Journal of Convex Analysis
Volume 33 (2026), No. 3&4, 781–800

Robust SVM Classification with ℓ_0-Norm Feature Selection

Miguel Carrasco
Facultad de Ingeniería y Ciencias Aplicadas, Universidad de los Andes, Santiago, Chile
macarrasco@miuandes.cl

Benjamin Ivorra
Interdisciplinary Mathematics Institute, Department of Applied Mathematics and Mathematical Analysis, Universidad Complutense de Madrid, España
ivorra@ucm.es

Julio López
Facultad de Ingeniería y Ciencias, Universidad Diego Portales, Santiago, Chile
julio.lopez@udp.cl

Angel M. Ramos
Interdisciplinary Mathematics Institute, Department of Applied Mathematics and Mathematical Analysis, Universidad Complutense de Madrid, España
angel@mat.ucm.es

Dedicated to the memory of Hedy Attouch.

Received: September 30, 2024
Accepted: May 10, 2025

We introduce a robust classification model designed for feature selection. Support vector machine (SVM) models continue to play a crucial role in binary classification, particularly with tabular data. Their robust variants are essential for developing classifiers that remain stable despite shifts in data distribution. Additionally, sparse classifiers are highly desirable, as they offer improved performance in classification tasks and help reduce overfitting, especially when the number of features exceeds the number of samples. In this context, penalty methods for feature selection are fundamental to the development of sparse optimization models. Despite its inherent nonlinearity and nonconvexity, the ℓ_0-norm remains a popular choice for this task and has been effectively applied in diverse fields, such as pattern recognition and signal processing. Our objective is to evaluate the performance of two classification approaches that integrate robust classification in SVM-type models with embedded feature selection. We propose a diagonal algorithm for numerically solving these optimization models, which we validate through numerical tests on benchmark datasets. The results are compared to those obtained using models based on ℓ_2-norm regularization.

Keywords: Feature selection, robust optimization, second-order cone programming, support vector machines.

2020 Mathematics Subject Classification: 90C17, 90C26, 62H30, 65K05, 68T05.

1. Introduction

One of the most important tasks in the field of Machine Learning is supervised classification. This task involves predicting the label of an individual based on statistical learning from a historical dataset of, say, m previously labeled individuals.

ISSN 0944-6532 / $ 2.50

In this article, we focus on binary classification, which occurs when the label takes values in $\{-1, +1\}$, indicating whether the individual, defined by a vector in $\mathbb{R}^n$, has a condition or not. In this context, the Support Vector Machine (SVM) model [10, 33] remains one of the most relevant in the literature for tabular data due to its strong performance in the classification task, good theoretical properties, and interesting geometric interpretation. Moreover, it has led to various extensions based on this approach; see [18, 28].

It is possible to observe, see [7], that the classifier obtained using an SVM-type model can be sensitive to slight changes in the data distribution, even when considering a cross-validation process. To improve the robustness of the classifier, alternatives have been proposed that aim to maximize the classification accuracy rate such as the so-called Minimax Probability Machine (MPM) [19], or the Minimum Error MPM [17]. Recently, robust versions have emerged that combine the capabilities of these approaches while also maximizing the margin, thereby enhancing predictive power [22, 23]. This is achieved through a probabilistic interpretation of the dataset, where upper bounds for the probability of misclassification are defined based on the knowledge of the mean and the variance-covariance matrix of the data. These probability bounds are obtained through the appropriate application of Chebyshev's inequality, [19, Lemma 1], which results in a Second-Order Cone Programming (SOCP) Problem [2].

On the other hand, these models may lose their predictive power when the feature set is much larger than the number of individuals ($n \gg m$). Several alternatives exist to address this issue, focusing on identifying the relevant features [14]. Among these methodologies, Recursive Feature Elimination methods aim to iteratively eliminate irrelevant features [16]. It is important to note that using these methods requires solving an optimization model at each iteration, which increases execution time as databases grow larger and models become more complex.

It is noteworthy that SVM is amenable to modification to directly incorporate feature selection into the optimization model (see, e.g., [32, 39]). One of the most popular alternatives is replacing the margin maximization performed by the SVM model with the minimization of the ℓ_1-norm, also known as LASSO regularization [24, 38]. Another useful approach is the use of the so-called ℓ_0-norm, which corresponds to the number of nonzero elements in the weight vector [5, 9]. This function has been previously used in several fields, such as machine learning, pattern recognition, and compressed sensing [5, 11, 13]. However, its use is quite challenging due to the complete lack of convexity and differentiability. The introduction of the ℓ_0-norm in the context of classification is achieved through approximations that allow for numerical resolution using DC-programming techniques [20, 21, 31, 36].

In this work, we aim to use the ℓ_0-norm for feature selection to effectively identify the relevant characteristics during the optimization process, while also incorporating a robust approach in the sense of minimizing the probability of misclassification. Our goal is to combine the strengths of robust models with feature selection by defining a robust SVM-type model that embeds the ℓ_0-norm for this purpose. To illustrate the efficiency of the proposed approaches, we conduct several benchmark experiments by combining our model with different feature selection strategies, namely, Recursive

Feature Elimination (RFE) and Direct Feature Elimination (DFE) as proposed in [7]. The results are compared with those obtained using ℓ_2-norm regularization.

The outline of the article is as follows: In Section 2, we present the classic SVM formulation along with its robust counterparts. We also briefly summarize relevant results related to feature selection that will be useful in this work. In Section 3, we define the models that are the subject of our study and the algorithm to solve them. Section 4 describes the datasets and analyze the obtained numerical results. Finally, Section 5 reviews the major findings and outline possible future lines of research.

