Applied Soft Computing 67 (2018) 94–105

Contents lists available at ScienceDirect

Applied Soft Computing

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

Dealing with high-dimensional class-imbalanced datasets: Embedded feature selection for SVM classification

Sebastián Maldonado [a,*], Julio López [b]

[a] Facultad de Ingenierí a y Ciencias Aplicadas, Universidad de los Andes, Monseñor Álvaro del Portillo 12455, Las Condes, Santiago, Chile
[b] Facultad de Ingenierí a y Ciencias, Universidad Diego Portales, Ejército 441, Santiago, Chile

ARTICLE INFO

Article history:
Received 9 August 2016
Received in revised form 26 February 2018
Accepted 28 February 2018
Available online 5 March 2018

Keywords:
Feature selection
Support Vector Data Description
Cost-sensitive learning
Embedded approaches
Imbalanced data classification

ABSTRACT

In this work, we propose a novel feature selection approach designed to deal with two major issues in machine learning, namely class-imbalance and high dimensionality. The proposed embedded strategy penalizes the cardinality of the feature set via the scaling factors technique, and is used with two support vector machine (SVM) formulations designed to deal with the class-imbalanced problem, namely Cost Sensitive SVM, and Support Vector Data Description. The proposed concave formulations are solved via a Quasi-Newton update and Armijo line search. We performed experiments on 12 highly imbalanced microarray datasets using linear and Gaussian kernel, achieving the highest average predictive performance with our approach compared with the most well-known feature selection strategies.

1. Introduction

Feature selection is an important machine learning topic, especially when facing class-imbalanced datasets [1,2]. Selecting the relevant attributes improves the model's generalization ability and reduces the risk of overfitting, which is higher in high-dimensional domains [3,4].

Support vector machine (SVM) [5] is an effective supervized method with appealing advantages such as adequate generalization to new instances thanks to the *structural risk minimization principle*, a single optimal solution, and a representation that depends on few data points. However, SVM neither determines the variables' importance nor is it designed to deal with the class-imbalance problem [6,7].

In this work, we propose a novel embedded strategy for SVM classification and feature selection which penalizes the use of features via a concave approximation of the cardinality of the scaling factors. The approach is based on two SVM models for skewed class distribution: Support Vector Data Description (SVDD) [8] and Cost-Sensitive SVM (CS-SVM) [9]. The formulations are solved via a two-step iterative strategy: first, SVDD or CS-SVM is solved to obtain a linear or nonlinear classifier without feature penalization, while a concave problem for feature penalization is subsequently defined and solved by updating the scaling factors using a Quasi-Newton strategy and Armijo line search. The main contribution of our method is the novel embedded feature selection formulation specially tailored for dealing with high-dimensionality under a class-imbalance condition. These two issues have received increasing attention in the machine learning literature, but they have not been studied together under an embedded feature selection framework based on feature penalization, to the best of our knowledge. A second contribution is the state-of-the-art optimization scheme; a projected Quasi-Newton gradient descent strategy (Powell's damped BFGS secant) for efficient convergence.

This paper is structured as follows. Section 2 introduces the class-imbalance problem and presents the two SVM-based formulations that are relevant for this work. Recent developments for feature selection are reviewed in Section 3. The proposed feature selection approach is 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 also address future developments.

2. The imbalanced data classification problem

The class-imbalance problem arises when datasets exhibit significant, and in some cases extreme, imbalances. When this situation occurs, traditional classifiers such as SVM tend to generate a trivial model by predicting every instance in the majority class [10]. This issue has been considered one of the main trends in data mining in the last decade [11]. Four families of methods have

* Corresponding author.
E-mail addresses: smaldonado@uandes.cl (S. Maldonado), julio.lopez@udp.cl (J. López).

https://doi.org/10.1016/j.asoc.2018.02.051
1568-4946/

been proposed to treat this problem: Resampling, Cost-Sensitive Learning, One Class Learning, and Feature Selection [12,6]. Next we briefly describe the first three strategies, while feature selection is subsequently described in Section 3.

2.1. Data resampling

Data resampling rebalances a dataset artificially by constructing a training set in which all classes can be shattered adequately by standard classification approaches. The two most intuitive resampling approaches are *random undersampling* and *random oversampling*. While the first approach discards examples from the majority class randomly, the latter duplicates randomly selected instances of the minority class [12,13].

Some proposed intelligent oversampling methods can be found in the literature. Instead of simply duplicating cases from the minority class, SMOTE [14] generates new examples by interpolating the pre-existing minority examples.

2.2. Cost-Sensitive learning and CS-SVM

A disadvantage of data resampling is that either approach described above removes potentially relevant information (undersampling) or adds new artificially-generated information that may lead to overfitting and/or increase running times [15]. To avoid such problems, classification methods can also be trained from imbalanced datasets directly via cost-sensitive learning.

Cost-sensitive techniques incorporate misclassification costs in the training process. We define C_- as the cost of misclassifying a majority class example as a minority one, and C_+ as the cost of misclassification in the target class, which is usually higher, i.e., $C_+ > C_-$.

There are several cost-sensitive learning approaches. Some of them include cost-sensitive adjustment of different classification techniques, which can be applied to the decision threshold or by modifying their formulations [7]. For SVM, the total misclassification cost $C\sum_{i=1}^{m}\xi_i$ can be divided into two expressions, one for each class, leading to the Cost-Sensitive SVM (CS-SVM) formulation:

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

where I^+ and I^- correspond to the sets of positive and negative instances, respectively [9,16]. Some implementations, like the one used in this work, define $C_+ = C$ and $C_- = w_- C$, where $w_- = 1$ recovers the standard SVM formulation.

2.3. One-Class Learning

One-Class Learning techniques usually perform model training with only the target class in order to construct a *description* of the data [8,17]. Originally designed for outlier detection [18], some approaches have been proposed to deal with the class-imbalance problem. One of these approaches was developed by Tax and Duin, namely, Support Vector Data Description (SVDD) [8]. SVDD finds a sphere that contains most of the training points but with minimum volume. Similar to SVM, slack variables $\boldsymbol{\xi}$ are introduced to avoid large spheres that may not represent the data very well. The formulation of SVDD in its dual version follows:

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

where $K : \Re^n \times \Re^n \to \Re$ is a kernel function satisfying Mercer's condition [19].

Problem (2) is equivalent to minimizing the distance from the center of the sphere to any of the support vectors on the boundary while assuring that almost all objects are in the sphere. When examples of the minority class are available, which is the case in class-imbalance classification, they can be incorporated into the formulation in a similar fashion as CS-SVM: the total cost $C\sum_{i=1}^{m}\xi_i$ is divided into two expressions, one for each class, requiring that most outliers (in our case the elements of the minority class) fall outside the boundary defined by the sphere.

The parameter C is user-defined and controls the trade-off between the number of data points included in the sphere and its volume. This parameter is defined as $C = \frac{1}{mr}$, where m is the total number of training points and r the proportion of outliers in the solution, which is usually set via cross-validation in the model selection process [6,8].

3. Feature Selection for SVM under class-imbalance

In this section, we discuss recent developments for feature selection and class-imbalanced classification, describing the feature selection methods that are relevant for this work.

Resampling techniques and/or cost-sensitive learning have been used jointly with filter methods for feature selection in class-imbalanced datasets [20–23]. Some filter approaches do not require any adaptation to be suitable for the class-imbalance problem. One example is the Fisher Score which computes the means and standard deviations for both classes independently, and, therefore, does not become negatively affected when the class distribution is too skewed [24]. This measure has the following form:

$$F(j) = \left|\frac{\mu_j^+ - \mu_j^-}{(\sigma_j^+)^2 + (\sigma_j^-)^2}\right|, \tag{3}$$

where μ_j^+ (μ_j^-) is the mean for the jth variable in the positive (negative) class and σ_j^+ (σ_j^-) is the respective standard deviation. This approach can be used before the learning task to identify and select the relevant variables, while resampling and/or cost-sensitive learning can be used subsequently for class-imbalance classification. Alternatively, some filter methods have been proposed to deal with the class-imbalance problem. FAST [25] filters out irrelevant variables by computing the AUC of each attribute, removing those with values close to 0.5. DBFS [26] follows a similar idea, although it uses Information Gain as the contribution measure instead of AUC.

Recursive Feature Elimination SVM (RFE-SVM) is a popular SVM-based approach proposed by Guyon et al. [27], in which a backward elimination procedure is performed to remove those variables whose elimination leads to the largest margin of class separation. This approach can be easily adapted for class-imbalance classification by using CS-SVM (Formulation (1)) instead of the traditional soft-margin SVM.

Following the notation used by Song et al. [28], we denote the set of available features by $\mathcal{S}$, initialized as the full set of variables, and

the subset of variables to be removed from $\mathcal{S}$ at each iteration by $\mathcal{I}$. The final number of selected attributes r needs to be predefined *a priori*; i.e., the stopping criterion is $|\mathcal{S}| = r$. The RFE-SVM strategy is presented in Algorithm 1.

Algorithm 1. Recursive Feature Elimination SVM Algorithm

repeat
  $\boldsymbol{\alpha} \longleftarrow$ SVM Training on $\mathcal{S}$
  $\mathcal{I} \longleftarrow \underset{\mathcal{I}}{\text{argmin}} \sum_{p\in\mathcal{I}} |W^2(\boldsymbol{\alpha}) - W^2_{(-p)}(\boldsymbol{\alpha})|, \quad \mathcal{I} \subset \mathcal{S}$
  $\mathcal{S} \longleftarrow \mathcal{S} \backslash \mathcal{I}$
until $|\mathcal{S}| = r$.

At each iteration, the RFE-SVM algorithm eliminates a subset $\mathcal{I}$ of variables whose removal minimizes the variation of $W^2(\boldsymbol{\alpha})$. This measure corresponds to the Euclidean norm of the weight vector $\mathbf{w}$ written in terms of the dual variables of the SVM model, and is inversely proportional to the margin of the classifier. The measure $W^2_{(-p)}(\boldsymbol{\alpha})$ is equivalent to $W^2(\boldsymbol{\alpha})$ with the only difference being that attribute p is removed from each training object. The contribution measure $W^2(\boldsymbol{\alpha})$ is given by:

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

The RFE-SVM method can be also extended to Support Vector Data Description [29]. In this work, the authors developed two contribution metrics for the RFE-SVM algorithm based on SVDD: the size of the boundary (SVDD-Radius-RFE Criterion Function) and the objective function of Problem (2) (SVDD-Dual-Objective-RFE Criterion Function).

