\boldsymbol{\alpha}, b)$, obtained by applying SVM to a training set $\mathcal{T}$, and a set of available features $\mathcal{S}$, the *0–1 loss* for a given feature $j \in \mathcal{S}$ is defined as:

$$\text{LOSS}_{0-1}((\boldsymbol{\alpha}, b), \mathcal{S} \setminus \{j\}, \mathcal{T}) = \sum_{s \in \mathcal{T}} \left| y_s - sgn\left(\sum_{i \in \mathcal{T}} \alpha_i y_i K(\mathbf{x}_i^{(-j)}, \mathbf{x}_s^{(-j)}) + b \right) \right| \tag{10}$$

where $\mathbf{x}_i^{(-j)}$ is the training object i with feature j removed and *sgn* is the signum function.

- **Balanced loss function:** In order to make the errors from an imbalanced data set comparable, we use the balanced loss function, which takes the weighted average of Type I and Type II errors. We split the training set $\mathcal{T}$ into $\mathcal{T}^+$ and $\mathcal{T}^-$, containing the positive and negative instances, respectively. The *balanced loss* for a given feature $j \in \mathcal{S}$, is defined as:

$$\begin{aligned} \text{LOSS}_{bl}((\boldsymbol{\alpha}, b), \mathcal{S} \setminus \{j\}, \mathcal{T}) = & \frac{\sum_{s \in \mathcal{T}^-} \left| y_s - sgn\left(\sum_{i \in \mathcal{T}^-} \alpha_i y_i K(\mathbf{x}_i^{(-j)}, \mathbf{x}_s^{(-j)}) + b \right) \right|}{|T^-|} \\ & + \frac{\sum_{s \in \mathcal{T}^+} \left| y_s - sgn\left(\sum_{i \in \mathcal{T}^+} \alpha_i y_i K(\mathbf{x}_i^{(-j)}, \mathbf{x}_s^{(-j)}) + b \right) \right|}{|T^+|} \end{aligned} \tag{11}$$

where $|T^+|$ ($|T^-|$) represents the cardinality of the training set T^+ (T^-), i. e. the number of positive (negative) instances.

- **Predefined loss function**: A predefined loss function can be used if the relation between the costs of Type I and Type II errors can be estimated for a given application. Since the scope of this work is mainly methodological, we will not consider this measure, but it can be useful in particular applications with imbalanced data sets, such as churn prediction or credit scoring.

4.2. Holdout strategy for backward feature elimination

We propose an additional modification to the backward elimination process, which is the assessment of each attribute's contribution in an "unseen" subset of the training data, instead of training the model and constructing the loss function with the same data set. To do so, the training data $\mathcal{T}$ is further splitted into a subset $\mathcal{TR}$ where SVM is trained, and a validation set $\mathcal{V}$ used to compute the respective loss function [20]. This is achieved by adding a holdout strategy to Algorithm 2, leading to the following algorithm.

Algorithm 3. Holdout Algorithm for Backward Feature Elimination (HO-BFE)

Input: The original set of features $\mathcal{S}$
Output: An ordered vector of features $\mathcal{S}^\dagger$
1. $\mathcal{S}^\dagger \leftarrow \emptyset$
2. **repeat**
3. $(\mathcal{TR}, \mathcal{V}) \leftarrow$ Holdout using $\mathcal{T}$
4. $\boldsymbol{\alpha} \leftarrow$ SVM Training using $\mathcal{TR}$
5. $\mathcal{I} \leftarrow \text{argmin}_{\mathcal{I}} \sum_{j \in \mathcal{I}}$ LOSS $(\boldsymbol{\alpha}, \mathcal{S} \setminus \{j\}, \mathcal{TR}, \mathcal{V}),\ \mathcal{I} \subset \mathcal{S}$
6. $\mathcal{S} \leftarrow \mathcal{S} \setminus \mathcal{I}$
7. $\mathcal{S}^\dagger \leftarrow (\mathcal{S}^\dagger, \mathcal{I})$
8. **until** $\mathcal{S} = \emptyset$

We redefine the loss measures used in Algorithm 3:

$$\text{LOSS}_{0-1}((\boldsymbol{\alpha}, b), \mathcal{S} \setminus \{j\}, \mathcal{TR}, \mathcal{V}) = \sum_{l \in \mathcal{V}} \left| y_l^v - sgn\left(\sum_{i \in \mathcal{TR}} \alpha_i y_i K(\mathbf{x}_i^{(-j)}, \mathbf{x}_l^{v(-j)}) + b \right) \right| \tag{12}$$

$$\mathrm{LOSS}_{bl}((\boldsymbol{\alpha}, b), \mathcal{S} \setminus \{j\}, \mathcal{TR}, \mathcal{V}) = \frac{\sum_{l \in \mathcal{V}^-} \left| y_l^v - sgn\left(\sum_{i \in \mathcal{TR}^-} \alpha_i y_i K(\mathbf{x}_i^{(-j)}, \mathbf{x}_l^{v(-j)}) + b\right)\right|}{|T^-|} + \frac{\sum_{l \in \mathcal{V}^+} \left| y_l^v - sgn\left(\sum_{i \in \mathcal{TR}^+} \alpha_i y_i K(\mathbf{x}_i^{(-j)}, \mathbf{x}_l^{v(-j)}) + b\right)\right|}{|T^+|} \tag{13}$$

where $\mathcal{V}$ is the Validation subset and $\mathbf{x}_l^v$ and y_l^v are this subset's objects and labels, respectively. $\mathbf{x}_l^{v(-j)}$ means validation object l with feature j removed.

One of the problems when applying this algorithm is that different runs may lead to different results in terms of selected features, in particular with few training examples and in the presence of imbalanced data sets. A more stable selected feature subset can be achieved by performing k different holdout splits and averaging the loss functions before eliminating features. The attributes whose elimination leads to a better predictive performance according to this average will be added to the ordered list of features $\mathcal{S}^\dagger$ in Step 4. In our experiments we use $k = 5$, and the influence of this parameter is further discussed in Section 5.3.

4.3. Backward feature elimination with SMOTE

Although the previous algorithms are designed to perform an adequate feature selection under conditions of imbalanced data sets, backward embedded methods require a base classifier (in this case SVM) that successfully discriminates between the two classes. In cases of highly imbalanced and overlapping classes, data resampling may be useful to achieve an adequate classification performance. For the particular case of microarray data, undersampling may not be the right procedure since only few examples are available. In order to overcome this problem, we may generate artificial target class instances using the oversampling method SMOTE on the training set. This procedure allows a better split of the training data set and makes some of the extreme cases of imbalanced data sets tractable. We propose to use SMOTE for the base algorithm Backward Feature Elimination (Algorithm 2) as well as for the base algorithm with holdout strategy (Algorithm 3), completing our proposed family of methods for feature selection based on SVM for imbalanced data sets. The respective algorithms, i.e. Algorithms 4 and 5 follow.