2. Preliminaries

In this section, we present the main models considered in this article. We begin with the traditional (SVM) model, followed by a discussion of some robust alternatives relevant to our study, including the *Minimum Error Minimax Probability Machine* (MEMPM) [17], and the *Cobb-Douglas Learning Machine* (CD-LeMa) [23]. Additionally, we provide a brief overview of feature selection, with particular emphasis on the primary tools that will be employed in the numerical experiments.

2.1. Review of binary classification models

Consider a dataset with m patterns, where each sample is represented by a vector $\mathbf{x}_i \in \mathbb{R}^n$, for $i = 1, \ldots, m$. Additionally, each sample belongs to one of two classes: positive or negative. The standard SVM model (specifically the Soft-Margin SVM) aims to maximize the margin (defined as $1/\|\mathbf{w}\|_2$, where $\|\cdot\|_2$ denotes the Euclidean norm) while allowing for some classification error in separating the data of the positive class from that of the negative class. It is defined as follows:

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

where $y_i = +1$ if the i-th sample belongs to the positive class, and $y_i = -1$ otherwise. Once $\mathbf{w}$ and b are obtained, a new sample $\mathbf{x} \in \mathbb{R}^n$ is assigned to the positive class if $\text{sign}\{\mathbf{w}^\top \mathbf{x} + b\} = +1$ and to the negative one otherwise.

This approach does not take into account the stochastic nature of the data, meaning that it assumes the input samples are fixed rather than random. As a consequence, the resulting classifiers may be overly tailored to a specific dataset and perform poorly when the data distribution changes. To overcome this drawback, we consider the following framework. We assume that $\widetilde{\mathbf{x}}_1$ and $\widetilde{\mathbf{x}}_2$ are n-dimensional random vectors that generate the samples of the positive and negative classes, respectively, with mean and covariance matrices given by $(\boldsymbol{\mu}_k, \Sigma_k)$ for $k = 1, 2$, where each $\Sigma_k \in \mathbb{R}^{n\times n}$ is a symmetric positive definite matrix. We denote by $\widetilde{\mathbf{x}} \sim (\boldsymbol{\mu}, \Sigma)$ that the random vector $\widetilde{\mathbf{x}}$ belongs to the family of distributions with a common mean $\boldsymbol{\mu}$ and covariance matrix Σ. We note that, in practice, the empirical estimates for the mean and the covariance matrix are used instead. Under these considerations, the stochastic models MEMPM [17] and CD-LeMa [23] can be summarized by the following optimization problem:

$$\begin{aligned} \max_{\mathbf{w}\neq\mathbf{0},b,\alpha_1,\alpha_2} \quad & f(\alpha_1,\alpha_2) \\ \text{s.t.} \inf_{\widetilde{\mathbf{x}}_i\sim(\boldsymbol{\mu}_i,\Sigma_i)} & \Pr\{(-1)^{i+1}(\mathbf{w}^\top\widetilde{\mathbf{x}}_i+b)\geq 0\}\geq\alpha_i,\ 0<\alpha_i<1,\ i=1,2, \end{aligned} \tag{2}$$

where the objective function f is given by

$$f(\alpha_1,\alpha_2)=\begin{cases} \theta\alpha_1+(1-\theta)\alpha_2 & , \text{ case: MEMPM,} \\ \theta\ln(\alpha_1)+(1-\theta)\ln(\alpha_2) & , \text{ case: CD-LeMa,} \end{cases} \tag{3}$$

with θ being a parameter between $(0,1)$. Here, θ and $1-\theta$ represent the prior probabilities of the random vectors $\widetilde{\mathbf{x}}_1$ and $\widetilde{\mathbf{x}}_2$, respectively. These models aim to find a hyperplane of the form $\mathbf{w}^\top\mathbf{x}+b=0$ that separates both classes without exactly knowing the distribution of the data (noting that the infimum is taken over all distributions with a common mean $\boldsymbol{\mu}$ and covariance matrix Σ). The MEMPM approach can be seen as an extension of the seminal work *Minimax Probability Machine* (MPM) [19], in which $\alpha_1=\alpha_2$ in Problem (2) above.

The following lemma will be useful for defining deterministic formulations associated with Problem (2) (see [19, Lemma 1] for details):

Lemma 2.1. (Multivariate Chebyshev Inequality) *Let $\widetilde{\mathbf{x}}$ be a n-dimensional random variable with mean $\boldsymbol{\mu}\in\mathbb{R}^n$ and covariance matrix $\Sigma\in\mathbb{R}^{n\times n}$, which is symmetric positive definite. Given $\mathbf{a}\in\mathbb{R}^n\setminus\{\mathbf{0}\}$, $b\in\mathbb{R}$, such that $\mathbf{a}^\top\boldsymbol{\mu}+b\geq 0$, and $\alpha\in(0,1)$, the condition*

$$\inf_{\widetilde{\mathbf{x}}\sim(\boldsymbol{\mu},\Sigma)}\Pr\{\mathbf{a}^\top\widetilde{\mathbf{x}}+b\geq 0\}\geq\alpha$$

holds if and only if $\mathbf{a}^\top\boldsymbol{\mu}+b\geq\kappa(\alpha)\sqrt{\mathbf{a}^\top\Sigma\mathbf{a}}$, where $\kappa(\alpha)=\sqrt{\frac{\alpha}{1-\alpha}}$.