A feature selection strategy was proposed in [6] to deal with the class-imbalance problem during the elimination process. Similar to RFE-SVM, the BFE-SVM$_{bl}$ algorithm follows a backward elimination strategy, but the balanced loss function is used as contribution measure instead of the margin. The balanced loss computes the weighted average of Type I and Type II errors. Let $\mathcal{T}^+$ and $\mathcal{T}^-$ be the training sets containing the positive and negative instances, respectively. The *balanced loss* for a given feature $j \in \mathcal{S}$, is defined as:

$$\begin{aligned} \text{LOSS}_{bl}((\boldsymbol{\alpha}, b), \mathcal{S}\backslash\{j\}, \mathcal{T}) \; = & \frac{\sum_{s\in\mathcal{T}^-} \left| y_s - sgn\left(\sum_{i\in\mathcal{T}^-} \alpha_i y_i K(\mathbf{x}_i^{(-j)}, \mathbf{x}_s^{(-j)}) + b\right)\right|}{|T^-|} \\ & + \frac{\sum_{s\in\mathcal{T}^+} \left| y_s - sgn\left(\sum_{i\in\mathcal{T}^+} \alpha_i y_i K(\mathbf{x}_i^{(-j)}, \mathbf{x}_s^{(-j)}) + b\right)\right|}{|T^+|}, \end{aligned} \tag{5}$$

where $|T^+|$ and $|T^-|$ represent the number of positive and negative examples, respectively. The Backward Feature Elimination SVM algorithm based on balanced loss is described in Algorithm 2.

Algorithm 2. BFE-SVM Algorithm based on balanced loss

repeat
  $\boldsymbol{\alpha} \longleftarrow$ SVM Training on $\mathcal{S}$
  $\mathcal{I} \longleftarrow \underset{\mathcal{I}}{\text{argmin}} \sum_{j\in\mathcal{I}} \text{LOSS}_{bl}(\boldsymbol{\alpha}, \mathcal{S}\backslash\{j\}), \quad \mathcal{I} \subset \mathcal{S}$
  $\mathcal{S} \longleftarrow \mathcal{S} \backslash \mathcal{I}$
until $|\mathcal{S}| = r$

Finally, an important feature selection method is the use of the l_1-norm or LASSO penalty instead of the l_2-norm used in the standard SVM formulation. This modification provides good compromise between predictive performance and sparsity [30]. This method can be extended to cost-sensitive learning by replacing the Euclidean norm by the LASSO penalty in the CS-SVM formulation. In this work, we use the l_1-SVM formulation implemented in the LIBLINEAR toolbox [31], which uses the squared loss function instead of the traditional hinge loss, as follows:

$$\begin{aligned} \min_{\mathbf{w},b,\boldsymbol{\xi}} \quad & \|\mathbf{w}\|_1 + C_+ \sum_{i\in I^+} \xi_i^2 + C_- \sum_{i\in I^-} \xi_i^2 \\ \text{s.t.} \quad & y_i \cdot (\mathbf{w}^\top \mathbf{x}_i + b) \geq 1 - \xi_i, \quad i = 1, \ldots, m, \\ & \xi_i \geq 0, \quad i = 1, \ldots, m, \end{aligned} \tag{6}$$

where $\|\cdot\|_1$ denotes the l_1-norm.

The approaches Fisher Score, RFE-SVM, and BFE-SVM$_{bl}$, and l_1-SVM are used as alternative strategies for benchmarking purposes, in combination with CS-SVM and the resampling technique SMOTE. As in our proposal, the first three strategies can be used with kernel-based SVM, unlike most embedded approaches, and therefore were chosen for comparison. The l_1-SVM is also studied empirically for completeness.

4. Proposed feature selection method for class-imbalance datasets

In this section, an SVM-based approach for solving the class-imbalance problem in high dimensional domains is presented. The reasoning behind this approach is that we can improve predictive performance of SVM formulations designed to deal with highly imbalanced data by eliminating those features that are irrelevant for the solution. We extend the ideas of KP-SVM [4] to SVDD and CS-SVM, improving the optimization scheme presented in that work.

The methods proposed here attempt to optimize the scaling factors incorporated in SVDD and CS-SVM while penalizing their cardinality of non-zero elements via a concave approximation of the zero norm. The anisotropic Gaussian kernel is introduced Section 4.1. The penalized formulations for SVDD and CS-SVM are presented in Sections 4.2 and 4.3, respectively. The proposed Quasi-Newton-based optimization scheme for these formulations is presented in Section 4.4.

4.1. The anisotropic Gaussian kernel

A kernel-based approach for feature penalization (Kernel-Penalized SVM or KP-SVM) was proposed in [4]. Following a strategy similar to the use of scaling factors [32–34], the widths of an anisotropic Gaussian kernel $\boldsymbol{\sigma}$ are updated via successive gradient steps. This kernel function has the following form:

$$K(\boldsymbol{\sigma} * \mathbf{x}_i, \boldsymbol{\sigma} * \mathbf{x}_s) = \exp\left(-\frac{||\boldsymbol{\sigma} * \mathbf{x}_i - \boldsymbol{\sigma} * \mathbf{x}_s||^2}{2}\right), \tag{7}$$

where $*$ is the component-wise vector product operator, which is defined as $\mathbf{a} * \mathbf{b} = (a_1 b_1, \ldots, a_n b_n)$ for two vectors $\mathbf{a}$ and $\mathbf{b}$ in $\Re^n$. Originally developed for the standard soft-margin SVM formulation, the KP-SVM method is extended to CS-SVM and SVDD in this work, while the optimization scheme is improved via Quasi-Newton and Armijo line search.

4.2. Kernel-Penalized SVDD formulation

In order to construct a penalized formulation for SVDD, we introduce the vector of nonnegative scaling factors $\boldsymbol{\sigma}$ to Formulation (2) and add a penalization function $f(\boldsymbol{\sigma})$, which represents a concave approximation of the cardinality of the non-zero scaling factors, also known as the zero "norm". Notice that this is not a norm because the triangle inequality does not hold [30]. Based on

the approximation proposed by Bradley and Mangasarian [30], this function has the following form:

$$f(\boldsymbol{\sigma}) = \mathbf{e}^T(\mathbf{e} - \exp(-\beta\boldsymbol{\sigma}) = \sum_{j=1}^{n}[1 - \exp\left(-\beta\sigma_j\right)], \tag{8}$$

where $\beta = 5$ is suggested in the literature [30] and $\mathbf{e}$ is a vector of ones with the adequate dimension. It has been proven that the smooth function (8) leads to an exact solution for the zero "norm" for finite values of β [30]. A sensitivity analysis has been performed for this parameter in some studies (see e.g. [4]), concluding that 5 is an adequate value for the approximation and leads to good empirical results, although these results are not strongly influenced by this parameter.

The KP-SVDD formulation follows:

$$\begin{aligned} \min_{\boldsymbol{\alpha},\boldsymbol{\sigma}} \quad & \sum_{i,s=1}^{m}\alpha_i\alpha_s K(\boldsymbol{\sigma} * \mathbf{x}_i, \boldsymbol{\sigma} * \mathbf{x}_s) + C_2 f(\boldsymbol{\sigma}) \\ \text{s.t.} \quad & \sum_{i=1}^{m}\alpha_i = 1, \\ & 0 \le \alpha_i \le C, \quad i = 1, \ldots, m, \\ & \sigma_j \ge 0, \quad j = 1, \ldots, n, \end{aligned} \tag{9}$$

with $C_2 > 0$. Notice that, for a Gaussian kernel, we have $\sum_{i=1}^{m}\alpha_i K(\mathbf{x}_i, \mathbf{x}_i) = 0$. This implies that the first term of Formulation (2) can be removed, leading to a minimization problem. In this work we use only Gaussian kernels for SVDD.

4.3. Kernel-Penalized CS-SVM formulation

Before presenting the Kernel-Penalized CS-SVM formulation, we first describe the kernel version of CS-SVM, which results from deriving the dual of Problem (1):

$$\begin{aligned} \max_{\boldsymbol{\alpha}} \quad & \sum_{i=1}^{m}\alpha_i - \frac{1}{2}\sum_{i,s=1}^{m}\alpha_i\alpha_s y_i y_s K(\mathbf{x}_i, \mathbf{x}_s) \\ \text{s.t.} \quad & \sum_{i=1}^{m}\alpha_i y_i = 0, \\ & 0 \le \alpha_i \le C_+, \quad i \in I^+, \\ & 0 \le \alpha_i \le C_-, \quad i \in I^-. \end{aligned} \tag{10}$$

Note that the previous formulation is equivalent to the standard soft-margin SVM, with the only difference being that the vector of dual variables $\boldsymbol{\alpha}$ is now upper-bounded by C_+ and C_- for the positive and negative instances, respectively. Our proposal involves the inclusion of scaling factors $\boldsymbol{\sigma}$ and the penalization function $f(\boldsymbol{\sigma})$ presented in Eq. (8) to Formulation (10). The Kernel-penalized CS-SVM formulation (KP-CSSVM) follows:

$$\begin{aligned} \max_{\boldsymbol{\alpha},\boldsymbol{\sigma}} \quad & \sum_{i=1}^{m}\alpha_i - \frac{1}{2}\sum_{i,s=1}^{m}\alpha_i\alpha_s y_i y_s K(\boldsymbol{\sigma} * \mathbf{x}_i, \boldsymbol{\sigma} * \mathbf{x}_s) - C_2 f(\boldsymbol{\sigma}) \\ \text{s.t.} \quad & \sum_{i=1}^{m}\alpha_i y_i = 0, \\ & 0 \le \alpha_i \le C_+, \quad i \in I^+, \\ & 0 \le \alpha_i \le C_-, \quad i \in I^-, \\ & \sigma_j \ge 0, \quad j = 1, \ldots, n. \end{aligned} \tag{11}$$

For the Kernel-penalized CS-SVM model, we explored two types of kernels: the anisotropic Gaussian kernel, as presented in Eq. (7), and the linear kernel that includes the scaling factors. The latter has the following form:

$$K(\boldsymbol{\sigma} * \mathbf{x}_i, \boldsymbol{\sigma} * \mathbf{x}_s) = \langle \boldsymbol{\sigma} * \mathbf{x}_i, \boldsymbol{\sigma} * \mathbf{x}_s \rangle. \tag{12}$$

4.4. The two-step Quasi-Newton optimization scheme

Instead of solving KP-SVDD (Problem (9)) and KP-CSSVM (Problem (11)) directly, an iterative algorithm is proposed to avoid facing the non-linearity that results with the introduction of the nonnegative scaling factors. The two-step process consists of optimizing $\boldsymbol{\alpha}$ and $\boldsymbol{\sigma}$ independently, fixing one of the variables at a time.