Algorithm 4. Backward Algorithm for Feature Elimination using SMOTE

Input: The original set of features $\mathcal{S}$
Output: An ordered vector of features $\mathcal{S}^\dagger$
1. $\mathcal{S}^\dagger \leftarrow \emptyset$
2. $\mathcal{T}' \leftarrow \mathrm{SMOTE}(\mathcal{T})$
3. **repeat**
4. $\quad \boldsymbol{\alpha} \leftarrow$ SVM Training using $\mathcal{T}'$
5. $\quad \mathcal{I} \leftarrow \mathrm{argmin}_{\mathcal{I}} \sum_{j \in \mathcal{I}} \mathrm{LOSS}(\boldsymbol{\alpha}, \mathcal{S} \setminus \{j\}),\ \mathcal{I} \subset \mathcal{S}$
6. $\quad \mathcal{S} \leftarrow \mathcal{S} \setminus \mathcal{I}$
7. $\quad \mathcal{S}^\dagger \leftarrow (\mathcal{S}^\dagger, \mathcal{I})$
8. **until** $\mathcal{S} = \emptyset$

Algorithm 5. Holdout Algorithm for Backward Feature Elimination using SMOTE

Input: The original set of features $\mathcal{S}$
Output: An ordered vector of features $\mathcal{S}^\dagger$
1. $\mathcal{S}^\dagger \leftarrow \emptyset$
2. $\mathcal{T}' \leftarrow \mathrm{SMOTE}(\mathcal{T})$
3. **repeat**
4. $\quad (\mathcal{TR}, \mathcal{V}) \leftarrow$ Holdout using $\mathcal{T}'$
5. $\quad \boldsymbol{\alpha} \leftarrow$ SVM Training using $\mathcal{TR}$
6. $\quad \mathcal{I} \leftarrow \mathrm{argmin}_{\mathcal{I}} \sum_{j \in \mathcal{I}} \mathrm{LOSS}(\boldsymbol{\alpha}, \mathcal{S} \setminus \{j\}, \mathcal{TR}, \mathcal{V}),\ \mathcal{I} \subset \mathcal{S}$
7. $\quad \mathcal{S} \leftarrow \mathcal{S} \setminus \mathcal{I}$
8. $\quad \mathcal{S}^\dagger \leftarrow (\mathcal{S}^\dagger, \mathcal{I})$
9. **until** $\mathcal{S} = \emptyset$

5. Experimental results and discussions

We applied the proposed approaches for feature selection on six DNA microarray data sets which have already been used for benchmark feature selection algorithms (e.g. [28,42]). We compared our proposals for feature selection with the alternative methods such as Fisher Score, l_1-SVM, l_0-SVM, and SVM-RFE (for both, linear as well as nonlinear classifiers).

We first show the data sets in Section 5.1, while Section 5.2 presents a summary of the results obtained for the proposed approaches, considering linear and Gaussian kernels, as well as SMOTE oversampling. The respective details are provided in Appendix A. Finally, a sensitivity analysis of the different parameters and a further discussion from the results is presented in Section 5.3.

5.1. Description of data sets and validation procedure

5.1.1. Lung cancer data set (LUNG)

The lung cancer data set contains the gene expression of 181 samples (31 malignant and 150 normal) described by 12533 features [3]. We compared our results using the following number of features: 20, 50, 100, 250, 1000, 2000, 4000, and 12,533 (i.e. no features removed).

5.1.2. GLIOMA data set

The GLIOMA data set contains 50 instances described by 4433 genes in four classes: *cancer glioblastomas*, *non-cancer glioblastomas*, *cancer oligodendrogliomas*, and *non-cancer oligodendrogliomas* [22,42]. To adapt this data set to binary classification we studied class *cancer oligodendrogliomas* (seven instances) versus the rest, using the following number of features: 20, 50, 100, 250, 1000, 2000, 4433 (i.e. no features removed).

5.1.3. SRBCT data set

The SRBCT data set contains 83 samples in four classes: *Ewing family of tumors*, *Burkitt lymphoma*, *neuroblastoma*, and *rhabdomyosarcoma* [18,42]. Each sample contains 2308 genes. To adapt this data set to binary classification we studied class *Burkitt lymphoma* (eleven instances) versus the rest, using the following number of features: 20, 50, 100, 250, 1000, 2308 (i.e. no features removed).

5.1.4. LUNG data set (LUNG2)

The LUNG2 data set contains 203 samples described by 3312 genes in five classes: *adenocarcinomas*, *squamous cell lung carcinomas*, *pulmonary carcinoids*, *small-cell lung carcinomas*, and *normal lung* [4,42]. To adapt this data set to binary classification we studied class *small-cell lung carcinomas* (20 instances) versus the rest, using the following number of features: 20, 50, 100, 250, 1000, 2000, 3312 (i.e. no features removed).

5.1.5. CAR data set

The CAR data set contains 174 samples described by 9182 genes in eleven classes: *prostate*, *bladder/ureter*, *breast*, *colorectal*, *gastroesophagus*, *kidney*, *liver*, *ovary*, *pancreas*, *lung adenocarcinomas* and *lung squamous cell carcinoma* [32,42]. To adapt this data set to binary classification we studied class *kidney* (eleven instances) versus the rest, using the following number of features: 20, 50, 100, 250, 1000, 2000, 4000 and 9182 (i.e. no features removed).

5.1.6. Bullinger data set (BULL)

The Bullinger data set [7] stems from a study for Adult Acute Myeloid Leukemia classification. The preprocessed data contains the gene expression of 94 samples (4 with preceding malignancy and 90 without preceding malignancy) described by 17,404 features. We compared our results using the following number of features: 20, 50, 100, 250, 1000, 2000, 4000, 8000, and 17,404 (i.e. no variables removed).

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

For model evaluation we chose leave-one-out (LOO) cross-validation, since all six data sets have relatively few instances, as is usually the case for microarray data sets [38,42]. In each iteration of the LOO approach, feature selection is performed in

Table 1
Number of features, number of examples, percentage of each class, and Imbalance Ratio (IR) for all six data sets.

Dataset	#Features	#Examples	%Class (min.,maj.)	IR
LUNG	12,533	181	(17.1, 82.9)	4.85
GLIOMA	4433	50	(14.0, 86.0)	6.14
SRBCT	2308	83	(13.3, 86.7)	6.55
LUNG2	3312	203	(9.8, 90.2)	9.15
CAR	9182	174	(6.3, 93.7)	14.8
BULL	17404	94	(4.3, 95.7)	22.5

the training set, ranking the features according to their respective contribution measures. Several SVM classifiers are finally constructed for an increasing number of ranked features, evaluating their performance with the test data. The classification performance (accuracy and gmean, see Section 2.4) is finally computed by averaging the test results.

In order to study the classification performance of the proposed linear feature selection approaches we compared the results for a given number of features with different feature selection algorithms. For linear SVM we used a penalty parameter $C = 1$. For nonlinear SVM classifiers with Gaussian kernel we used $C = 1000$ and $\sigma = 3$, as suggested in Rakotomamonjy [27]. We present the results for the following proposed feature selection models.

- BFE $-$ SVM$_{0-1}$: BFE-SVM for feature ranking (Algorithm (2)) using linear kernel and 0–1 loss function (10).
- BFE $-$ SVM$_{bl}$: BFE-SVM for feature ranking (Algorithm (2)) using linear kernel and balanced loss function (11).
- HO $-$ BFE$_{0-1}$: HO-BFE for feature ranking (Algorithm (3)) using linear kernel and 0–1 loss function (12).
- HO $-$ BFE$_{bl}$: HO-BFE for feature ranking (Algorithm (3)) using linear kernel and balanced loss function (13).
- BFE $-$ SVM$_{0-1k}$: BFE-SVM for feature ranking (Algorithm (2)) using Gaussian kernel and 0–1 loss function (10).
- BFE $-$ SVM$_{blk}$: BFE-SVM for feature ranking (Algorithm (2)) using Gaussian kernel and balanced loss function (11).
- HO $-$ BFE$_{0-1k}$: HO-BFE for feature ranking (Algorithm (3)) using Gaussian kernel and 0–1 loss function (12).
- HO $-$ BFE$_{blk}$: HO-BFE for feature ranking (Algorithm (3)) using Gaussian kernel and balanced loss function (13).
- All previous approaches considering SMOTE resampling over the training subset.

Additionally, the following well-known feature selection algorithms are applied for comparison purposes.