By using Lemma 2.1, Problem (2) can be cast equivalently as the following nonlinear nonconvex SOCP problem

$$\begin{aligned} \max_{\mathbf{w}\neq\mathbf{0},b,\alpha_1,\alpha_2} \quad & f(\alpha_1,\alpha_2) \\ \text{s.t.} \quad & \hat{y}_i(\mathbf{w}^\top\boldsymbol{\mu}_i+b)\geq\kappa(\alpha_i)\sqrt{\mathbf{w}^\top\Sigma_i\mathbf{w}},\ 0<\alpha_i<1, \\ & \hat{y}_i(\mathbf{w}^\top\boldsymbol{\mu}_i+b)\geq 0, \quad i=1,2, \end{aligned} \tag{4}$$

where we set $\hat{y}_1=+1$ and $\hat{y}_2=-1$ for the class represented by $\widetilde{\mathbf{x}}_1$ and $\widetilde{\mathbf{x}}_2$, respectively. Additionally, we define $\kappa(\alpha_i)=\sqrt{\frac{\alpha_i}{1-\alpha_i}}$ for $i=1,2$.

In the case of the MEMPM approach, the authors reformulate the problem as a fractional programming problem and solve it iteratively using a Quadratic Interpolation method. In contrast, the CD-LeMa approach takes advantage of the log-concavity of the objective function, allowing it to be solved numerically using a successive convex approximation algorithm [4, 6].

2.2. Review of feature selection in SVM models

As we mentioned earlier, the SVM model and its robust counterparts described previously do not take into consideration the importance of features, which can lead to a decrease in model performance [7]. Several alternatives have been developed to ad-

dress this issue, including Recursive Feature Elimination (**RFE**) and Direct Feature Elimination (**DFE**). **RFE** consists of iteratively selecting features and optimizing the classification model, while **DFE** selects features based on the classification model optimized with all features. The process ends with a minimal subset of features that perform well on the test set. However, these methods are computationally expensive and, also, may disregard feature combinations that could be useful for classification.

Another alternative that will be useful in our context is the penalty approach, which involves adding a regularization term to the optimization model for feature selection. Examples include the ℓ_1-norm [5, 24, 38], defined as $\|\mathbf{w}\|_1 = \sum_i |w_i|$, and the non-convex ℓ_p-norm (with $0 < p < 1$) [8, 30]. In this article, we focus on this penalty approach by considering the so-called *zero norm* (denoted as the ℓ_0-norm or simply $\|\cdot\|_0$), which, as mentioned in the introduction, corresponds to the number of non-zero elements in the weight vector and is mathematically defined as

$$\|\mathbf{w}\|_0 = \text{card}\{i : w_i \neq 0\}.$$

Of course, this function is not a norm since it does not satisfy the homogeneity. Additionally, it is highly nonlinear, nonsmooth, and non-convex, making the associated optimization problems generally impossible to solve, as their solutions may require intractable combinatorial searches. Despite these challenges, it has been applied in several fields, including machine learning, where the goal is to improve sparsity [5, 11, 13]. In this context, Bradley and Mangasarian [5] introduced the ℓ_0-SVM model, which replaces the ℓ_2-norm in the standard SVM formulation (1) with the ℓ_0-norm. To address the complexity of this model, they proposed the following continuous approximation:

$$\|\mathbf{w}\|_0 \approx \sum_{i=1}^{n} (1 - \exp(-r|w_i|)), \tag{5}$$

where $r > 0$ is a hyperparameter. The advantage of employing this approximation is that it transforms the ℓ_0-SVM formulation into the minimization of a smooth concave objective function over a polyhedral set. This reformulation can be solved using the successive linearization approximation (SLA) algorithm, which guarantees finite termination at a stationary point that can also be a global solution (for more details, see [5]). Alternatively, other approximations for the ℓ_0-norm have been proposed in the literature, such as those found in [9, 27, 31, 36].

To stabilize the aforementioned models, other approaches suggest combining the ℓ_2-norm with either the ℓ_1-norm or the ℓ_0-norm, as the ℓ_2-norm is suitable for achieving good classification performance in SVMs. [25, 34].

3. Proposed approaches

We now present our robust classification proposals that integrate feature selection with the goal of obtaining sparse classifiers. Following this, we will outline a numerical solution strategy for these proposals, along with a description of the implemented algorithm.

3.1. Robust classification models with embedded feature selection

In order to obtain sparse and robust classifiers, we consider regularized versions of the MEMPM [17] and CD-LeMa [23] approaches defined previously in formulation (4).

First, it is easy to see that by denoting $\kappa_i = \sqrt{\frac{\alpha_i}{1-\alpha_i}}$, one has $\alpha_i = \kappa_i^2/(1+\kappa_i^2)$, for $i = 1, 2$, and therefore, Problem (4) can be rewritten in terms of the variable κ_i instead of α_i. Second, to avoid the constraint $\mathbf{w} \neq \mathbf{0}$ and the multiplicity of solutions that can lead to numerical instabilities, we use the fact that the constraints are positively homogeneous (i.e., scaling $(\mathbf{w}, b)$ by t leads to the same solution) and, therefore, we impose the constraints $\hat{y}_i(\mathbf{w}^\top \boldsymbol{\mu}_i + b) \geq 1$ instead of $\hat{y}_i(\mathbf{w}^\top \boldsymbol{\mu}_i + b) \geq 0$. Finally, we regularize the objective function using the ℓ_0-norm. We aim to minimize the number of nonzero elements in our classifier while maximizing the rate of correct classification. In its compact form, after all these changes, the ℓ_0-norm regularization of Problem (4) is given by the following formulation:

$$\begin{aligned} \min_{\mathbf{w},b,\kappa_1,\kappa_2} \quad & G(\mathbf{w}, b, \kappa_1, \kappa_2) \\ \text{s.t.} \quad & \hat{y}_i(\mathbf{w}^\top \boldsymbol{\mu}_i + b) \geq \kappa_i \sqrt{\mathbf{w}^\top \Sigma_i \mathbf{w}}, \ \kappa_i \geq 0, \\ & \hat{y}_i(\mathbf{w}^\top \boldsymbol{\mu}_i + b) \geq 1, \quad i = 1, 2. \end{aligned} \tag{6}$$

where the objective function is

$$G(\mathbf{w}, b, \kappa_1, \kappa_2) = \begin{cases} \|\mathbf{w}\|_0 - C_1 \frac{1}{1+\kappa_1^2} - C_2 \frac{1}{1+\kappa_2^2} & , \text{ case: } \ell_0\text{-MEMPM}, \\ \|\mathbf{w}\|_0 - C_1 \ln\left(\frac{\kappa_1^2}{1+\kappa_1^2}\right) - C_2 \ln\left(\frac{\kappa_2^2}{1+\kappa_2^2}\right) & , \text{ case: } \ell_0\text{-CD-LeMa}. \end{cases} \tag{7}$$

Here, C_1 and C_2 are positive hyperparameters. In the literature, other possible regularizations are considered for these SVM formulations, such as the ℓ_1-norm [7], as mentioned above, or a combination of the ℓ_2-norm with the ℓ_0-norm to enhance predictive power through feature selection [20]. However, this is beyond the scope of this article.

To provide a comprehensive comparison, we also consider the ℓ_2-norm regularization to both the MEMPM and CD-LeMa models, which have been studied in [22, 23]. This alternative is useful for assessing the performance of our sparsity-driven approach against a more traditional method. The idea of theses approaches is to replace the objective function of Problem (6) by

$$(\mathbf{w}, b, \kappa_1, \kappa_2) = \begin{cases} \frac{1}{2}\|\mathbf{w}\|_2^2 - C_1 \frac{1}{1+\kappa_1^2} - C_2 \frac{1}{1+\kappa_2^2} & , \text{ case: } \ell_2\text{-MEMPM}, \\ \frac{1}{2}\|\mathbf{w}\|_2^2 - C_1 \ln\left(\frac{\kappa_1^2}{1+\kappa_1^2}\right) - C_2 \ln\left(\frac{\kappa_2^2}{1+\kappa_2^2}\right), & \text{case: } \ell_2\text{-CD-LeMa}. \end{cases} \tag{8}$$

3.2. Approximation scheme and diagonal two-step algorithm

In this section, we propose a numerical strategy for solving Problem (6). The idea is to approximate the ℓ_0-norm function as well as the nonlinear constraints. This will allow us to define an algorithm that solves a sequence of convex programming problems.

Let us consider the approximation of the ℓ_0-norm defined in (5) and take $r > 0$ as a fixed parameter. The literature suggests taking $r = 5$ [5, 20], although some works treat it as a hyperparameter and determine its value through cross-validation [37]. Since the ℓ_0-norm is part of the objective function in (6), we can introduce auxiliary variables as part of the optimization process and replace it with the following:

$$\rho(\mathbf{z}) = -\sum_{i=1}^{n} \exp(-rz_i), \quad \text{with } -\mathbf{z} \leq \mathbf{w} \leq \mathbf{z} \text{ and } \mathbf{z} \geq 0, \tag{9}$$

which is a concave function. For the nonlinear SOC constraints of (6)

$$(\hat{y}_i(\mathbf{w}^\top \boldsymbol{\mu}_i + b) \geq \kappa_i \sqrt{\mathbf{w}^\top \Sigma_i \mathbf{w}}, \text{ with } \ i = 1, 2),$$

we apply the well-known inequality $2\sqrt{ab} \leq a + b$ to the right-hand side and add an extra parameter $t_i > 0$, obtaining:

$$\hat{y}_i(\mathbf{w}^\top \boldsymbol{\mu}_i + b) \geq \frac{1}{2}\Big(t_i \kappa_i^2 + \frac{\mathbf{w}^\top \Sigma_i \mathbf{w}}{t_i}\Big), \quad i = 1, 2.$$

We note that the inequality matches the SOC constraint when we take

$$t_i = \frac{\sqrt{\mathbf{w}^\top \Sigma_i \mathbf{w}}}{\kappa_i}.$$

Finally, we change the variables to $\vartheta_i = \kappa_i^2$ and we obtain the following approximation of (6):

$$\begin{aligned} \min_{\mathbf{w},\mathbf{z},b,\vartheta_i,t_i} \quad & \rho(\mathbf{z}) + g(\vartheta_1, \vartheta_2) \\ \text{s.t.} \quad & \hat{y}_i(\mathbf{w}^\top \boldsymbol{\mu}_i + b) \geq \frac{1}{2}\Big(t_i \vartheta_i + \frac{\mathbf{w}^\top \Sigma_i \mathbf{w}}{t_i}\Big), \\ & \hat{y}_i(\mathbf{w}^\top \boldsymbol{\mu}_i + b) \geq 1, \ \vartheta_i \geq 0, \ t_i > 0, \ \text{ for } i = 1, 2, \\ & -\mathbf{z} \leq \mathbf{w} \leq \mathbf{z}, \quad \mathbf{z} \geq \mathbf{0}, \end{aligned} \tag{10}$$

where in this case

$$g(\vartheta_1, \vartheta_2) = \begin{cases} -C_1 \frac{1}{1+\vartheta_1} - C_2 \frac{1}{1+\vartheta_2} & , \text{ case: } \ell_0\text{-MEMPM}, \\ -C_1 \ln\big(\frac{\vartheta_1}{1+\vartheta_1}\big) - C_2 \ln\big(\frac{\vartheta_2}{1+\vartheta_2}\big) & , \text{ case: } \ell_0\text{-CD-LeMa}. \end{cases} \tag{11}$$

It is not difficult to show that model (10) is equivalent to (6) when the ℓ_0-norm is replaced with the function $\rho(\cdot)$ defined above (9).