In the first step the algorithm, CS-SVM or SVDD are trained for a given anisotropic kernel width $\boldsymbol{\sigma}^*$. Notice that if the kernel widths are fixed, then the penalty function does not depend on $\boldsymbol{\alpha}$ and therefore can be excluded from this step. Since the kernel matrix does not include decision variables at this point, we recover the original CS-SVM or SVDD formulations. These problems can be solved efficiently with any of the state-of-the-art solvers available. The output of this step is the solution vector $\boldsymbol{\alpha}^*$. The KP-SVDD and KP-CSSVM problems for this first step follow:

$$\begin{aligned} \min_{\boldsymbol{\alpha}} \quad & \sum_{i,s=1}^{m}\alpha_i\alpha_s K(\boldsymbol{\sigma}^* * \mathbf{x}_i, \boldsymbol{\sigma}^* * \mathbf{x}_s) \\ \text{s.t.} \quad & \sum_{i=1}^{m}\alpha_i = 1, \\ & 0 \le \alpha_i \le C, \quad i = 1, \ldots, m, \end{aligned} \tag{13}$$

and

$$\begin{aligned} \max_{\boldsymbol{\alpha}} \quad & \sum_{i=1}^{m}\alpha_i - \frac{1}{2}\sum_{i,s=1}^{m}\alpha_i\alpha_s y_i y_s K(\boldsymbol{\sigma}^* * \mathbf{x}_i, \boldsymbol{\sigma}^* * \mathbf{x}_s) \\ \text{s.t.} \quad & \sum_{i=1}^{m}\alpha_i y_i = 0, \\ & 0 \le \alpha_i \le C_+, \quad i \in I^+, \\ & 0 \le \alpha_i \le C_-, \quad i \in I^-. \end{aligned} \tag{14}$$

For the second step we optimize $\boldsymbol{\sigma}$ for a given solution $\boldsymbol{\alpha}^*$. Fixing $\boldsymbol{\alpha}^*$ results in a non-linear problem with only positive constraints since all constraints in KP-SVDD (Problem (9)) and KP-CSSVM (Problem (11)) but the ones related to $\boldsymbol{\sigma}$ can be excluded, together with the first term of the objective function of KP-CSSVM. The KP-SVDD and KP-CSSVM formulations for the second step follow:

$$\begin{aligned} \min_{\boldsymbol{\sigma}} \quad & F(\boldsymbol{\sigma}) \\ \text{s.t.} \quad & \sigma_j \ge 0, \quad j = 1, \ldots, n, \end{aligned} \tag{15}$$

with

$$F(\boldsymbol{\sigma}) = \sum_{i,s=1}^{m}\alpha_i^*\alpha_s^* K(\boldsymbol{\sigma} * \mathbf{x}_i, \boldsymbol{\sigma} * \mathbf{x}_s) + C_2 f(\boldsymbol{\sigma}) \tag{16}$$

for KP-SVDD, and

$$F(\boldsymbol{\sigma}) = \frac{1}{2}\sum_{i,s=1}^{m}\alpha_i^*\alpha_s^* y_i y_s K(\boldsymbol{\sigma} * \mathbf{x}_i, \boldsymbol{\sigma} * \mathbf{x}_s) + C_2 f(\boldsymbol{\sigma}) \tag{17}$$

for KP-CSSVM, where $f(\boldsymbol{\sigma})$ is the penalty function proposed at the beginning of this section (cf. Eq. (8)). Notice that we cast KP-CSSVM into a minimization problem for this second step. In order to solve the previous problems efficiently, we follow a Quasi-Newton strategy with an orthogonal projection for variable $\boldsymbol{\sigma}$ to assure that its components remain positive. The Quasi-Newton procedure updates the kernel widths iteratively until one or more $\boldsymbol{\sigma}$ components fall below a predefined threshold ϵ. When this occurs, we

remove the variables j with $\sigma_j < \epsilon$ and return to step 1. The algorithm for scaling factor updating and feature elimination follows:

Algorithm 3. Scaling factor updating and feature elimination algorithm

1: $\boldsymbol{\sigma} \longleftarrow \sigma_0 \mathbf{e}$
2: **repeat**
3: $\boldsymbol{\alpha} \longleftarrow$ CS-SVM or SVDD Training on $\mathcal{S}$ **(step 1)**
4: $B \longleftarrow I$
5: **repeat (step 2)**
6: $\mathbf{d} \longleftarrow P_{\Re^n_+}(\boldsymbol{\sigma} - B^{-1}\nabla F(\boldsymbol{\sigma})) - \boldsymbol{\sigma}$
7: Update λ via the Armijo rule
8: $\boldsymbol{\sigma} \longleftarrow \boldsymbol{\sigma} + \lambda \mathbf{d}$
9: Update B via the Quasi-Newton rule
10: **until** $\exists j \mid \sigma_j < \epsilon$
11: $\mathcal{I} \longleftarrow \{j \in \mathcal{S} | \sigma_j < \epsilon\}$
12: $\mathcal{S} \longleftarrow \mathcal{S} \backslash \mathcal{I}$
13: **until** $|\mathcal{S}| = r$

The algorithm initializes with an isotropic kernel width obtained from a predefined value $\boldsymbol{\sigma} = \sigma_0 \mathbf{e}$. The value for σ_0 can be obtained via cross-validation and kernel-based SVM using the full set of variables (Model Validation 1).

After CS-SVM or SVDD is trained on $\mathcal{S}$ (step 1), the Quasi-Newton procedure (step 2) is started by initializing a matrix B as the identity matrix I. This matrix is updated iteratively following the *Quasi-Newton rule* (line 9 of Algorithm 3). The implementation of updating rules for B satisfying the assumption of positive definiteness is of major importance for obtaining a reasonably efficient algorithm for practical applications. In some specific convex problems the use of the Hessian matrix could both guarantee the validity of the above assumption, as well as the quick convergence to the stationary point. However, the Hessian matrix is not positive definite in general non-linear problems, and computation of second order derivatives is usually too expensive in terms of the number of operations and the computation time in most engineering applications. In these cases the use of Quasi-Newton rules is a standard approach that provides positive definite matrices from the knowledge of just the first order derivatives. For example, the damped BFGS secant update proposed by Powell in [35] (see also [36]) was designed to circumvent the lack of positive definiteness in the Hessian matrix at the solution. This update procedure has the following form:

$$B^{k+1} = B^k - \frac{B^k \mathbf{p}^k (\mathbf{p}^k)^\top B^k}{(\mathbf{p}^k)^\top B^k \mathbf{p}^k} + \frac{\mathbf{r}^k (\mathbf{r}^k)^\top}{(\mathbf{p}^k)^\top \mathbf{r}^k}, \tag{18}$$

where

$$\mathbf{p}^k = \boldsymbol{\sigma}^{k+1} - \boldsymbol{\sigma}^k,$$

$$\mathbf{r}^k = \theta^k \mathbf{q}^k + (1 - \theta^k) B^k \mathbf{p}^k,$$

$$\mathbf{q}^k = \nabla F(\boldsymbol{\sigma}^{k+1}) - \nabla F(\boldsymbol{\sigma}^k),$$

$$\theta^k = \begin{cases} 1, & \text{if } (\mathbf{p}^k)^\top \mathbf{q}^k \geq \eta (\mathbf{p}^k)^\top B^k \mathbf{p}^k, \\ \dfrac{(1-\eta)(\mathbf{p}^k)^\top B^k \mathbf{p}^k}{(\mathbf{p}^k)^\top B^k \mathbf{p}^k - (\mathbf{p}^k)^\top \mathbf{q}^k}, & > \text{if } (\mathbf{p}^k)^\top \mathbf{q}^k < \eta (\mathbf{p}^k)^\top B^k \mathbf{p}^k, \end{cases}$$

for some $\eta \in (0, 1)$. Moreover, Powell's damped BFGS secant has proved to be very successful computationally (see e.g. [37]). Hence, we choose this update, which preserves the positive definiteness of the matrix B_k even far away from the solution. We set $\eta = 0.2$, as suggested in [35].

In line 6 of Algorithm 3, the descent direction $\mathbf{d}$ of the function $F(\boldsymbol{\sigma})$ is computed from B and $\nabla F(\boldsymbol{\sigma})$, the gradient of $F(\boldsymbol{\sigma})$. For KP-SVDD and a given feature j, the partial derivative of $F(\boldsymbol{\sigma})$ with respect to σ_j is given by

$$\frac{\partial F(\boldsymbol{\sigma})}{\partial \sigma_j} = -\sum_{i,s=1}^{m} \sigma_j (x_{ij} - x_{sj})^2 \alpha_i^* \alpha_s^* K(\boldsymbol{\sigma} * \mathbf{x}_i, \boldsymbol{\sigma} * \mathbf{x}_s) + C_2 \beta \exp(-\beta \sigma_j). \tag{19}$$

For KP-CSSVM, we distinguish two cases: the linear kernel and the Gaussian kernel. For the first case, the partial derivative of $F(\boldsymbol{\sigma})$ with respect to σ_j is given by

$$\frac{\partial F(\boldsymbol{\sigma})}{\partial \sigma_j} = \sum_{i,s=1}^{m} \sigma_j \alpha_i^* \alpha_s^* y_i y_s x_{ij} x_{sj} + C_2 \beta \exp(-\beta \sigma_j), \tag{20}$$

while the partial derivative of $F(\boldsymbol{\sigma})$ with respect to σ_j for the Gaussian kernel follows:

$$\frac{\partial F(\boldsymbol{\sigma})}{\partial \sigma_j} = -\frac{1}{2} \sum_{i,s=1}^{m} \sigma_j (x_{ij} - x_{sj})^2 \alpha_i^* \alpha_s^* y_i y_s K(\boldsymbol{\sigma} * \mathbf{x}_i, \boldsymbol{\sigma} * \mathbf{x}_s) + C_2 \beta \exp(-\beta \sigma_j). \tag{21}$$

The term $P_{\Re^n_+}$ in line 6 of Algorithm 3 denotes the orthogonal projection on the nonnegative orthant $\Re^n_+$, and is defined by $P_{\Re^n_+}(\mathbf{x}) = \max(\mathbf{0}, \mathbf{x})$, for $\mathbf{x} \in \Re^n$. This is important since it assures that the components of $\boldsymbol{\sigma}$ remain positive, as imposed in Formulation (15) from the second step of the proposed algorithm. The inclusion of this orthogonal projection allows us the use of unconstrained non-linear optimization techniques for optimizing $F(\boldsymbol{\sigma})$.