- l_1-SVM$_r$: l_1-SVM for feature ranking (Formulation (8)), using standard SVM as classifier (Formulation (2)).
- SVM $-$ RFE$_l$: SVM-RFE for feature ranking (linear kernel, Algorithm (1)), using standard SVM as classifier (Formulation (2)).
- l_0-SVM: l_0-SVM for feature ranking (Formulation (9)), using standard SVM as classifier (Formulation (2)).
- Fisher + SVM: Fisher Score for feature ranking (Formulation (6)), using standard SVM as classifier (Formulation (2)).
- SVM $-$ RFE$_{nl}$: SVM-RFE for feature ranking (Gaussian kernel, Algorithm (1)), using kernel-based SVM as classifier (Formulation (4)).
- All previous approaches considering SMOTE resampling over the training subset.

5.2. Summary of classification results for imbalanced microarray datasets

In this section, we present a summary of the results obtained from our experiments; the details are provided in Appendix A. In order to assess the best performance and the stability of the respective approaches, we provide the maximum and average leave-one-out (LOO) gmean values for all predefined subsets of attributes for all six microarray datasets; see Table 2.

According to the results presented in Table 2, the approaches from our proposed family of methods provide best overall performance in four out of six data sets, and in the remaining two the performance is similar compared to the best alternative approaches. Within the proposed strategies, the Holdout approaches tend to perform better than the basic backward approach that considers only the training samples. The balanced loss function outperforms the standard 0–1 loss function on imbalanced data sets, with the only exception of the GLIOMA dataset.

Another important result is that in general there is a low influence of data resampling on classification performance. Only in the BULL data set resampling was relevant, achieving 69.9% gmean with linear HO $-$ BFE$_{bl}$ for a data set that was not possible to shatter. We conclude that in extremely imbalanced and overlapping data sets, SMOTE can be very useful to improve the classifier's performance, but it does not provide a significant improvement otherwise.

Excluding data sets where a classifier using all available features (i.e. no feature selection) obtains already very high classification performance, an adequate feature selection significatively improves predictive performance.

It may sound counterintuitive that less information is actually better than using all variables. However, several researchers have already proved the advantages of feature selection to mitigate the *curse of dimensionality* [15,21,27]. This is especially true for microarray data sets where the importance of feature selection has been emphasized (see e.g. [31]) since usually only very few observations can be obtained that are typically described by a huge number of genes (features).

Another important observation is that linear feature selection methods perform consistently better than kernel-based approaches. We infer that the difficulty of setting the right combination of parameters C and σ without using a time consuming crossvalidation strategy (which may also lead to overfitting) explains this loss in performance. For this work, we set a fixed value of C for linear methods and a fixed tuple (C, σ) for kernel-based methods, as has been proposed in the literature (see, for example, [27]). Empirically, we observe a strong relationship between the number of variables in the solution and the optimal value for σ (an SVM classifier normally requires a larger σ when the number of features increases [20,21]), and therefore a model selection procedure for each number of ranked features is recommended. This topic is further discussed in the following subsection.

Table 2
Maximum and average LOO gmean, in percentage. Average over all ranked features. Comparison among all approaches. Best results for each dataset are presented in bold.

		LUNG		GLIOMA		SRBCT		LUNG2		CAR		BULL	
		max.	avg.	max.	avg.	max.	avg.	max.	avg.	max.	avg.	max.	avg.
No. resampling	l_1-SVM$_r$	96.4	96.7	63.6	74.7	99.9	100	**100**	**100**	92.6	95	0	0
	SVM $-$ RFE$_l$	96.6	96.7	66.5	75.6	99.9	100	**100**	**100**	91.6	95.3	0	0
	Fisher + SVM	97.8	98.4	66.9	73.8	99.9	100	99.3	100	90.5	90.5	0	0
	l_0-SVM	96.4	96.7	66.5	75.6	99.9	100	**100**	**100**	92.6	95	0	0
	SVM $-$ RFE$_{nl}$	98.0	98.4	59.9	83.5	70.4	100	81.9	100	63.6	90.5	0	0
	BFE $-$ SVM$_{0-1}$	95.8	98.4	**83.7**	**92.9**	95.3	100	**100**	**100**	90.5	95.3	0	0
	BFE-SVM$_{bl}$	95.8	98.4	80.8	92.6	99.9	100	**100**	**100**	92.1	95.3	0	0
	HO $-$ BFE$_{0-1}$	97.5	100	83.0	92.6	95.3	100	**100**	**100**	91.0	95.3	0	0
	HO-BFE$_{bl}$	**98.1**	**100**	81.7	90.4	96.5	100	**100**	**100**	**93.1**	**100**	0	0
	BFE $-$ SVM$_{0-1k}$	94.7	96.7	53.9	82.50	0	0	0	0	0	0	0	0
	BFE $-$ SVM$_{blk}$	94.7	96.7	72.5	92.6	58.2	95.3	**100**	**100**	0	0	0	0
	HO $-$ BFE$_{0-1k}$	97.6	99.7	54.1	82.5	55.1	89.8	71.6	100	58.7	100	0	0
	HO $-$ BFE$_{blk}$	96.9	98.0	55.9	84.5	58.2	95.3	72	100	55.8	95.3	0	0
SMOTE	l_1-SVM$_r$	92.1	96.7	58.9	72.9	99.9	100	99.1	100	91.6	95.3	0	0
	SVM $-$ RFE$_l$	97.1	98.4	64.5	74.7	**100**	**100**	**100**	**100**	91.6	95.3	0	0
	Fisher + SVM	97.7	98.4	69.7	73.8	99.9	100	99.3	100	90.5	90.5	0	0
	l_0-SVM	96.6	98.4	65.6	75.6	99.8	100	**100**	**100**	92.1	95.3	0	0
	SVM $-$ RFE$_{nl}$	92.8	100	62.0	81.5	81.9	100	85.2	100	82.8	95.3	0	0
	BFE $-$ SVM$_{0-1}$	95.3	99.7	82.7	92.6	96.8	100	**100**	**100**	91.5	95.3	25.6	64.1
	BFE-SVM$_{bl}$	92.9	98.4	72.7	78.4	99.2	100	99.2	100	92.6	95.3	14.3	48.6
	HO $-$ BFE$_{0-1}$	93.2	98.4	78.8	92.6	98.7	100	**100**	**100**	92.1	95.3	17.8	67.1
	HO-BFE$_{bl}$	97.1	100	74.8	82.5	95.8	100	98.3	100	92.6	95.3	11.9	**69.9**
	BFE $-$ SVM$_{0-1k}$	82.0	98.4	65.4	83.5	56.1	89.8	71.5	100	63.4	99.7	10.5	57.7
	BFE $-$ SVM$_{blk}$	83.7	98.0	51.8	75.6	69.1	95.3	81.3	100	61.2	94.5	25.3	59.2
	HO $-$ BFE$_{0-1k}$	85.8	98.4	70.8	84.5	53.6	90.5	79.4	100	63.4	99.7	**25.7**	64.5
	HO $-$ BFE$_{blk}$	81.4	98.4	60.8	91.5	65.2	94.7	78.6	100	89.0	90.5	16.6	68.3

Table 3
G-mean, in percentage. Average and maximum along all ranked features. Sensitivity analysis for parameter C.

	BFE $-$ SVM$_{0-1}$		BFE-SVM$_{bl}$		HO $-$ BFE$_{0-1}$		HO-BFE$_{bl}$	
	Mean	Max	Mean	Max	Mean	Max	Mean	Max
2^{-3}	37.1	75.6	37.1	75.6	37.8	90.4	37.6	73.8
2^{-2}	48.7	92.6	48.7	92.6	45.6	82.5	42.3	74.7
2^{-1}	67	92.6	71.7	92.6	51.4	81.5	67.1	75.6
2^{0}	77.4	92.6	80.8	92.6	59.1	97.6	70.7	90.4
2^{1}	82.2	92.6	82.2	92.6	53.2	95.2	75.8	82.5
2^{2}	82.3	92.6	82.1	92.6	58.3	73.8	76.3	83.5
2^{3}	82.3	92.6	82.1	92.6	65.1	75.6	81.7	90.4
2^{4}	82.2	92.6	82.1	92.6	62	76.2	74.1	81.5
2^{5}	82.2	92.6	82.1	92.6	52.4	74.7	77.4	88.2

Table 4
G-mean, in percentage. Average and maximum along all ranked features. Sensitivity analysis for parameter σ.