Next, we propose a *Diagonal Two-Step* Algorithm (see Algorithm 1 below) to solve the formulation (10) (or equivalently, the approximated version of Problem 6). Essentially, it involves linearizing the concave function and minimizing the resulting problem with the auxiliary parameters t_i fixed. These parameters are updated in the next step according to $t_i = \sqrt{\mathbf{w}^\top \Sigma_i \mathbf{w}/\vartheta_i}$. This process is summarized in Algorithm 1 below.

Algorithm 1: Diagonal two-step algorithm for solving problem (10).

Input: $C_1, C_2, \boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \Sigma_1$, and Σ_2.

Output: Solution $(\mathbf{w}^k, b^k, \vartheta_1^k, \vartheta_2^k)$.

1: Start with $t_1^0, t_2^0 > 0$ and $\mathbf{z}^0$. Set $k = 0$.

2: **repeat**

3: For a given $t_1^k, t_2^k > 0$, and $\mathbf{z}^k$, compute $(\mathbf{w}^{k+1}, \mathbf{z}^{k+1}, b^{k+1}, \vartheta_1^{k+1}, \vartheta_2^{k+1})$ by solving the following smooth convex optimization problem

$$\begin{aligned} \min_{\mathbf{w},\mathbf{z},b,\vartheta_i} \quad & \nabla\rho(\mathbf{z}^k)^\top \mathbf{z} + g(\vartheta_1,\vartheta_2) \\ \text{s.t.} \quad & \hat{y}_i(\mathbf{w}^\top \boldsymbol{\mu}_i + b) \geq \frac{1}{2}\big(t_i^k \vartheta_i + \frac{\mathbf{w}^\top \Sigma_i \mathbf{w}}{t_i^k}\big), \\ & \hat{y}_i(\mathbf{w}^\top \boldsymbol{\mu}_i + b) \geq 1, \ \vartheta_i \geq 0, \quad \text{for } i = 1,2, \\ & -\mathbf{z} \leq \mathbf{w} \leq \mathbf{z}, \quad \mathbf{z} \geq \mathbf{0}. \end{aligned} \tag{12}$$

4: Compute t_1^{k+1}, t_2^{k+1} by $t_i^{k+1} = \sqrt{\dfrac{\mathbf{w}^{k\top} \Sigma_i \mathbf{w}^k}{\vartheta_i^k}}$, for $i = 1,2$. (13)

5: $k = k + 1$

6: **until** Stopping criterion is reached

An alternative way to solve model (10) is through the so-called Difference of Convex (DC) Algorithm, originally proposed in [12]; see also [20, 21]. In DC programming, during each main iteration, we linearize the concave function for a given $\mathbf{z}^k$ and solve the resulting problem. To solve this inner problem, we apply an iterative scheme that depends on the parameter t_i until a convergence criterion is satisfied. We then update the value of $\mathbf{z}^k$ and repeat the process until another convergence criterion is fulfilled. We call our algorithm *Diagonal* because, in contrast to DC programming, we take a descent step with the linearized concave function and directly perform another step to update the parameters t_i, thus avoiding the double loop of the DC algorithm.

Case	N_p	N_f	Shorthand Notation
Colorectal [3]	62	2000	$\mathbf{B}_C$
Gravier [15]	168	2905	$\mathbf{B}_G$
Lymphoma [1]	96	4026	$\mathbf{B}_L$
Pomeroy [26]	60	7128	$\mathbf{B}_P$
West [35]	49	7129	$\mathbf{B}_W$
Shipp [29]	77	7129	$\mathbf{B}_S$

Table 1: Summary of benchmark datasets considered in Section 4.2. Each dataset's name (**Dataset**), number of patterns (N_p), number of features (N_f), and shorthand notation used in the paper are provided.

4. Experimental evaluation of model performance

In this section, we analyze the performance of the proposed models using six benchmark datasets to demonstrate their efficiency in feature selection and solution robustness. Section 4.1 offers a comprehensive overview of the numerical experiments conducted. In Section 4.2, we present and discuss the results, highlighting key findings and insights drawn from the experiments.

4.1. Descriptions of the benchmark datasets and methodology

In this work, we examine six benchmark datasets, detailed in Table 1, which includes each dataset's name, number of patterns ($N_p \in \mathbb{N}$), number of features ($N_f \in \mathbb{N}$),

and the shorthand notation used for reference. Further descriptions of these datasets can be found in [21].

For each of these benchmark datasets, the models introduced in (7) and (8) are applied, denoted as: $\mathbf{CDL}_2(\ell_2$-CD-LeMa), $\mathbf{CDL}_0(\ell_0$-CD-LeMa), $\mathbf{MEP}_2(\ell_2$-MEMPM), and $\mathbf{MEP}_0(\ell_0$-MEMPM). To evaluate the feature selection performance of each model, a Recursive Feature Elimination (**RFE**) method is used, which progressively eliminates features without compromising classification performance [16]. To accomplish this, we propose the following methodology, thoroughly detailed in [7]:

Step 1 For each benchmark dataset and model, we first identify appropriate values for the hyperparameters C_1 and C_2 as defined in equations (7) and (8), using a two-step parameter search:

Step 1.1 We perform a parameter sweep over a predefined set Γ_{s1} using a Leave One Out (LOO) cross-validation method, optimizing for AUC (Area Under the ROC Curve).