The eighth line of Algorithm 3 corresponds to the updating rule for $\boldsymbol{\sigma}$. The descent parameter λ is found using the Armijo rule (line 7 of Algorithm 3). This rule consists of choosing the step size parameter λ_k that satisfies the following inequality:

$$F(\boldsymbol{\sigma}_k + \lambda_k \mathbf{d}_k) \leq F(\boldsymbol{\sigma}_k) + \nu \lambda_k \nabla F(\boldsymbol{\sigma}_k)^\top \mathbf{d}_k, \tag{22}$$

with fixed $\nu \in (0, 1)$.

Lines 10–12 of Algorithm 3 describe the backward feature elimination strategy. The Quasi-Newton approach will adjust the scaling factors iteratively until one or more components of $\boldsymbol{\sigma}$ fall below a threshold ϵ (line 10). When this occurs, a set of attributes $\mathcal{I}$ to remove from $\mathcal{S}$ is constructed with all the variables j with $\sigma_j < \epsilon$ (line 11). We then remove $\mathcal{I}$ from $\mathcal{S}$ (line 12).

Line 13 of Algorithm 3 provides the stopping criterion. The algorithm stops when a predefined number of selected features r is found.

Next, the flow chart related to the kernel-penalized algorithm is presented in Figs. 1 and 2 On the one hand, Fig. 1 presents the interaction between the two validation processes: Model Validation 1, which involves the model selection for SVDD or CSSVM using all available information; and Model Validation 2, which is performed with Algorithm 3 to determine the parameters related to the method: descent parameter λ and threshold parameter ϵ. On the other hand, Fig. 2 details the various steps of Algorithm 3, which corresponds to the KP-SVDD or KP-CSSVM training.

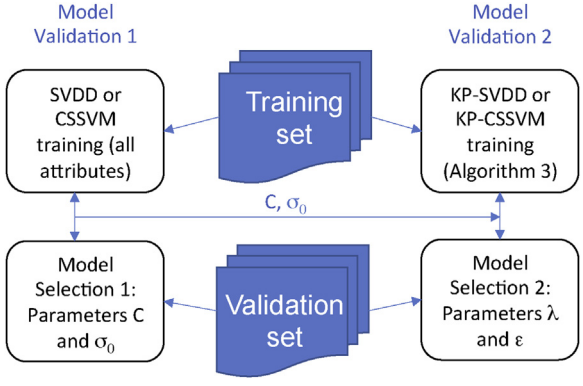

Fig. 1. Flow chart of the two validation processes required for the proposed methods.

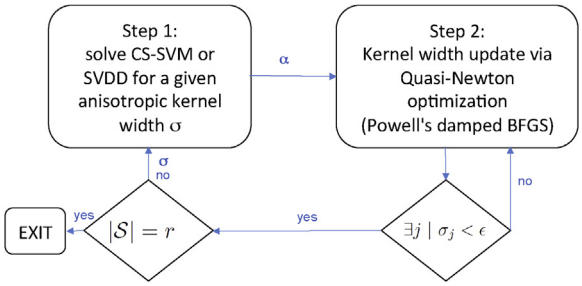

Fig. 2. Flow chart of the KP-SVDD or KP-CSSVM training algorithm.

5. Experimental results

We applied the proposed feature selection approaches KP-SVDD and KP-CSSVM to 12 microarray data sets for binary classification, comparing them with well-known alternative feature methods. In Section 5.1, we provide a description of the data sets and the experimental settings. Section 5.2 provides a summary of the performance obtained for all the different SVM approaches studied before applying feature selection. Section 5.3 summarized the performance obtained by our proposals and the benchmark feature selection methods. The detailed results for all methods and datasets are presented in Appendix A, in which the different AUC values are graphically represented for all studied subsets of n variables. Finally, the running times for all feature selection methods are reported and discussed in Section 5.4.

5.1. Datasets and experimental settings

The proposed embedded methods for feature selection and class-imbalance classification were applied on 12 two-class microarray datasets: Gordon's Lung Cancer (GORDON) [38], the GLIOMA dataset [39], the SRBCT dataset [40], Bhatacharjee's Lung Cancer (BHAT1, BHAT2, BHAT3, and BHAT4) [41], the CAR dataset (CAR1, CAR2, CAR3 and CAR4) [42], and Bullinger's Adult Acute Myeloid Leukemia (BULL) [43]. LUNG and BULL are binary classification problems, and therefore they were used without aggregating the classes. For both datasets, the minority class corresponds to malignant tissue samples and the majority class to benign samples. In contrast, GLIOMA, SRBCT, BHAT, and CAR are multi-class problems that were adapted for class-imbalance binary classification by aggregating all classes but one. The minority class is *cancer oligodendrogliomas*, *Burkitt lymphoma*, *pulmonary carcinoids*, *normal lung*, *squamous cell lung carcinomas*, *small-cell lung carcinomas*, *kidney cancer*, *gastroesophageal cancer*, *liver cancer*, and *pancreas cancer* for GLIOMA, SRBCT, BHAT1, BHAT2, BHAT3, BHAT4, CAR1, CAR2, CAR3, and CAR4, respectively.

The datasets presented have already been used for benchmark feature selection algorithms (see e.g. [6,44]). Table 1 summarizes the relevant meta-data for each benchmark dataset (the number of variables, the sample size, the percentage of observations in each class, and the imbalance ratio (IR)). Additionally, the class overlap is studied for each dataset with the Generalized Fisher Ratio(GFR) (see e.g. [45]). This measure computes the separation between the classes for all variables, and is defined as:

$$F_{gen} = \frac{\sum_{k=1}^{C} n_k \delta(m, m_k)}{\sum_{k=1}^{C} \sum_{i=1}^{n_k} \delta(x_i^k, m_k)}, \tag{23}$$

where n_k denotes the number of examples of class k, m is the global mean, m_k is the mean of class k, x_i^k represents instance i from class k, and δ is a distance measure (in our case the Euclidean norm).

From Table 1, we can observe that our datasets are very diverse in terms of number of variables (from 2308 to 17,404), imbalance ratio (from 4.85 to 32.83) and class overlap (from 0.247 (very low overlap) to 0.041 (high overlap)). The most challenging dataset is BULL, which has the highest overlap and class imbalance, while BHAT1 is linearly separable.

Table 1
Number of features, number of examples, percentage of each class, imbalance ratio (IR), and generalized Fisher ratio (GFR) for all 12 datasets.

Dataset	#features	#examples	%class(min.,maj.)	IR	GFR
GORDON	12,533	181	(17.1,82.9)	4.85	0.180
GLIOMA	4434	50	(14.0,86.0)	6.14	0.234
SRBCT	2308	83	(13.3,86.7)	6.55	0.169
BHAT3	3312	203	(10.3,89.7)	8.67	0.075
BHAT1	3312	203	(9.9,90.1)	9.15	0.247
BHAT2	3312	203	(8.4,91.6)	10.94	0.076
CAR2	9182	174	(6.9,93.1)	13.50	0.137
CAR1	9182	174	(6.3,93.7)	14.82	0.086
BULL	17,404	94	(4.3,95.7)	22.50	0.041
CAR3	9182	174	(4.0,96.0)	23.86	0.146
CAR4	9182	174	(3.4,96.6)	28.00	0.048
BHAT4	3312	203	(3.0,97.0)	32.83	0.046

Together with our proposals, the Kernel-Penalized feature selection strategy for Support Vector Data Description and Cost-Sensitive SVM, the following feature selection methods were studied and have been reported for benchmarking purposes: Fisher Score, Recursive Feature Elimination, and BFE-SVM based on balanced loss. The choice of these approaches was based on their flexibility to be used with SVDD and CS-SVM in both linear and kernel-based versions. The methodological procedure follows:

- We first trained Support Vector Data Description and Cost-Sensitive SVM using all available variables. In this step we identified the best of the two approaches for a set of possible hyperparameters. We also explored whether or not SMOTE oversampling improves the results.
- For model selection, we followed the leave-one-out cross-validation (LOO) procedure, which is a common strategy for tumor prediction with DNA microarray data [46]. The following values were studied, using AUC (Area Under the Curve) as performance metric: $C, \sigma \in \{2^{-7}, 2^{-6}, \ldots, 2^{6}, 2^{7}\}$. For CS-SVM and l_1-CSSVM, we explored the following values for the weight parameter $w_- \in \{0.01, 0.1, 0.5, 1\}$ [6,47].
- For the best model selected in the previous step, feature selection was performed in the training set of the LOO validation procedure, ranking the features according to their respective contribution measures and monitoring the classification performance for the following number of variables: $n = \{20, 50, 100, 250, 500, 1000\}$. It is important to mention that our experimental setting compares three feature ranking approaches (Fisher, RFE-SVM, and BFE-SVM) and three embedded methods (l1-CSSVM, KP-CSSVM, KPSVDD). The techniques of the latter group can identify irrelevant variables automatically, although a threshold is necessary to decide which attributes are finally discarded. For these methods, the subsets are constructed based on the n largest weights (in magnitude) for l1-CSSVM, and the n largest values for the anisotropic kernel width $\boldsymbol{\sigma}$ for our proposals.
- We used the default parameters of a well-known implementation of the oversampling technique SMOTE, as developed in Chawla et al. [14]. As is suggested in that paper, the nearest neighbors were set to 5. We performed 400% oversampling (default option), which means four from the five nearest neighbors were chosen. SMOTE oversampling is always applied on the training set for each fold of the LOO procedure.
- For the Kernel Penalized approach, we set $\epsilon = 0.00001$. We redefined the parameter λ found using the Armijo rule (step 7 of Algorithm (3)) as $\lambda := 0.1\lambda$. The reasoning behind this is to guarantee small kernel width updates at the first iterations of the

Table 2
Predictive performance for all classification approaches. Six datasets.