	BFE $-$ SVM$_{0-1k}$		BFE $-$ SVM$_{blk}$		HO $-$ BFE$_{0-1k}$		HO $-$ BFE$_{blk}$	
	Mean	Max	Mean	Max	Mean	Max	Mean	Max
3^{0}	29.7	89.3	29.7	89.3	26.4	74.7	26.3	72.9
3^{1}	53.9	82.5	72.5	92.6	54.1	82.5	55.9	84.5
3^{2}	77.7	92.6	77.7	92.6	73	83.5	68.4	75.6
3^{3}	63.6	92.6	63.6	92.6	59	83.5	58	83.5
3^{4}	41.9	75.6	41.9	75.6	37	74.7	37.1	75.6
3^{5}	15.9	73.8	15.9	73.8	13.9	73.8	14	73.8

Table 5
G-mean, in percentage. Average and maximum along all ranked features. Sensitivity analysis for the percentage of variables eliminated at each iteration of the algorithm.

	$BFE-SVM_{0-1}$		$BFE\text{-}SVM_{bl}$		$HO-BFE_{0-1}$		$HO\text{-}BFE_{bl}$	
	Mean	Max	Mean	Max	Mean	Max	Mean	Max
10%	82.2	92.6	78.8	92.6	64.9	82.5	77.1	83.5
20%	81.2	92.6	81.8	92.6	73.1	82.5	74.1	88.2
30%	80.3	92.6	79.9	92.6	64.9	89.3	75.9	82.5
40%	80.4	91.5	79.9	92.6	53.1	74.7	77.5	92.6
50%	77.7	92.6	79.3	91.5	71.7	82.5	71.4	73.8

5.3. Sensitivity analysis of the relevant parameters and discussion

In this subsection we study the performance of the proposed methodologies by performing sensitivity analysis of the relevant parameters, characterizing their influence on the final solution. We also discuss issues related to stability, the presence of redundant variables in the final solution, and computational time.

5.3.1. Sensitivity analysis

To illustrate the influence of the parameters on the classifier, we varied C, σ (kernel-based approaches) and the percentage of variables removed at every iteration of the algorithm considering the GLIOMA data set. This data set was selected because it has the highest variance in the results, and therefore it is adequate to illustrate difference between models. Table 3 presents the results of varying C along the values $C \in \{2^{-3}, 2^{-2}, \ldots, 2^{4}, 2^{5}\}$, for all proposed linear feature selection methods and eliminating 10% of the attributes at each iteration. Subsequently, Table 4 presents the results of varying σ along the values $\sigma \in \{3^{1}, 3^{2}, \ldots, 3^{5}\}$, for all proposed nonlinear feature selection methods, for a fixed value of C ($C = 1000$) and eliminating 10% of the attributes at every iteration. Finally, Table 5 presents the results of varying the percentage of variables eliminated at every iteration of the best approach, along 10%, 20%, 30%, 40%, and 50%. In all cases, we present the average and maximum performance (in terms of gmean) for all subsets of selected features.

From Table 3 we observe that results are very stable along the values above 1. The difference between all values are not significant according to an ANOVA test (p value 0.848). In contrast, in Table 4 we observe that prediction performance can be strongly affected by the choice of the kernel, and the range of proper values is very small. Additionally, the choice of the kernel depends on the number of selected attributes: the kernel width should start with high values and decrease together with the number of selected features. Finally, from Table 5 we observe that predictive performance slightly decreases when more attributes are removed simultaneously, as the intuition suggests, but these differences are not statistically significant.

5.3.2. Stability

To study the method's stability regarding the holdout step, different runs of the algorithm were considered for both variants: one holdout step and five holdout steps. For all runs we consider the same parameters (linear kernel with $C = 1$ and 10% of feature elimination). Applying an ANOVA test, we detect non-significant differences for all methods ($p = 0.588$ for $HO - BFE_{0-1}$ with one holdout step, $p = 0.117$ $HO - BFE_{bl}$ with one holdout step, $p = 0.959$ for $HO - BFE_{0-1}$ with five holdout steps, and $p = 0.353$ for $HO - BFE_{bl}$ with five holdout steps), but the methods based on 5 holdout steps are statistically more stable than the ones based on one holdout step.

Another possible issue is that the approach is order-dependent in case of similar values of the loss function. The percentage of "ties" in the values of the loss function can be computed for all methods to assess the influence of the order in which the variables are presented. Averaging along all subsets of features, we obtain 90.0% of "ties" for $BFE - SVM_{0-1}$, 72.7% for $BFE - SVM_{bl}$, 73.0% for $HO - BFE_{0-1}$, and 0% for $HO - BFE_{bl}$. We conclude that the holdout steps and the use of the balanced loss function effectively mitigates the risk of eliminating relevant features given by order-dependency.

5.3.3. Redundant variables

Our next experiment explores the appearance of (highly) redundant variables in the solution. Wrapper and embedded methods take the relationship between variables, and therefore they should be more effective in identifying and eliminating redundancy. In contrast, if several attributes are simultaneously removed at every iteration, this may result in a higher risk of removing relevant attributes. For instance, when the weight vector is used to eliminate variables, relevant but redundant variables may share weight in the hyperplane and be removed together. To reduce this risk, our approach attempts at recreating the main goal for this task: to achieve the best predictive performance in an "unseen" subset, using a cost-sensitive metric. Empirically, we proved that this strategy correctly identifies those attributes that are relevant to discriminate in a class-imbalanced problem, given the results presented in previous section. To demonstrate that our approach also correctly identifies those irrelevant and redundant attributes, we studied the percentage of highly redundant attributes (Pearson's $\rho > 0.85$) for all methods and for all different subset of selected features. These results are presented in Table 6.

Table 6
Percentage of highly redundant attributes for all methods, and for all different subset of selected features.

	Fisher + SVM	l_0-SVM	SVM-RFE$_l$	BFE-SVM$_{0-1}$	BFE-SVM$_{bl}$	HO-BFE$_{0-1}$	HO-BFE$_{bl}$
4434	0.12	0.12	0.12	0.12	0.12	0.12	0.12
2000	0.38	0.16	0.14	0.09	0.09	0.08	0.09
1000	0.83	0.24	0.15	0.09	0.09	0.10	0.11
500	1.57	0.28	0.17	0.12	0.12	0.15	0.14
200	2.17	0.47	0.13	0.17	0.17	0.28	0.23
100	3.09	0.22	0.10	0.42	0.34	0.22	0.24
50	3.76	0.24	0.16	0.41	0.65	0.24	0.24
20	4.74	0.53	0.00	0.00	0.53	1.05	0.53
10	4.44	0.00	0.00	0.00	0.00	2.22	2.22

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

Model	Time (s)
Fisher + SVM	1.32
l_0-SVM	2.89
SVM-RFE$_l$	4.33
BFE-SVM$_{0-1}$	6.18
BFE-SVM$_{bl}$	9.12
HO-BFE$_{0-1}$	5.78
HO-BFE$_{bl}$ (1 runs)	8.71
HO-BFE$_{bl}$ (5 runs)	38.89

According to Table 6, Fisher Score has a significantly higher percentage of redundant variables, compared to all other wrapper and embedded approaches. The difference between these approaches is not significant, although RFE-SVM and BFE − SVM$_{bl}$ are the best approaches in this context.