Step 1.2 The search is refined over a smaller set Γ_{s2}, resulting in a final pair of hyperparameters (C_1^{s2}, C_2^{s2}).

Step 2 With the optimized hyperparameters, we solve the optimization problem using a 10-fold cross-validation approach, and implement the **RFE** method, progressively eliminating subsets of features and evaluating the model's performance.

The computational cost of **RFE** is high since each feature removal step requires solving a new optimization problem. To address this, we introduce the Direct Feature Elimination (**DFE**) method, which significantly reduces the number of optimization problems solved while achieving comparable results. The minimum number of features required by **DFE** to match or exceed **RFE**'s performance, denoted M_{N_f}, is also computed for comparison between the methods.

Through these steps, the **DFE** method reduces the number of optimization problems from 5070 (for **RFE**) to 1260, making it three times faster. All experiments were performed on a high-performance workstation with AMD EPYC 7542 CPUs, 512 GB RAM, using MATLAB 2022b for parallel computation. Average CPU time per core, T_{CPU}, was recorded for each optimization problem.

Finally, we assess the robustness of model solutions to feature perturbations. Using coefficients C_1 and C_2 from previous experiments with the **DFE** method, we generate 100 perturbed feature sets for each dataset and model by applying random perturbations of $\pm\delta\%$ (with $\delta = 5\%$) to the original values. The perturbed solutions are denoted as $\mathbf{w}_{p,i}$ and $b_{p,i}$, while the unperturbed solution is $\mathbf{w}_u$ and b_u. Then, we compute the robustness metric N_{rob} defined as:

$$N_{\text{rob}} = \frac{1}{100}\sum_{i=1}^{100} \frac{\left\| \begin{bmatrix} \mathbf{w}_{p,i} \\ b_{p,i} \end{bmatrix} - \begin{bmatrix} \mathbf{w}_u \\ b_u \end{bmatrix} \right\|_2}{\left\| \begin{bmatrix} \mathbf{w}_u \\ b_u \end{bmatrix} \right\|_2}, \tag{14}$$

which measures the average percentage variation between perturbed and unperturbed solutions.

4.2. Analysis of experimental results

In Tables 2–7, we present the results obtained from the experiments discussed in Section 4.1. For each case, we report the mean AUC values as the number of features is progressively reduced (i.e., considering 100%, 75%, 50%, 25%, 10%, 5%, and 1% of the N_f features), along with M_{N_f} and T_{CPU}. Figures 1–6 display the evolution of the mean AUC values throughout these experiments.

Model	100% N_f	75% N_f	50% N_f	25% N_f	10% N_f	5% N_f	1% N_f	M_{N_f}	T_{CPU}
CDL$_2$+RFE	88.4	88.4	88.4	**89.7**	**89.7**	**89.7**	90.6	-	153
CDL$_2$+DFE	88.4	88.4	**89.7**	**89.7**	86.9	85.6	68.7	474	-
CDL$_0$+RFE	**88.7**	**88.7**	88.7	88.7	88.7	88.7	88.7	-	339
CDL$_0$+DFE	**88.7**	**88.7**	88.7	88.7	88.7	88.7	88.7	**4**	-
MEP$_2$+RFE	88.4	88.4	**89.7**	**89.7**	**89.7**	**89.7**	**90.7**	-	**19**
MEP$_2$+DFE	88.4	88.4	**89.7**	**89.7**	88.4	85.7	65.2	528	-
MEP$_0$+RFE	87.1	87.1	87.1	87.1	87.1	87.1	87.1	-	97
MEP$_0$+DFE	87.1	87.1	87.1	87.1	87.1	87.1	87.1	**4**	-

Table 2: Summary of results for benchmark case $\mathbf{B}_C$, showing mean AUC values at different levels of feature reduction, M_{N_f} for **DFE**, and mean CPU time T_{CPU} for **RFE**. In each column, we highlight in bold the highest mean AUC value and the lowest M_{N_f} and T_{CPU} values.

Model	100% N_f	75% N_f	50% N_f	25% N_f	10% N_f	5% N_f	1% N_f	M_{N_f}	T_{CPU}
CDL$_2$+RFE	74.2	75.0	74.2	74.7	72.5	74.7	76.3	-	686
CDL$_2$+DFE	74.2	74.0	67.4	55.3	50.0	50.0	50.0	2147	-
CDL$_0$+RFE	74.6	74.6	76.1	76.0	76.0	75.0	75.0	-	910
CDL$_0$+DFE	74.6	74.6	74.6	74.6	74.6	74.6	74.6	**16**	-
MEP$_2$+RFE	74.7	75.2	74.2	70.7	68.3	70.4	69.7	-	**65**
MEP$_2$+DFE	74.7	73.2	67.9	55.0	50.0	50.0	50.0	1535	-
MEP$_0$+RFE	**77.1**	76.6	76.2	76.0	75.4	75.4	75.0	-	142
MEP$_0$+DFE	**77.1**	**77.1**	**77.1**	**77.1**	**77.1**	**77.1**	**77.1**	19	-

Table 3: Summary of results for benchmark case $\mathbf{B}_G$, showing mean AUC values at different levels of feature reduction, M_{N_f} for **DFE**, and mean CPU time T_{CPU} for **RFE**. In each column, we highlight in bold the highest mean AUC value and the lowest M_{N_f} and T_{CPU} values.