	GORDON	GLIOMA	SRBCT	BHAT3	BHAT1	BHAT2
SVM_l	98.4	75.1	95.0	94.7	**100**	93.9
SVM_l + SO	98.4	75.1	95.0	94.7	**100**	93.9
SVM_G	98.4	77.4	95.4	94.7	**100**	93.9
SVM_G + SO	98.4	77.4	95.4	94.7	**100**	96.5
CS-SVM_l	98.4	83.7	99.3	94.7	**100**	93.9
CS-SVM_l + SO	98.4	83.7	99.3	94.7	**100**	93.9
CS-SVM_G	98.4	82.4	95.4	**96.0**	**100**	96.5
CS-SVM_G + SO	98.4	**88.4**	95.4	95.7	**100**	**98.9**
l_1-CSSVM	**99.7**	60.3	**100**	95.4	**100**	96.5
l_1-CSSVM + SO	99.3	57.8	**100**	94.7	**100**	96.5
$SVDD_G$	91.2	70.8	81.5	88.1	95.6	83.7

Table 3
Predictive performance for all classification approaches. Six datasets.

	CAR2	CAR1	BULL	CAR3	CAR4	BHAT4
SVM_l	83.3	90.9	50.0	85.7	83.3	91.7
SVM_l + SO	83.3	90.9	50.0	85.7	83.3	91.7
SVM_G	83.3	90.9	50.0	85.7	83.3	83.3
SVM_G + SO	83.3	90.9	50.0	85.7	83.3	83.3
CS-SVM_l	83.3	90.9	50.0	85.7	83.3	91.7
CS-SVM_l + SO	83.3	90.9	50.0	85.7	83.3	91.7
CS-SVM_G	83.3	90.9	50.0	85.7	83.3	**100**
CS-SVM_G + SO	83.3	90.9	**87.5**	85.7	90.5	**100**
l_1-CSSVM	83.3	**95.5**	50.0	85.7	**100**	**100**
l_1-CSSVM + SO	**91.6**	**95.5**	74.4	85.7	**100**	**100**
$SVDD_G$	82.3	73.3	50.0	**90.0**	79.8	53.7

Quasi-Newton strategy (step 8 of Algorithm (3)), similar to the learning rate parameter in a Neural Network [48]. We explored the following values for the trade-off parameter for feature penalization: $C_2 \in \{2^{-7}, 2^{-6}, \ldots, 2^6, 2^7\}$, where $C_2 = 0.125$ led to the best empirical results in general.

Regarding model implementation, we used LIBSVM [49], LIBLINEAR [31], and DDTOOLS [50] on Matlab for CS-SVM, l_1-CSSVM, and SVDD approaches, respectively.

5.2. Classification performance summary without feature selection

Tables 2 and 3 presents the performance in terms of AUC of all classification methods without performing feature selection, obtained by the validation procedure described above. We explored the following approaches: standard SVM (Formulation (10) with $w_- = 1$), CS-SVM (Formulation (10) with $w_- < 1$), l_1-CSSVM (Formulation (6) with $w_- < 1$), and SVDD. For standard SVM and CS-SVM we present the results using a linear or a Gaussian kernel, which is highlighted in Tables 2 and 3 with a subindex l or G, respectively. The l_1-CSSVM method is designed only as a linear method. Additionally, we report the results for these methods with SMOTE oversampling (denoted by +SO in Tables 2 and Table 3), and without any data resampling. The best technique in terms of AUC is written in bold type.

In Tables 2 and 3, we observe that the best results are usually achieved with CS-SVM, although standard SVM has similar performance in some cases. The SVDD method is outperformed by CS-SVM, being CAR3 the only exception, and seems to be less suitable than the latter for high-dimensional problems.

5.3. Feature selection performance summary

Tables 4 and 5 present a summary of the feature selection results, where the best classification approach is used jointly with the feature selection strategies mentioned above. We also incorporated our Kernel-Penalized strategy using SVDD, despite the fact that this model was usually outperformed by CS-SVM. The best SVM approach without feature selection is also included in Tables 4 and 5 (column Best no FS). The summary includes the average and the maximum performance for all subsets of variables n (a total of six AUC computations), while the optimal subset size n^* is also presented in Table 4. The highest AUC is highlighted in bold type. In case of ties in terms of AUC, we present the smallest n^*.

In Tables 4 and 5, we observe that the best predictive results were achieved using the proposed KP-CSSVM and the l_1-CSSVM method. In the case of KP-SVDD, the method is outperformed by the others due to its dependence on a suboptimal classifier, in this case the SVDD method. It can be also noticed from Table 4 that there are large deviations between some mean and maximum AUC values computed from the different subsets of features. This occurs because most several feature selection methods fail at identifying

Table 4
Performance for all feature selection approaches. Six datasets.

Dataset	Best no FS	Fisher	RFE	BFE_{bl}	l_1-CSSVM	KP-CSSVM	KP-SVDD
GORDON							
Mean	–	**98.8**	97.0	92.0	98.1	94.5	79.3
Max	99.7	**99.7**	96.8	98.4	98.4	98.4	87.4
n^*	12,533	**20**	50	1000	250	1000	1000
GLIOMA							
Mean	–	61.8	67.6	58.8	53.7	**76.9**	59.7
Max	88.4	81.4	83.7	73.3	61.1	**87.2**	67.9
n^*	4434	1000	1000	1000	20	**100**	500
SRBCT							
Mean	–	84.6	93.1	89.2	94.6	**95.3**	75.9
Max	100	100	97.9	99.3	100	**100**	83.6
n^*	2308	500	1000	1000	1000	**250**	50
BHAT3							
Mean	–	90.4	88.8	89.1	90.7	**94.9**	80.9
Max	96.0	92.6	89.9	92.6	92.6	**95.1**	87.8
n^*	3312	500	20	1000	50	**1000**	1000
BHAT1							
Mean	–	99.6	99.6	95.7	**99.8**	99.1	85.9
Max	100	100	100	97.5	100	**100**	93.1
n^*	3312	20	20	250	250	**50**	500
BHAT2							
Mean	–	96.4	93.6	**98.1**	97.5	97.2	75.8
Max	98.9	96.8	93.6	**99.5**	99.2	**99.5**	81.8
n^*	3312	100	20	**250**	100	**250**	1000

Table 5
Performance for all feature selection approaches. Six datasets.

Dataset	Best no FS	Fisher	RFE	BFE_{bl}	l_1-CSSVM	KP-CSSVM	KP-SVDD
CAR2							
Mean	–	83.3	78.9	81.9	**88.2**	86.1	74.5
Max	91.6	87.5	82.4	87.5	**94.6**	87.5	79.2
n^*	9182	250	1000	500	**250**	50	1000
CAR1							
Mean	–	90.9	90.9	87.7	**95.4**	90.0	63.2
Max	95.5	90.9	90.9	90.9	**100**	90.9	73
n^*	9182	20	20	50	**1000**	250	1000
BULL							
Mean	–	60.3	77.1	66.7	**86.4**	80.9	50.0
Max	87.5	62.5	87.5	87.5	87.5	**100**	50
n^*	17,404	250	500	500	10	**500**	20
CAR3							
Mean	–	85.7	86.7	82.6	**90.8**	88.8	85.5
Max	90	85.7	**92.9**	92.9	**92.9**	92.9	91.3
n^*	9182	20	**20**	1000	**20**	50	1000
CAR4							
Mean	–	91.7	88.1	69.8	**100**	95.5	90.2
Max	100	100	91.7	75	**100**	100	92.6
n^*	9182	250	20	1000	**20**	50	1000
BHAT4							
Mean	–	**99.9**	91.7	76.8	99.7	98.7	64.0
Max	100	**100**	91.7	83.3	**100**	100	92.6
n^*	3312	**20**	20	1000	**20**	250	1000

Table 6
Holm's post-hoc test for pairwise comparisons. Maximum performance.

Method	Mean rank	Mean AUC	p value	$\alpha/(k-i)$	Action
KP-CSSVM	2.08	95.95	–	–	Not reject
l_1-CSSVM	2.58	93.86	0.51	0.050	Not reject
Fisher + CSSVM	3.08	91.43	0.19	0.025	Not reject
BFE_{bl}-CSSVM	3.71	89.81	0.03	0.017	Not reject
RFE-CSSVM	4.04	91.58	0.01	0.013	Reject
KP-SVDD	5.50	81.69	0.00	0.01	Reject

the correct variables and therefore AUC drops quickly to 0.5. In other cases, the performance remains stable independent from the feature selection strategy used (see e.g. BHAT1 and CAR1, see Figs. A.4 and A.5 in Appendix A).

In the presence of class-imbalance, datasets which are harder to classify correctly (such as BULL or GLIOMA) may show a large deviation in performance since classifiers tend to favor the majority class when it is not possible to construct accurate classifiers, leading to poorly balanced performance (AUC close to 0.5). High dimensionality makes the problem worse if the feature selection methods fail at identifying the relevant variables, or too many variables are removed. For the GLIOMA dataset, for example (Fig. A.2 in Appendix A), most methods fail at selecting a subset of 100 relevant variables of the 4434 available, achieving an AUC of 0.5. The proposed KP-CSSVM, however, achieves the best performance. If 20 variables are selected, all methods fail at predicting accurately since too few variables are chosen, and important information is lost. A similar case can be observed for the BULL dataset.

In Tables 4 and 5, we can also observe that feature selection can improve predictive performance by improving the model's generalization ability. According to our experimental results, for five of 12 datasets feature selection improved predictive performance (CAR1, CAR2, CAR3, BHAT2, and BULL); results were similar in five datasets (GORDON, SRBCT, BHAT1, BHAT4, and CAR4); and only in two cases CS-SVM with all attributes had a slightly better predictive performance than feature selection approaches (GLIOMA and BHAT3). There is also evidence in the literature that supports this fact (see e.g. [27]).

From Tables 4 and 5, we can see that no method outperformed the others for all the studied datasets, and the differences in performance are rather small in some experiments. The overall performance was studied further by applying tests for statistical significance to the five feature selection methods mentioned before. As suggested by Demšar [51], a post-hoc test is performed in order to identify which methods outperform others statistically. In this work, the Holm's test proposed in [52] was used. This test constructs a Z statistic based on the mean ranks for each method. The method with the lowest average rank is set as the baseline, and then a pairwise comparison is performed between each the other methods and this approach (see [52] for more details).

The results for the Holm's test are presented in Table 6 for the maximum performance for all subsets of variables n. We decided to create a single test for the best performance of each method since it represents their expected performance when implemented. The various subsets of variables n are arbitrarily defined, and can be seen as an additional parameter to tune. We also believe that it is a fair comparison since we use the same subset size for all the methods.