5.3.4. Computational time

Table 7 reports the average CPU times of one fold for all algorithms, considering similar parameters (ranking construction and SVM classification). For these experiments we use an Intel i7-3630QM system with CPU at 2.4 GHz and 16 GB of RAM.

From Table 7 we observe that the running times for all proposed algorithms are relatively similar, with the only exception of the HO-BFE version, whose running time is about five times higher, as expected, since it performs five repetitions of the algorithm for stability reasons. Fisher Score is the fastest approach, followed by the l_0-SVM. There is no significant difference between the use of different contribution metrics nor the use of a holdout step versus using the entire training set.

5.3.5. Conclusions from sensitivity analysis

We draw the following conclusions from these experiments:

- Results obtained by varying parameter C are very stable above $C = 1$, and therefore it can be fixed to a sufficiently large value during the experimental framework.
- In contrast, kernel-based methods behave very unstable in terms of performance. The main reason for this is the difficulty in selecting the optimum value for parameter σ. The right σ value seems to be strongly dependent to the number of selected features for the classification task. To achieve better performance with kernel methods it would be necessary to perform a grid search at each step of the backward algorithm, leading to a very time-consuming validation strategy.
- The holdout step provides a framework that adequately emulates the ultimate goal of feature selection, which is selecting those features that are relevant for predicting new data, resulting in an important gain in terms of performance. This method, however, requires the average of several holdout steps in order to assure stable results. This fact is particularly important in microarray applications, where only limited number of training samples are available. Empirical results demonstrate the effectiveness and stability of this approach, which can be performed in tractable running times.
- The correct identification of redundant attributes is an important topic in high-dimensional applications, such as classification of microarray data. Backward approaches with batch elimination of attributes have a high risk of eliminating relevant variables at the first iterations, leading to poor predictive results. Furthermore, filter approaches that do not take into account correlations between attributes may include a high percentage of redundant variables in their final selection. We empirically assess both approaches, by first varying the percentage of batch elimination in our investigation (Table 5), and second computing the percentage of highly redundant attributes selected by each method (Table 6). The above-mentioned sensitivity analysis confirms the good results obtained in terms of performance by our approach varying the batch elimination criterion, finding a good trade off between performance and running time around 10–20%. Given these results

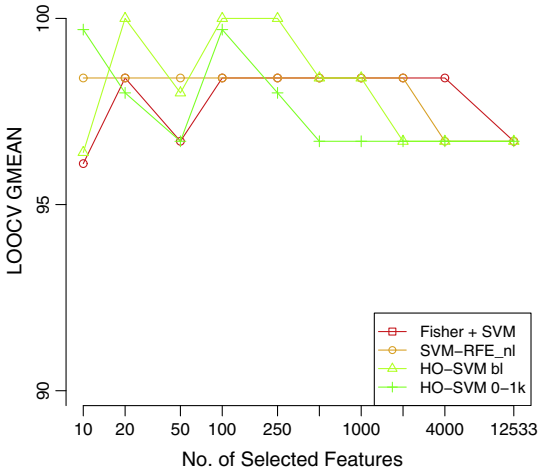
we conclude that the method successfully finds those attributes that are relevant to construct the hyperplane of the respective classifier. We also confirm that embedded approaches are more effective at eliminating redundancy than filter methods, according to Table 6.

- Regarding computational complexity of our approaches, the backward elimination process is similar to RFE-SVM, differing only in the holdout step and the computation of the contribution metrics. Computationally speaking, these two steps are significantly less expensive than the backward elimination process. The complexity of RFE-SVM is of the order of $max(n, m)m^2$ considering the operations of SVM (computation of the kernel matrix and its inversion) and the successive iterations with a decreasing number of features (assuming a reduction by a factor of 2 at each iteration) [14].

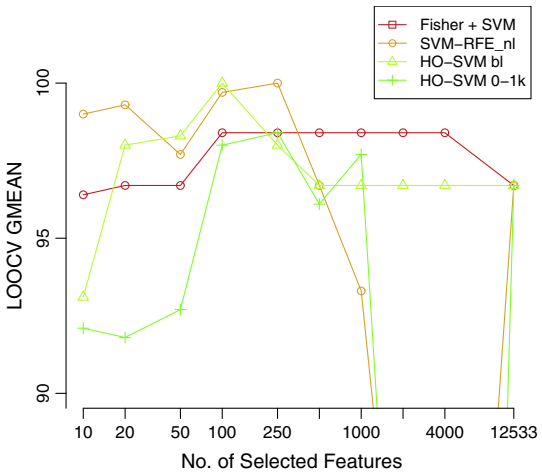
6. Conclusions and future work

We introduced a family of methods based on a backward elimination approach for feature ranking and embedded classification using Support Vector Machines, which has been adapted to select those attributes that are relevant to discriminate between classes under imbalanced data conditions.

Four different strategies were designed to accomplish this objective: We combined two different backward elimination strategies, one based on training performance (BFE-SVM) and another based on predictive performance via successive holdout steps (HO-BFE). We applied two different evaluation measures, a standard 0–1 loss function and the balanced loss function to explicitly favor those attributes whose elimination leads to less errors in the minority class. Our approaches present the following advantages, based on a comparison with other feature selection approaches for SVM in high-dimensional applications with imbalanced datasets (such as microarray datasets).

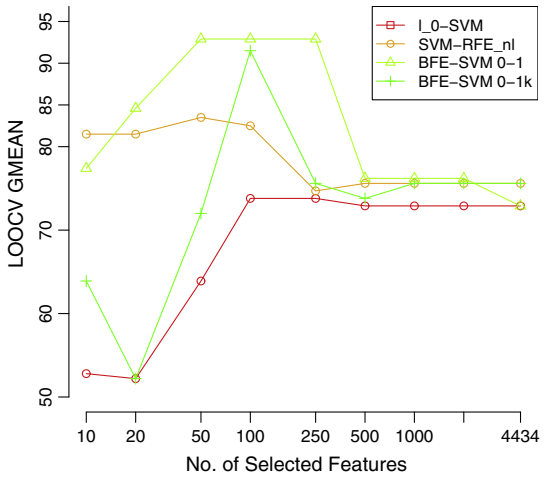


(a) No resampling

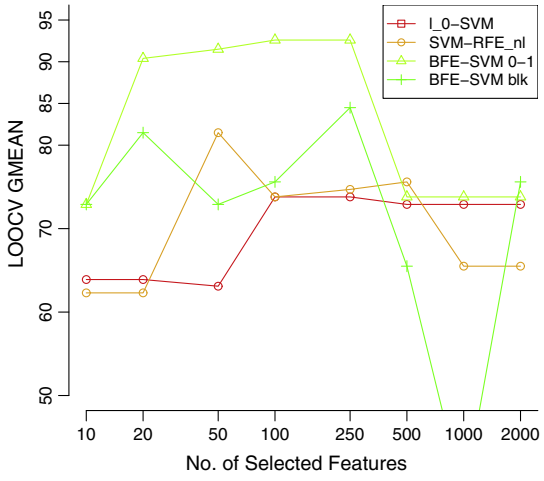


(b) SMOTE

Fig. A.1. GMEAN versus the number of ranked variables for different feature selection approaches. LUNG dataset.