As shown in the tables and figures, the ℓ_2-norm models (**CDL$_2$** and **MEP$_2$**) and ℓ_0-norm models (**CDL$_0$** and **MEP$_0$**) exhibit distinct behaviors when combined with

Recursive Feature Elimination (**RFE**) or Direct Feature Elimination (**DFE**). The ℓ_0-norm tends to produce sparser models, significantly reducing the number of features without substantially compromising performance, especially with the **DFE** method. Except for the $\mathbf{B}_C$ case, the $\mathbf{CDL}_0$ and $\mathbf{MEP}_0$ models generally achieve better mean AUC values when reducing the number of features, making them more resilient to feature selection.

Model	100% N_f	75% N_f	50% N_f	25% N_f	10% N_f	5% N_f	1% N_f	M_{N_f}	T_{CPU}
CDL_2+RFE	94.2	94.2	94.2	94.2	95.5	**96.7**	93.3	-	262
CDL_2+DFE	94.2	94.2	94.2	94.2	86.6	71.0	50.0	710	-
CDL_0+RFE	95.6	**96.6**	**96.6**	93.9	93.9	93.3	95.6	-	841
CDL_0+DFE	95.6	95.6	95.6	95.6	95.6	95.6	95.6	9	-
MEP_2+RFE	95.5	95.5	95.5	95.5	95.5	95.5	93.4	-	**87**
MEP_2+DFE	95.5	95.5	95.5	94.2	91.6	65.8	50.0	622	-
MEP_0+RFE	**96.6**	93.8	95.0	95.9	**96.6**	96.6	**96.6**	-	319
MEP_0+DFE	**96.6**	**96.6**	**96.6**	**96.6**	**96.6**	96.6	**96.6**	**8**	-

Table 4: Summary of results for benchmark case $\mathbf{B}_L$, showing mean AUC values at different levels of feature reduction, M_{N_f} for **DFE**, and mean CPU time T_{CPU} for **RFE**. In each column, we highlight in bold the highest mean AUC value and the lowest M_{N_f} and T_{CPU} values.

Model	100% N_f	75% N_f	50% N_f	25% N_f	10% N_f	5% N_f	1% N_f	M_{N_f}	T_{CPU}
CDL_2+RFE	60.5	60.5	60.5	61.8	69.5	68.5	68.5	-	744
CDL_2+DFE	60.5	60.5	61.3	54.0	50.0	50.0	50.0	6056	-
CDL_0+RFE	**73.7**	**73.7**	**73.7**	**73.7**	**73.7**	**73.7**	**73.7**	-	1850
CDL_0+DFE	**73.7**	**73.7**	**73.7**	**73.7**	**73.7**	**73.7**	**73.7**	**2**	-
MEP_2+RFE	60.5	60.5	61.8	63.0	69.0	72.0	59.0	-	**492**
MEP_2+DFE	60.5	61.8	62.8	54.0	50.0	50.0	50.0	2308	-
MEP_0+RFE	61.4	61.4	61.4	61.4	61.4	60.4	62.1	-	815
MEP_0+DFE	61.4	61.4	61.4	61.4	61.4	61.4	61.4	5	-

Table 5: Summary of results for benchmark case $\mathbf{B}_P$, showing mean AUC values at different levels of feature reduction, M_{N_f} for **DFE**, and mean CPU time T_{CPU} for **RFE**. In each column, we highlight in bold the highest mean AUC value and the lowest M_{N_f} and T_{CPU} values.

In contrast, the ℓ_2-norm models retain a larger number of features, which can sometimes result in lower efficiency when applying **DFE** compared to **RFE**. This difference is evident in the figures, where the mean AUC values remain more stable when

removing features using the ℓ_0-norm compared to the ℓ_2-norm, which often degrades more quickly as features are eliminated.

To illustrate, focusing on the $\mathbf{B}_G$ case, the **CDL$_0$+DFE** model yielded $M_{N_f} = 16$, while the **CDL$_2$+DFE** model retained $M_{N_f} = 2147$ features. This highlights the advantage of the ℓ_0-norm in reducing the feature set while maintaining a competitive AUC.

Model	100% N_f	75% N_f	50% N_f	25% N_f	10% N_f	5% N_f	1% N_f	M_{N_f}	T_{CPU}
CDL$_2$+RFE	65.0	65.0	68.3	65.0	65.0	65.8	69.2	-	1071
CDL$_2$+DFE	65.0	65.0	66.7	64.1	53.3	48.3	50.0	5128	-
CDL$_0$+RFE	**90.8**	**90.8**	**90.8**	**90.8**	**90.8**	**90.8**	**90.8**	-	500
CDL$_0$+DFE	**90.8**	**90.8**	**90.8**	**90.8**	**90.8**	**90.8**	**90.8**	**1**	-
MEP$_2$+RFE	65.0	65.0	68.3	66.7	67.5	65.8	64.1	-	**494**
MEP$_2$+DFE	65.0	70.8	69.1	61.7	56.7	50.0	50.0	2069	-
MEP$_0$+RFE	**90.8**	**90.8**	**90.8**	**90.8**	**90.8**	**90.8**	**90.8**	-	872
MEP$_0$+DFE	**90.8**	**90.8**	**90.8**	**90.8**	**90.8**	**90.8**	**90.8**	**1**	-

Table 6: Summary of results for benchmark case $\mathbf{B}_W$, showing mean AUC values at different levels of feature reduction, M_{N_f} for **DFE**, and mean CPU time T_{CPU} for **RFE**. In each column, we highlight in bold the highest mean AUC value and the lowest M_{N_f} and T_{CPU} values.