In Table 6, we can see that the proposed KP-CSSVM, l_1-CSSVM, the Fisher Score, and BFE_{bl} outperform RFE-CSSVM and KP-SVDD using $\alpha = 0.05$. Although KP-CSSVM has the best overall performance, we conclude that no method outperforms the others statistically in terms of maximum AUC among all the subsets of variables.

Table 7
Average running times in seconds. All datasets.

	Fisher	RFE	BFE	l1-CSSVM	KP-CSSVM	KP-SVDD
GORDON	1."4	18."0	19."8	0."2	46."9	18."9
GLIOMA	0."5	0."5	1."0	0."0	1."8	3."1
SRBCT	0."3	0."8	0."9	0."0	1."9	1."1
BHAT3	0."4	1."2	1."6	0."1	28."5	2."1
BHAT1	0."4	0."4	0."8	0."1	2."5	2."0
BHAT2	0."4	1."1	1."1	0."1	3."2	2."2
CAR2	1."0	2."2	2."5	0."2	13."4	12."2
CAR1	1."0	1."9	2."0	0."2	12."5	12."1
BULL	1."8	2."4	2."5	0."2	34."2	38."4
CAR3	1."0	1."3	1."7	0."2	12."0	11."8
CAR4	1."0	2."0	1."7	0."2	11."7	11."5
BHAT4	0."4	0."6	0."8	0."1	2."3	2."2

5.4. Running time analysis

Finally, Table 7 provides a comparison for each feature selection method in terms of average running times using LOO cross-validation, and considering the best set of parameters obtained using the model selection procedure. The experiments were performed on an HP Envy dv6 with 16 GB RAM, 750 GB SSD, a i7-2620M processor with 2.70 GHz, and using Microsoft Windows 8.1 Operating System (64-bits). For all methods but l_1-CSSVM, we computed one step of the algorithm immediately after running the SVM training with all available attributes (the first ranking for methods Fisher, RFE and BFE_{bl}, and the quasi-Newton step and first kernel width update for our proposal), taking the average of all runs of the LOO cross-validation. For the l_1-CSSVM method, we compute its average training time.

On Table 7, we first observe that all running times are tractable and comparable. Although our method is slower due to the quasi-Newton step (the computation of the gradient is similar compared to the RFE and BFE algorithms in terms of complexity), all methods can construct the spinodal presented in Appendix A in less than five minutes for all datasets. The Limited-memory BFGS method (LBFGS) can be used instead of the damped BFGS method proposed by Powell [35] to solve large-scale problems and/or to improve running times (see [36] for more details about of LBFGS).

6. Conclusions

A novel embedded feature selection approach is presented in this work for SVM in the presence of the class-imbalance problem. The main idea is to extend KP-SVM [4], a framework for simultaneous classification and variable penalization via gradient descent, for dealing with a skewed class distribution.

Several modifications have been made to the KP-SVM method. First, we adapted the strategy for SVM formulations that are suitable for the class-imbalance issue, namely Support Vector Data Description, and Cost-sensitive SVM. Originally developed only for the Gaussian kernel, we also extended the KP-SVM method to linear kernel functions. Additionally, the optimization process has been improved significantly: a Quasi-Newton strategy is proposed to enhance convergence, speeding up the optimization of the kernel width variables; while the gradient parameter that controls the kernel width updating process λ is computed via Armijo search. These modifications allow adjusting the kernel width dynamically while the number of attributes decreases. The method leads to good predictive performance thanks to the use of baseline classifiers especially designed for imbalance classification, and the use of kernel methods provides a flexible framework for modeling complex non-linear patterns.

From the experimental work, we conclude that KP-CSSVM is an excellent alternative for feature selection and class-imbalance classification, since it is designed to deal with the latter issue

and penalizes the use of features by adjusting the kernel width in an iterative process. The Quasi-Newton methodology guarantees faster convergence than other well-known steepest descent approaches, like gradient descent. The Armijo search strategy also finds the adequate value for the kernel width updating process. In contrast, KP-SVDD is outperformed by other feature selection approaches based on CS-SVM. Although our proposal has positive results compared to SVDD using all available variables, the latter method performed worse than CS-SVM in our high-dimensional experiments, causing the inferior performance of our proposed method. Since SVDD is designed to perform outlier detection, it does not seem suitable for datasets where traditional discriminative approaches are able to perform very well (close to 100% accuracy). If the baseline method is not properly classifying, it is also expected that it also fails at identifying the relevant variables.

On the other hand, SMOTE has proved to be helpful in some cases, although in most experiments it has similar performance to no resampling. Oversampling was particularly useful for CS-SVM for the most overlapped datasets, namely GLIOMA and Bullinger Leukemia. Regarding the choice of the kernel, Gaussian kernel achieves best performance in seven out of 12 datasets, while the linear kernel worked better in the remaining five. This demonstrates the usefulness of the non-linear models, although superior performance cannot be guaranteed, and both types of functions should be explored.

The SVDD method, which usually leads to positive results when facing class-imbalance datasets (see e.g. [8,45]), did not perform as expected in this study when all attributes were used, probably because of the nature of the datasets chosen for this study (few samples, high-dimensionality, low noise/class overlap). Discriminative approaches such as CS-SVM seem to perform better when the datasets are easy to shatter.

Another important conclusion is that feature selection improves classification performance compared with using SVM with all features. AUC was improved or maintained in five out of six datasets (see Table 4 and Appendix A). But on the GLIOMA dataset, the best result for the best feature selection approach (KP-CSSVM using 100 attributes) is slightly worse compared with that obtained without feature selection. It is also important to notice that a gain in performance is not the only reason for performing feature selection: identifying models that perform well, as well as using all variables, leads to important insights for decision-making. For the GLIOMA dataset, in particular, it provides the 100 genes that have the highest impact on this type of cancer.

One reason that feature selection was not able to outperform SVM with all variables is that model selection was performed using only the full dataset; the SVM parameters are not tuned for every subset of n variables. Although the strategy followed in this work is standard and supported by the literature (see e.g. [4,47]), there are some parameters that are sensitive to varying the number of variables selected (for example, the kernel width, see [4]), which may cause a decrease in performance. There are several opportunities for future research in the following directions:

- It would be interesting to extend the proposed method to multi-class classification, where it is common to find high dimensional applications that exhibit skewed class distribution [44]. As an example, four of our six microarray datasets are multi-class problems that have this issue.
- The present work can also be useful in business analytics, where feature selection is usually performed to gain insight into the process that generates the data, e.g. to understand consumer behavior [53]. Some examples of business analytics applications that usually have to deal with class-imbalance are fraud detection, credit scoring, and churn prediction [53].
- Finally, there is a need for computationally efficient classification methods for high-dimensional tasks. We live in a Big Data era, and the size of the datasets is increasing dramatically in terms of instances as well as variables [53,3]. Out method can be extended to more efficient classification methods, such as linear-programming versions of SVM [54,55].

Acknowledgements

The first author was funded by FONDECYT projects 1160738 and 1140831. The second author was supported by FONDECYT project 1160894. This research was partially funded by the Complex Engineering Systems Institute, ISCI (ICM-FIC: P05-004-F, CONICYT: FB0816). The authors are grateful to Bing Zhu and the anonymous reviewers, who contributed to improving the quality of the paper.

Appendix A. Detailed results for all datasets

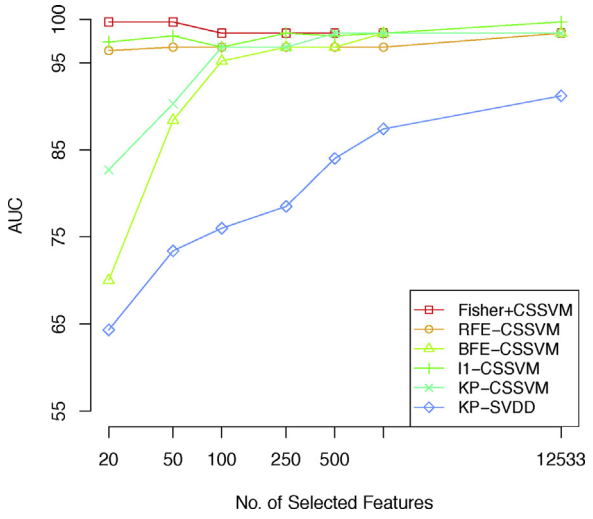

Fig. A.1. Performance versus n for various feature selection approaches. GORDON dataset.

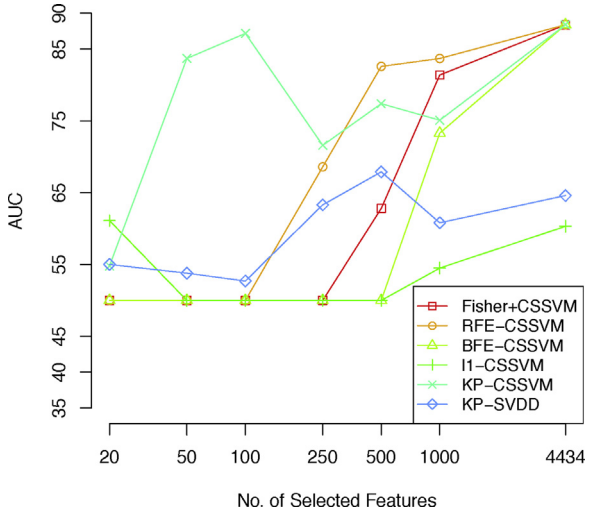

Fig. A.2. Performance versus n for various feature selection approaches. GLIOMA dataset.

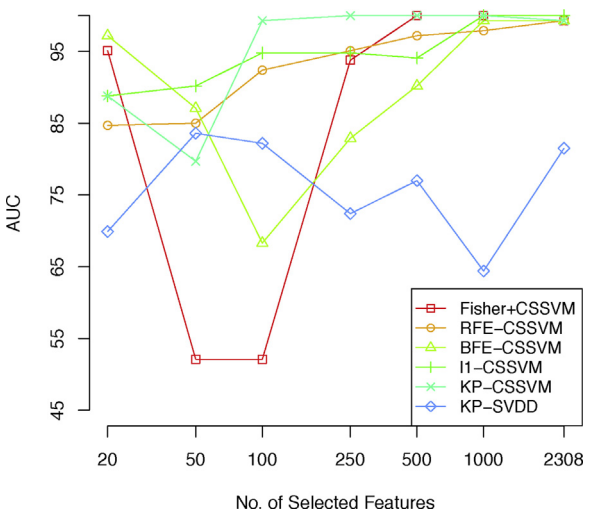

Fig. A.3. Performance versus n for various feature selection approaches. SRBCT dataset.