(a) No resampling



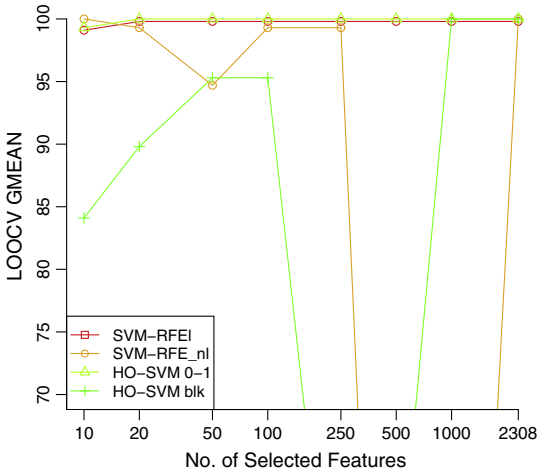
(b) SMOTE

Fig. A.2. GMEAN versus the number of ranked variables for different feature selection approaches. GLIOMA dataset.

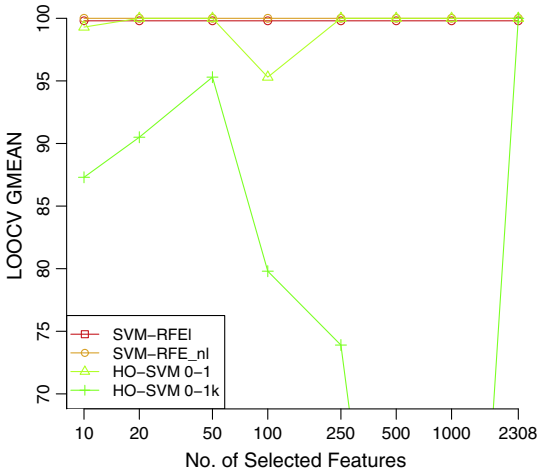
- The proposed approaches outperform other feature ranking techniques in terms of predictive performance for different SVM-based feature selection techniques, achieving particularly good results on highly imbalanced data sets, based on their ability to identify irrelevant variables using the classifier and minimizing the number of errors in the minority class, which is assumed to have a higher cost.
- The proposed methods allow for the explicit incorporation of misclassification costs in the assessment of each attribute's contribution, leading to a feature selection process especially designed for a particular application.
- Our strategies are very flexible and allow the use of different kernel functions for nonlinear feature selection and classification using SVM. Furthermore, they can also be generalized to various classification methods, other than SVM.

In Section 5 we proposed an experimentation setup to avoid an uneven comparison between linear and nonlinear methods, considering similar efforts in both cases. Although no significant gain was obtained in our experiments by using kernel methods, the proposed strategies have the potential to achieve better results under a more exhaustive model selection, by embedding the feature selection process. Given the good results obtained while combining SVM for feature ranking with the same approach for classification, we suggested performing a grid search for parameters C and σ in each iteration of the algorithm. This was through monitoring the performance in order to obtain a final solution along the process, without considering feature selection and classification as independent problems. According to our results, the success of a nonlinear backward elimination strategy is strongly dependent on the right definition of the decision boundary, and in particular on the correct setting of the parameter σ for high-dimensional applications.

The experimentation procedure also allows the comparison with feature selection approaches that are not explicitly adapted to imbalanced data sets. According to our results, the adapted version of HO-BFE using balanced loss leads to



(a) No resampling



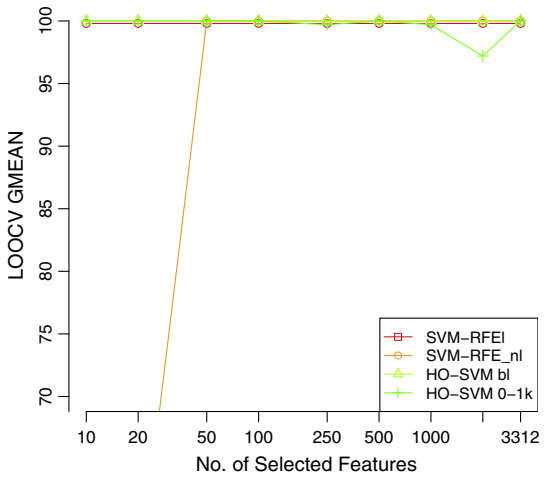
(b) SMOTE

Fig. A.3. GMEAN versus the number of ranked variables for different feature selection approaches. SRBCT dataset.

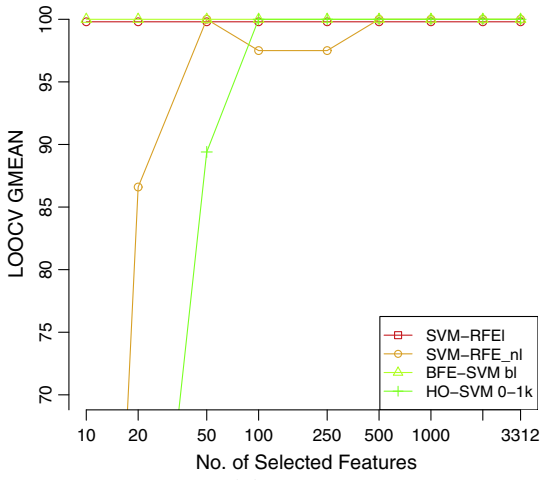
consistently better results in terms of accuracy and gmean compared to alternative feature selection approaches, resulting in an attractive alternative for high-dimensional problems with imbalanced data sets. The main drawback, however, is the higher running time of the algorithm when performing different holdout partitions and averaging them, instead of using only the training sample. However, the predictive performance of this procedure is significantly higher in most cases. Furthermore, the order of the algorithm for BFE-SVM with balanced loss is exactly the same as that of RFE-SVM, and it may lead to better results by defining an adequate loss function for classification performance with imbalanced data sets.

Another important conclusion was the need for an intelligent oversampling in extreme cases of class imbalance and overlap, in which no adequate classifier can be found, since embedded and wrapper feature selection strongly depend on the classification method. In other cases SMOTE oversampling did not improve the solutions found without data resampling.

There are several opportunities for future work. First, classification of imbalanced data sets can be performed by using one-class SVM approaches, such as SVDD (Support Vector Data Description [34]). We are currently working on embedded feature selection strategies for this approach in order to identify those features that are relevant in constructing the data description. Secondly, the costs of Type I and Type II errors may be different and the balanced loss function could be adapted to different profit or cost functions in a particular domain. Credit scoring, fraud detection, and churn prediction are some examples of applications with imbalanced data sets where a profit function can be constructed to evaluate both the performance of the classifier and the relevance of its features. In applications where high correlation among sets of features is to be expected, simultaneous feature grouping and selection as suggested e.g. by [44] could be an interesting approach to improve problem understanding and classifier performance. Along the same line it would be interesting in a given application to have a domain expert interpreting the selected features and establishing their relevance in the particular case.



(a) No resampling



(b) SMOTE

Fig. A.4. GMEAN versus the number of ranked variables for different feature selection approaches. LUNG2 dataset.

Acknowledgments

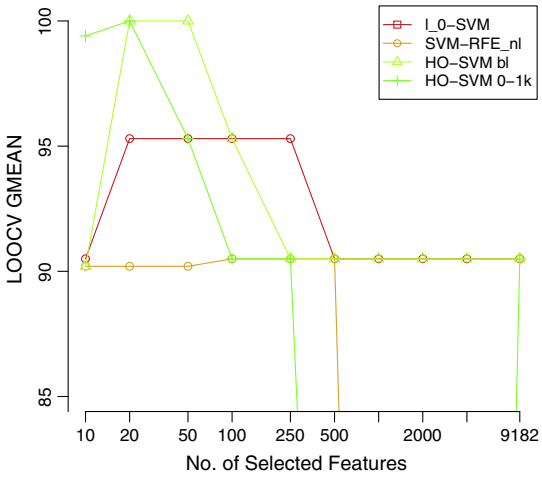
Support from the Millennium Science Institute on Complex Engineering Systems (www.isci.cl; ICM: P-05-004-F, CONICYT: FBO16) is greatly acknowledged. The first author was supported by FONDECYT Project 11121196.