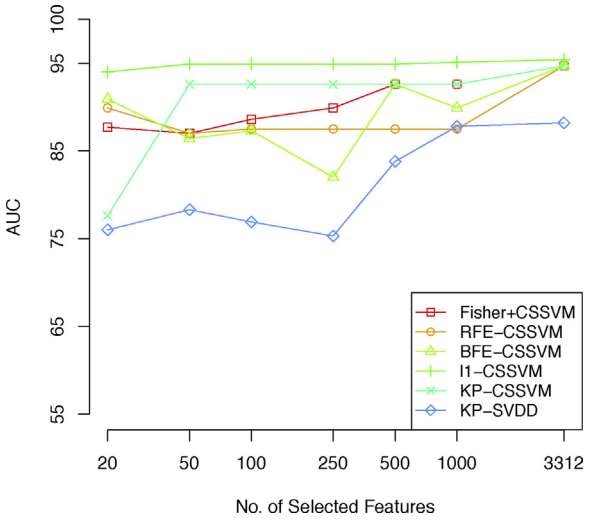

Fig. A.4. Performance versus n for various feature selection approaches. BHAT3 dataset.

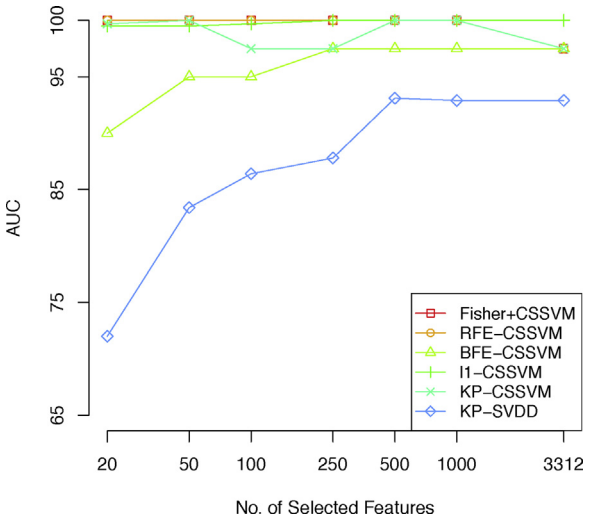

Fig. A.5. Performance versus n for various feature selection approaches. BHAT1 dataset.

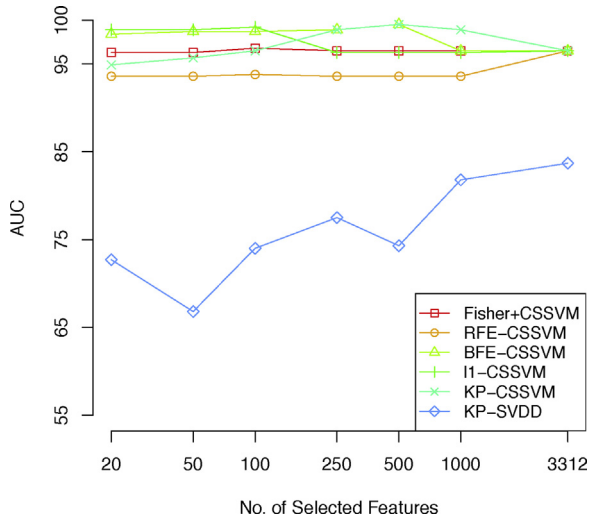

Fig. A.6. Performance versus n for various feature selection approaches. BHAT2 dataset.

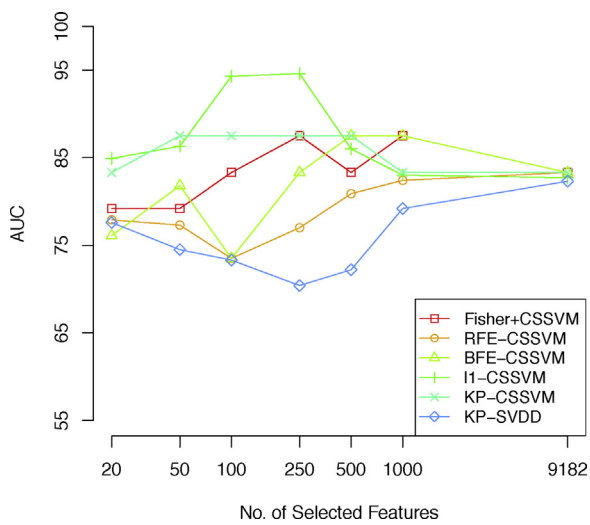

Fig. A.7. Performance versus n for various feature selection approaches. CAR2 dataset.

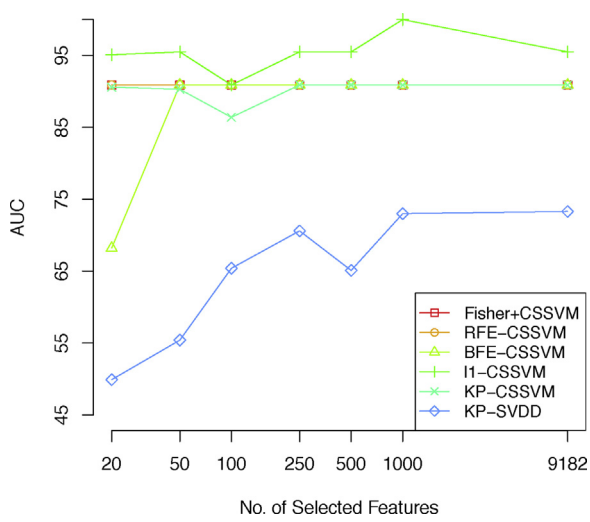

Fig. A.8. Performance versus n for various feature selection approaches. CAR1 dataset.

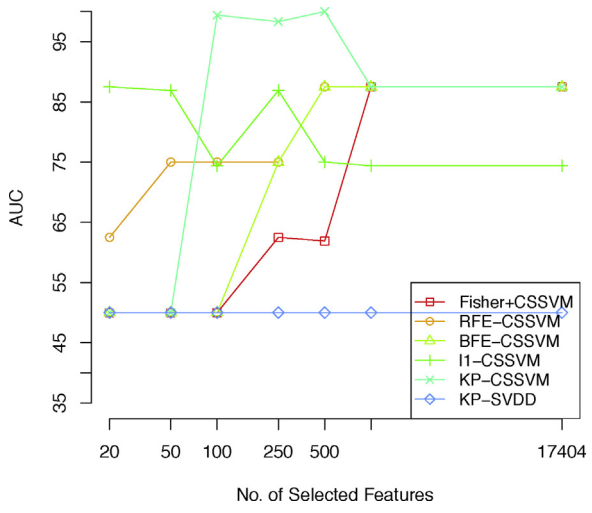

Fig. A.9. Performance versus n for various feature selection approaches. BULL dataset.

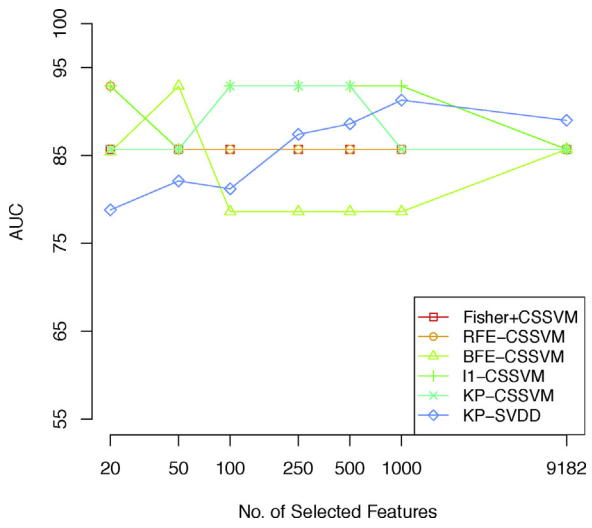

Fig. A.10. Performance versus n for various feature selection approaches. CAR3 dataset.

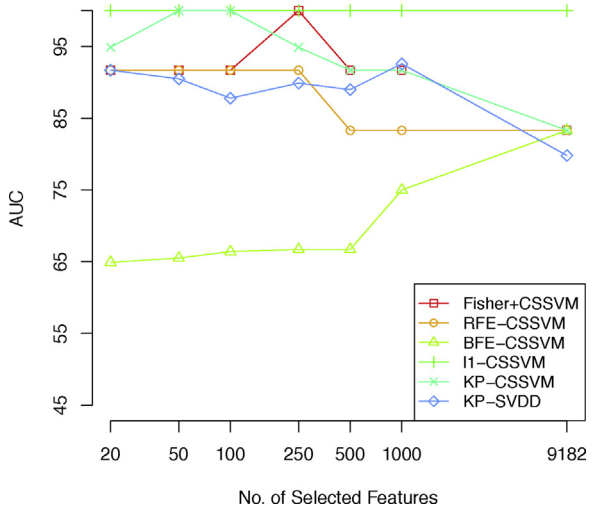

Fig. A.11. Performance versus n for various feature selection approaches. CAR4 dataset.

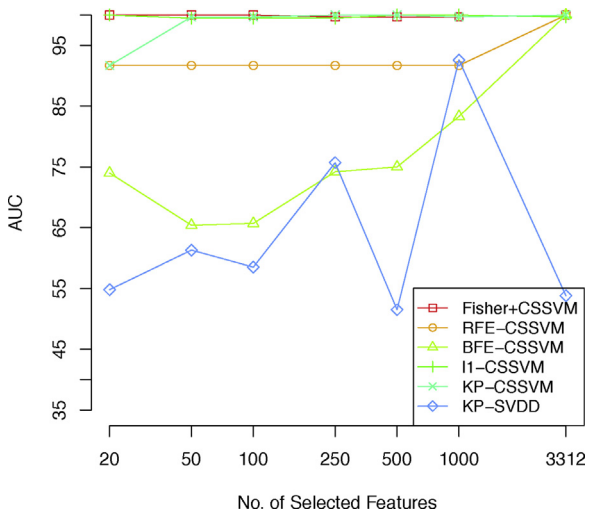

Fig. A.12. Performance versus n for various feature selection approaches. BHAT4 dataset.

References

[1] V. García, J. Sánchez, R. Mollineda, On the effectiveness of preprocessing methods when dealing with different levels of class imbalance, Knowl.-Based Syst. 25 (2012) 13–21.
[2] Q. Zhou, H. Zhou, T. Li, Cost-sensitive feature selection using random forest: selecting low-cost subsets of informative features, Knowl.-Based Syst. 95 (2016) 1–11.
[3] V. Bolón-Canedo, N. Sánchez-Maroño, A. Alonso-Betanzos, Recent advances and emerging challenges of feature selection in the context of big data, Knowl.-Based Syst. 86 (2015) 33–45.
[4] S. Maldonado, R. Weber, J. Basak, Kernel-penalized SVM for feature selection, Inform. Sci. 181 (2011) 115–128.
[5] V. Vapnik, Statistical Learning Theory, John Wiley and Sons, 1998.
[6] S. Maldonado, R. Weber, F. Famili, Feature selection for high-dimensional class-imbalanced data sets using support vector machines, Inform. Sci. 286 (2014) 228–246.
[7] H. Yu, C. Mu, C. Sun, W. Yang, X. Yang, X. Zuo, Support vector machine-based optimized decision threshold adjustment strategy for classifying imbalanced data, Knowl.-Based Syst. 76 (2015) 67–78.
[8] D. Tax, R. Duin, Support vector data description, Mach. Learn. 54 (2004) 45–66.
[9] F. Bach, D. Heckerman, E. Horvitz, Considering cost asymmetry in learning classifiers, J. Mach. Learn. Res. 7 (2006) 1713–1741.
[10] H. He, E. García, Learning from imbalanced data, IEEE Trans. Knowl. Data Eng. 21 (2009) 1263–1284.
[11] J. Benítez, N. García-Pedrajas, F. Herrera, Special issue on new trends in data mining, Knowl.-Based Syst. 25 (2012) 1–2.
[12] N. Chawla, N. Japkowicz, A. Kotcz, Editorial: special issue on learning from imbalanced data sets, SIGKDD Explor. 6 (2004) 1–6.
[13] N. García-Pedrajas, J. Pérez-Rodríguez, M. García-Pedrajas, D. Ortiz-Boyer, C. Fyfe, Class imbalance methods for translation initiation site recognition in DNA sequences, Knowl.-Based Syst. 25 (2012) 22–34.
[14] N.V. Chawla, L. Hall, K. Bowyer, W. Kegelmeyer, Smote: synthetic minority oversampling technique, J. Artif. Intell. Res. 16 (2002) 321–357.
[15] J. Van Hulse, T. Khoshgoftaar, A. Napolitano, R. Wald, Feature selection with high-dimensional imbalanced data, Proceedings of the IEEE International Conference on Data Mining Workshops (2009) 507–514.
[16] B. Schökopf, A.J. Smola, Learning with Kernels, MIT Press, 2002.
[17] W. Zhu, P. Zhong, A new one-class SVM based on hidden information, Knowl.-Based Syst. 60 (2014) 35–43.
[18] B. Schölkopf, J. Platt, J. Shawe-Taylor, A.J. Smola, R.C. Williamson, Estimating the support of a high-dimensional distribution, Neural Comput. 13 (2001) 1443–1471.
[19] J. Mercer, Functions of positive and negative type, and their connection with the theory of integral equations, Philos. Trans. R. Soc. Lond. 209 (1909) 415–446.
[20] R. Blagus, L. Lusa, Class prediction for high-dimensional class-imbalanced data, BMC Bioinform. 11 (2010) 523.
[21] A.A. Shanab, T.M. Khoshgoftaar, R. Wald, J. Van Hulse, Comparison of approaches to alleviate problems with high-dimensional and class-imbalanced data, 2011 IEEE International Conference on Information Reuse and Integration (IRI) (2011) 234–239.
[22] M. Wasikowski, X. Chen, Combating the small sample class imbalance problem using feature selection, IEEE Trans. Knowl. Data Eng. 22 (2010) 1388–1400.
[23] Z. Zheng, X. Wu, R. Srihari, Feature selection for text categorization on imbalanced data, SIGKDD Explor. 6 (2004) 80–89.

[24] R. Duda, P. Hard, D. Stork, Pattern Classification, Wiley-Interscience Publication, 2001.
[25] X. Chen, M. Wasikowski, Fast: a ROC-based feature selection metric for small samples and imbalanced data classification problems, Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2008) (2009) 124–132.
[26] 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. 81-82 (2012) 67–103.
[27] I. Guyon, S. Gunn, M. Nikravesh, L.A. Zadeh, Feature Extraction, Foundations and Applications, Springer, Berlin, 2006.
[28] L. Song, A. Smola, A. Gretton, J. Bedo, K. Borgwardt, Feature selection via dependence maximization, J. Mach. Learn. Res. 13 (2012) 1393–1434.
[29] Y.-S. Jeong, I.-H. Kang, M.-K. Jeong, D. Kong, A new feature selection method for one-class classification problems, IEEE Trans. Syst. Man Cybern. Part C Appl. Rev. 42 (2012) 1500–1509.
[30] P. Bradley, O. Mangasarian, Feature selection via concave minimization and support vector machines, in: Machine Learning proceedings of the fifteenth International Conference (ICML'98), Morgan Kaufmann, San Francisco, CA, 1998, pp. 82–90.
[31] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, C.-J. Lin, Liblinear: a library for large linear classification, J. Mach. Learn. Res. 9 (2008) 1871–1874.
[32] S. Canu, Y. Grandvalet, Adaptive scaling for feature selection in SVMS, in: Advances in Neural Information Processing Systems, vol. 15, MIT Press, Cambridge, MA, USA, 2002, pp. 553–560.
[33] O. Chapelle, V. Vapnik, O. Bousquet, S. Mukherjee, Choosing multiple parameters for support vector machines, Mach. Learn. 46 (2002) 131–159.
[34] J. Neumann, C. Schnörr, G. Steidl, Combined SVM-based feature selection and classification, Mach. Learn. 61 (2005) 129–150.
[35] M. Powell, A fast algorithm for nonlinearly constrained optimization calculations, in: G. Watson (Ed.), Numerical Analysis, vol. 630 of Lecture Notes in Mathematics, Springer, Berlin, Heidelberg, 1978, pp. 144–157.
[36] J. Nocedal, S.J. Wright, Numerical Optimization, Springer-Verlag, New York, 2006.
[37] W. Hock, K. S, Test Examples for Nonlinear Programming Codes, Lecture Notes in Economics and Mathematical Systems, Springer-Verlag, New York, 1981.
[38] D. Beer, S. Kardia, C. Huang, T. Giordano, A. Levin, D. Misek, L. Lin, G. Chen, T. Gharib, D. Thomas, M. Lizyness, R. Kuick, S. Hayasaka, J. Taylor, M. Iannettoni, M. Orringer, S. Hanash, Gene-expression profiles predict survival of patients with lung adenocarcinoma, Nat. Med. 8 (2002) 816–824.
[39] C. Nutt, D. Mani, R. Betensky, P. Tamayo, J. Cairncross, C. Ladd, U. Pohl, C. Hartmann, M. McLaughlin, T. Batchelor, P. Black, A. von Deimling, S. Pomeroy, T. Golub, D. Louis, Gene expression-based classification of malignant gliomas correlates better with survival than histological classification, Cancer Res. 63 (2003) 1602–1607.
[40] J. Khan, J. Wei, M. Ringner, L. Saal, M. Ladanyi, F. Westermann, F. Berthold, M. Schwab, C. Antonescu, C. Peterson, P. Meltzer, Classification and diagnostic prediction of cancers using gene expression profiling and artificial neural networks, Nat. Med. 7 (2001) 673–679.
[41] A. Bhattacharjee, W. Richards, J. Staunton, C. Li, S. Monti, P. Vasa, C. Ladd, J. Beheshti, R. Bueno, M. Gillette, M. Loda, G. Weber, E. Mark, E. Lander, W. Wong, B. Johnson, T. Golub, D. Sugarbaker, M. Meyerson, Classification of human lung carcinomas by mRNA expression profiling reveals distinct adenocarcinoma subclasses, Proc. Natl. Acad. Sci. U.S.A. 98 (2001) 13790–13795.
[42] A. Su, J. Welsh, L. Sapinoso, S. Kern, P. Dimitrov, H. Lapp, P. Schultz, S. Powell, C. Moskaluk, H.J. Frierson, G. Hampton, Molecular classification of human carcinomas by use of gene expression signatures, Cancer Res. 61 (2001) 7388–7393.
[43] L. Bullinger, K. Dohner, S. Frohling, E. Bair, R. Schlenk, R. Tibshirani, H. Dohner, J. Pollack, Use of gene-expression profiling to identify prognostic subclasses in adult acute myeloid leukemia, N. Engl. J. Med. 350 (2004) 1605–1616.
[44] K. Yang, Z. Cai, J. Li, G. Lin, A stable gene selection in microarray data analysis, BMC Bioinform. 7 (2006) 228.
[45] S. Maldonado, C. Montecinos, Robust classification of imbalanced data using ensembles of one-class and two-class SVMS, Intell. Data Anal. Spec. Iss. Business Anal. Intell. Optim. 18 (2014) 95–112.
[46] T. Lin, R. Liu, C. Chen, Y. Chao, S. Chen, Pattern classification in DNA microarray data of multiple tumor types, Pattern Recogn. 39 (2006) 2426–2438.
[47] A. Rakotomamonjy, Variable selection using SVM-based criteria, J. Mach. Learn. Res. 3 (2003) 1357–1370.
[48] D. Kriesel, A Brief Introduction to Neural Networks, 2007.
[49] C.-C. Chang, C.-J. Lin, LIBSVM: a library for support vector machines, ACM Trans. Intell. Syst. Technol. 2 (2011), 27-1–27:27. Software available at http://www.csie.ntu.edu.tw/cjlin/libsvm.
[50] D. Tax, Ddtools, The Data Description Toolbox for Matlab, Version 2.1.1, 2014.
[51] J. Demšar, Statistical comparisons of classifiers over multiple data set, J. Mach. Learn. Res. (2006) 1–30.
[52] S. Holm, A simple sequentially rejective multiple test procedure, Scand. J. Stat. 6 (1979) 65–70.
[53] B. Baesens, Analytics in a Big Data World, John Wiley and Sons, 2014.
[54] M. Carrasco, J. López, S. Maldonado, A multi-class SVM approach based on the l1-norm minimization of the distances between the reduced convex hulls, Pattern Recogn. 48 (2015) 1598–1607.
[55] N. Djuric, L. Lan, S. Vucetic, Z. Wang, Budgetedsvm: a toolbox for scalable SVM approximations, J. Mach. Learn. Res. 14 (2013) 3813–3817.