Appendix A. Detailed feature selection results

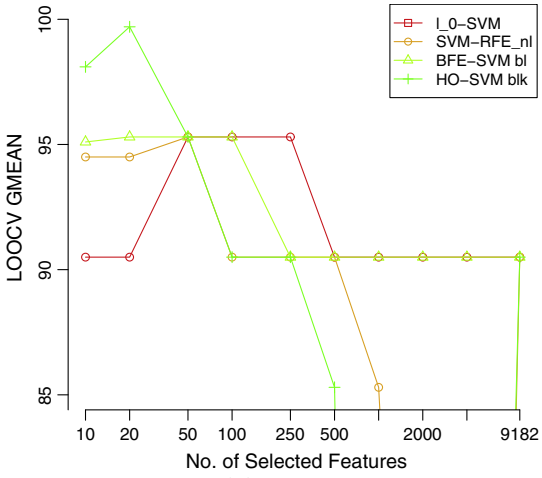
The results of the leave-one-out crossvalidation (in terms of g-mean) are shown in Figs. A.1 (LUNG dataset), A.2 (GLIOMA dataset), A.3 (SRBCT dataset), A.4 (LUNG2 dataset), A.5 (CAR dataset) and A.6 (BULL dataset). The figures present the best proposed and alternative approaches using linear and Gaussian kernel respectively for an increasing number of selected features. The upper figure shows the results without resampling, while the lower figure depicts the results obtained using the same approaches but with SMOTE oversampling.

From Fig. A.1(a) we observe that the proposed approach HO-BFE using balanced loss and linear kernel has the best predictive performance, achieving perfect classification when using 20, 100, and 250 features, followed by the best proposed approach using Gaussian kernel (HO-BFE using 0–1 loss). Alternative approaches behave relatively similar in this data set. The results are relatively similar when using SMOTE (see Fig. A.1(b)), with the exception of the kernel methods, which show poor performance when using more than 1000 features.

For GLIOMA data set (see Fig. A.2(a)), the proposed approaches BFE-SVM with 0–1 loss using linear and Gaussian kernel achieved best performance when using 100 variables. The difference is significant compared with alternative approaches,



(a) No resampling



(b) SMOTE

Fig. A.5. GMEAN versus the number of ranked variables for different feature selection approaches. CAR dataset.

where RFE-SVM with Gaussian kernel achieved the best performance. All methods behave similarly when using 500 variables or more. No improvements are obtained when using SMOTE (see Fig. A.2(b)), while some approaches worsen their performance significantly.

For the SRBCT data set all linear methods achieve very good and stable results, while kernel approaches failed at identifying relevant attributes in some cases; see A.3(a). Best performance is obtained by our approach BFE-SVM with balanced loss and RFE-SVM with linear kernel. Results are very similar when using SMOTE oversampling; see Fig. A.3(b).

A similar situation can be observed for LUNG2 (Fig. A.4), where perfect results are achieved by the approaches HO-BFE with balanced loss and RFE-SVM with linear kernel. Kernel approaches fail at correctly identifying relevant features, especially when using SMOTE oversampling.

Fig. A.5(a) presents the results for the CAR data set without resampling. Here we observe similar performance for all approaches when using 500 attributes or more, and then our approaches HO-BFE with balanced loss (best proposed linear method) HO-BFE with 0–1 loss (best kernel-based method) and outperform alternative feature selection strategies, improving classification performance significantly. Once again, no significant improvement is achieved when using SMOTE resampling, as shown in Fig. A.5(b).

Results for the BULL data set, the one with the highest imbalance ratio, are presented in Fig. A.6. For the case without resampling, no approach is able to perform feature selection and classification adequately, and the classifiers tend to predict all instances to the majority class. SMOTE helped to find discriminative classifiers with an adequate balanced classification performance with fewer features for the proposed approaches only.

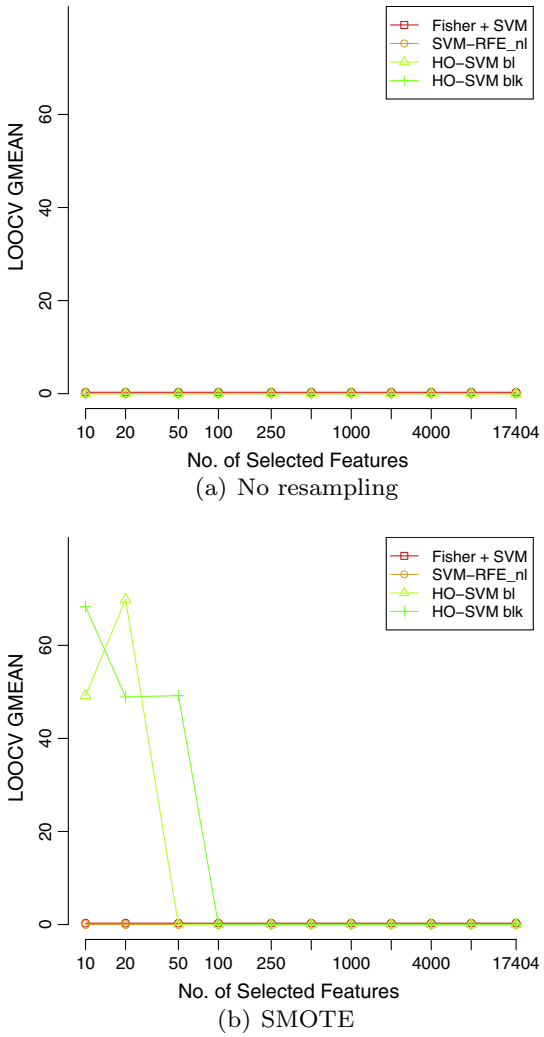


Fig. A.6. GMEAN versus the number of ranked variables for different feature selection approaches. BULL dataset.

References

[1] M. Alibeigi, S. Hashemi, A. Hamzeh, DBFS: an effective density based feature selection scheme for small sample size and high dimensional imbalanced data sets, Data Knowl. Eng. (2012) 67–103.
[2] K. Balasubramanian, B.K. Sriperumbudur, G. Lebanon, Ultrahigh dimensional feature screening via RKHS embeddings, in: Proceedings of the Sixteenth International Conference on Artificial Intelligence and Statistics, 2013.
[3] D.G. Beer, S.L. Kardia, C.C. Huang, T.J. Giordano, A.M. Levin, D.E. Misek, L. Lin, G. Chen, T.G. Gharib, D.G. Thomas, M.L. Lizyness, R. Kuick, S. Hayasaka, J.M.G. Taylor, M.D. Iannettoni, M.B. Orringer, S. Hanash, Gene-expression profiles predict survival of patients with lung adenocarcinoma, Nat. Med. 8 (2002) 816–824.
[4] A. Bhattacharjee, W.G. Richards, J. Staunton, C. Li, S. Monti, P. Vasa, C. Ladd, J. Beheshti, R. Bueno, M. Gillette, M. Loda, G. Weber, E.J. Mark, E.S. Lander, W. Wong, B.E. Johnson, T.R. Golub, D.J. Sugarbaker, M. Meyerson, Classification of human lung carcinomas by mRNA expression profiling reveals distinct adenocarcinoma subclasses, Proc. Nat. Acad. Sci. USA 98 (2001) 13790–13795.
[5] R. Blagus, L. Lusa, Class prediction for high-dimensional class-imbalanced data, BMC Bioinform. 2010 (2010) (11:523).
[6] P. Bradley, O. Mangasarian, Feature selection vía concave minimization and support vector machines, in: Machine Learning Proceedings of the Fifteenth International Conference (ICML'98), Morgan Kaufmann, 1998, pp. 82–90.
[7] L. Bullinger, K. Dohner, S. Frohling, E. Bair, R.F. Schlenk, R. Tibshirani, H. Dohner, J.R. Pollack, Use of gene-expression profiling to identify prognostic subclasses in adult acute myeloid leukemia, New England J. Med. 350 (16) (2004) 1605–1616.
[8] N.V. Chawla, N. Japkowicz, A. Kotcz, Editorial: special issue on learning from imbalanced data sets, SIGKDD Explor. 6 (1) (2004) 1–6.
[9] N.V. Chawla, L.O. Hall, K.W. Bowyer, W.P. Kegelmeyer, SMOTE: synthetic minority oversampling technique, J. Artif. Intell. Res. (16) (2002) 321–357.
[10] X. Chen, M. Wasikowski, FAST: a roc-based feature selection metric for small samples and imbalanced data classification problems, in: Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2008), 2009, pp. 124–132.
[11] S. Chowdhury, J.K. Sing, D.K. Basu, M. Nasipuri, Face recognition by generalized two-dimensional FLD method and multi-class support vector machines, Appl. Soft Comput. 11 (7) (2011) 4282–4292.
[12] J. Fan, R. Samworth, Y. Wu, Ultrahigh dimensional feature selection: beyond the linear model, J. Mach. Learn. Res. 10 (2009) 2013–2038.
[13] A. Fernández, M.J. del Jesus, F. Herrera, On the 2-tuples based genetic tuning performance for fuzzy rule based classification systems in imbalanced data-sets, Inf. Sci. 180 (8) (2010) 1268–1291.

[14] I. Guyon, S. Gunn, M. Nikravesh, L.A. Zadeh, Feature Extraction, Foundations and Applications, Springer, Berlin, 2006.
[15] I. Guyon, J. Weston, S. Barnhill, V. Vapnik, Gene selection for cancer classification using support vector machines, Mach. Learn. 46 (1–3) (2002) 389–422.
[16] H. He, E. García, Learning from imbalanced data, IEEE Trans. Knowl. Data Eng. 21 (9) (2009) 1263–1284.
[17] A.R. Hidalgo-Munoz, M.M. López, I.M. Santos, A.T. Pereira, M. Vázquez-Marrufo, A. Galvao-Carmona, A.M. Tomé, Application of SVM-RFE on EEG signals for detecting the most relevant scalp regions linked to affective valence processing, Expert Syst. Appl. 40 (6) (2013) 2102–2108.
[18] J. Khan, J.S. Wei, M. Ringner, L.H. Saal, M. Ladanyi, F. Westermann, F. Berthold, M. Schwab, C.R. Antonescu, C. Peterson, P.S. Meltzer, Classification and diagnostic prediction of cancers using gene expression profiling and artificial neural networks, Nat. Med. 7 (2001) 673–679.
[19] D.A. Kumar, V. Ravi, Predicting credit card customer churn in banks using data mining, Int. J. Data Anal. Techn. Strategies 1 (1) (2008) 4–28.
[20] S. Maldonado, R. Weber, A wrapper method for feature selection using Support Vector Machines, Inf. Sci. 179 (13) (2009) 2208–2217.
[21] S. Maldonado, R. Weber, J. Basak, Kernel-penalized SVM for feature selection, Inf. Sci. 181 (1) (2011) 115–128.
[22] C.L. Nutt, D.R. Mani, R.A. Betensky, P. Tamayo, J.G. Cairncross, C. Ladd, U. Pohl, C. Hartmann, M.E. McLaughlin, T.T. Batchelor, P.M. Black, A. von Deimling, S.L. Pomeroy, T.R. Golub, D.N. Louis, Gene expression-based classification of malignant gliomas correlates better with survival than histological classification, Cancer Res. 63 (2003) 1602–1607.
[23] C.H. Papadimitriou, Computational Complexity, Addison Wesley, 1994 (Reading).
[24] B.-J. Park, S.-K. Ohb, W. Pedrycz, The design of polynomial function-based neural network predictors for detection of software defects, Inf. Sci. 229 (2013) 40–57.
[25] R.C. Prati, G.E.A.P.A. Batista, M.C. Monard, Class Imbalances versus class overlapping: an analysis of a learning system behavior, MICAI 2004: Advances in Artificial Intelligence, Lecture Notes in Computer Science, vol. 2972, Springer, Berlin Heidelberg, 2004, pp. 312–321.
[26] Y. Qu, H. Su, L. Guo, J. Chu, A novel SVM modeling approach for highly imbalanced and overlapping classification, Intell. Data Anal. 15 (2011) 319–341.
[27] A. Rakotomamonjy, Variable selection using SVM-based criteria, J. Mach. Learn. Res. 3 (2003) 1357–1370.
[28] G. Rätsch, T. Onoda, K.-R. Müller, Soft margins for AdaBoost, Mach. Learn. 42 (3) (2001) 287–320.
[29] A.A. Shanab, T.M. Khoshgoftaar, R. Wald, J.V. Hulse, Comparison of approaches to alleviate problems with high-dimensional and class-imbalanced data, in: 2011 IEEE International Conference on Information Reuse and Integration (IRI), 2011, pp. 234–239.
[30] M. Sokolova, N. Japkowicz, S. Szpakowicz, Beyond accuracy, F-score and ROC: a family of discriminant measures for performance evaluation, Advances in Artificial Intelligence, vol. 4304, Springer, Berlin Heidelberg, 2006, pp. 1015–1021.
[31] L. Song, A. Smola, A. Gretton, J. Bedo, K. Borgwardt, Feature selection via dependence maximization, J. Mach. Learn. Res. 13 (2012) 1393–1434.
[32] A.I. Su, J.B. Welsh, L.M. Sapinoso, S.G. Kern, P. Dimitrov, H. Lapp, P.G. Schultz, S.M. Powell, C.A. Moskaluk, H.F. Jr. Frierson, G.M. Hampton, Molecular classification of human carcinomas by use of gene expression signatures, Cancer Res. 61 (2001) 7388–7393.
[33] Y. Sun, M.S. Kamel, A.K.C. Wong, Y. Wang, Cost-sensitive boosting for classification of imbalanced data, Pattern Recogn. 40 (12) (2007) 3358–3378.
[34] D.M.J. Tax, R. Duin, Support vector data description, Mach. Learn. 54 (2004) 45–66.
[35] J. Van Hulse, T.M. Khoshgoftaar, A. Napolitano, R. Wald, Feature selection with high-dimensional imbalanced data, in: Proceedings of the 2009 IEEE International Conference ICDMW '09, 2009, pp. 507–514.
[36] V. Vapnik, Statistical Learning Theory, John Wiley and Sons, New York, 1998.
[37] P. Villar, A. Fernández, R.A. Carrasco, F. Herrera, Feature selection and granularity learning in genetic fuzzy rule-based classification systems for highly imbalanced data-sets, International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems 20 (3) (2012) 369–397.
[38] 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.
[39] S. Wang, D. Li, X. Song, Y. Wei, H. Li, A feature selection method based on improved fisher's discriminant ratio for text sentiment classification, Expert Syst. Appl. 38 (7) (2011) 8696–8702.
[40] M. Wasikowski, X. Chen, Combating the small sample class imbalance problem using feature selection, IEEE Trans. Knowl. Data Eng. 22 (10) (2010) 1388–1400.
[41] 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.
[42] K. Yang, Z. Cai, J. Li, G. Lin, A stable gene selection in microarray data analysis, BMC Bioinform. 7 (2006) 228.
[43] Z. Zheng, X. Wu, R. Srihari, Feature selection for text categorization on imbalanced data, SIGKDD Explor. 6 (1) (2004) 80–89.
[44] Y. Zhu, X. Shen, W. Pan, Simultaneous grouping pursuit and feature selection over an undirected graph, J. Am. Stat. Assoc. 108 (502) (2013) 713–725.