Model	100% N_f	75% N_f	50% N_f	25% N_f	10% N_f	5% N_f	1% N_f	M_{N_f}	T_{CPU}
CDL$_2$+RFE	96.7	96.7	96.7	96.7	96.7	95.0	95.0	-	1085
CDL$_2$+DFE	96.7	96.7	96.7	85.7	58.3	50.0	52.0	4436	-
CDL$_0$+RFE	94.1	94.1	94.1	94.1	94.1	94.1	94.1	-	1812
CDL$_0$+DFE	94.1	94.1	94.1	94.1	94.1	94.1	94.1	**11**	-
MEP$_2$+RFE	93.6	93.6	93.6	94.4	94.4	94.4	96.0	-	409
MEP$_2$+DFE	93.6	92.6	90.0	79.3	62.4	50.0	50.0	5828	-
MEP$_0$+RFE	**97.5**	**97.5**	**97.5**	**97.5**	**97.5**	**97.5**	**97.5**	-	**368**
MEP$_0$+DFE	**97.5**	**97.5**	**97.5**	**97.5**	**97.5**	**97.5**	**97.5**	13	-

Table 7: Summary of results for benchmark case $\mathbf{B}_S$, showing mean AUC values at different levels of feature reduction, M_{N_f} for **DFE**, and mean CPU time T_{CPU} for **RFE**. In each column, we highlight in bold the highest mean AUC value and the lowest M_{N_f} and T_{CPU} values.

In terms of computational time (T_{CPU}), ℓ_2-norm models tend to be faster, particularly when using the **MEP$_2$** approach. This is mainly due to the challenges in

solving the optimization problem for the ℓ_0-norm, which often requires more time to reach an approximate solution. However, this difference in computational effort becomes less significant when using the **DFE** method, which reduces the computational time by a factor of five, making the computational cost more reasonable. Furthermore, when combined with the ℓ_0-norm, **DFE** offers a practical and efficient alternative, producing results similar to **RFE**.

Among the ℓ_0-norm models, $\mathbf{MEP}_0$ consistently outperforms $\mathbf{CDL}_0$ in both mean AUC values and computational time, making it the most effective model for feature selection.

Benchmark	100% N_f	75% N_f	50% N_f	25% N_f	10% N_f	5% N_f	1% N_f	M_{N_f}	T_{CPU}
$\mathbf{B}_C$	90.9	90.9	90.9	90.9	90.9	90.9	90.9	22	30
$\mathbf{B}_G$	79.4	79.4	79.4	79.4	79.4	79.4	80.3	32	122
$\mathbf{B}_L$	95.3	95.3	95.3	95.3	95.3	95.3	95.3	23	133
$\mathbf{B}_P$	74.3	74.3	74.3	74.3	74.3	74.3	74.3	8	565
$\mathbf{B}_W$	83.3	83.3	83.3	83.3	83.3	83.3	83.3	14	423
$\mathbf{B}_S$	97.5	97.5	97.5	97.5	97.5	97.5	97.5	35	590

Table 8: Summary of results for benchmark cases using the ℓ_1-norm model $\mathbf{CDL}_1$+**RFE** presented in [7]. The table shows mean AUC values at different levels of feature reduction, the mean number of selected features M_{N_f} (for **DFE**), and mean CPU time T_{CPU} (for **RFE**). In each column, we highlight in bold the highest mean AUC value and the lowest M_{N_f} and T_{CPU} values.

Overall, the ℓ_0-norm combined with **DFE** provides an efficient trade-off between sparsity of features and computational cost, outperforming the ℓ_2-norm in terms of feature reduction. However, the ℓ_2-norm models may offer more stability in some cases due to their inherent regularization properties.

	$\mathbf{SVM}_0$	$\mathbf{CDL}_0$	$\mathbf{MEP}_0$
$\mathbf{B}_C$	291	27	66
$\mathbf{B}_L$	42	1	33
$\mathbf{B}_W$	3	1	57
$\mathbf{B}_G$	144	89	316
$\mathbf{B}_P$	107	36	43
$\mathbf{B}_S$	470	2	62

Table 9: Values of the robustness metric N_{rob} for models $\mathbf{SVM}_0$, $\mathbf{CDL}_0$, and $\mathbf{MEP}_0$ on each benchmark dataset.

Additionally, it is worth comparing these results with those obtained using the ℓ_1-norm, as presented in [7] and briefly summarized in Table 8, where the $\mathbf{CDL}_1$ method is considered. In that study, ℓ_0-norm results were generally worse, likely

due to the difficulty in achieving good convergence in the optimization algorithms. This suggests that further research into alternative approaches for approximating the ℓ_0-norm is needed to enhance the applicability of this method. Nonetheless, the results presented here are promising and underscore the potential of ℓ_0-norm models.

Finally, we focus on the robustness of models $\mathbf{CDL}_0$ and $\mathbf{MEP}_0$ by computing the robustness metric N_{rob} (defined by (14)) for each benchmark dataset. Besides, we solve these benchmarks using model (1), substituting the ℓ_2-norm with the ℓ_0-norm, which we denote as $\mathbf{SVM}_0$ (see [5]). The values of N_{rob} are presented in Table 9. We observe that $\mathbf{CDL}_0$ generally achieves the lowest N_{rob} values, indicating higher robustness, and in four cases, $\mathbf{MEP}_0$ also demonstrates lower values. This suggests that the proposed models tend to exhibit robust performance against input data perturbations. Similar behavior was observed in [7], where robust models employing ℓ_1- and ℓ_2-norms were considered.

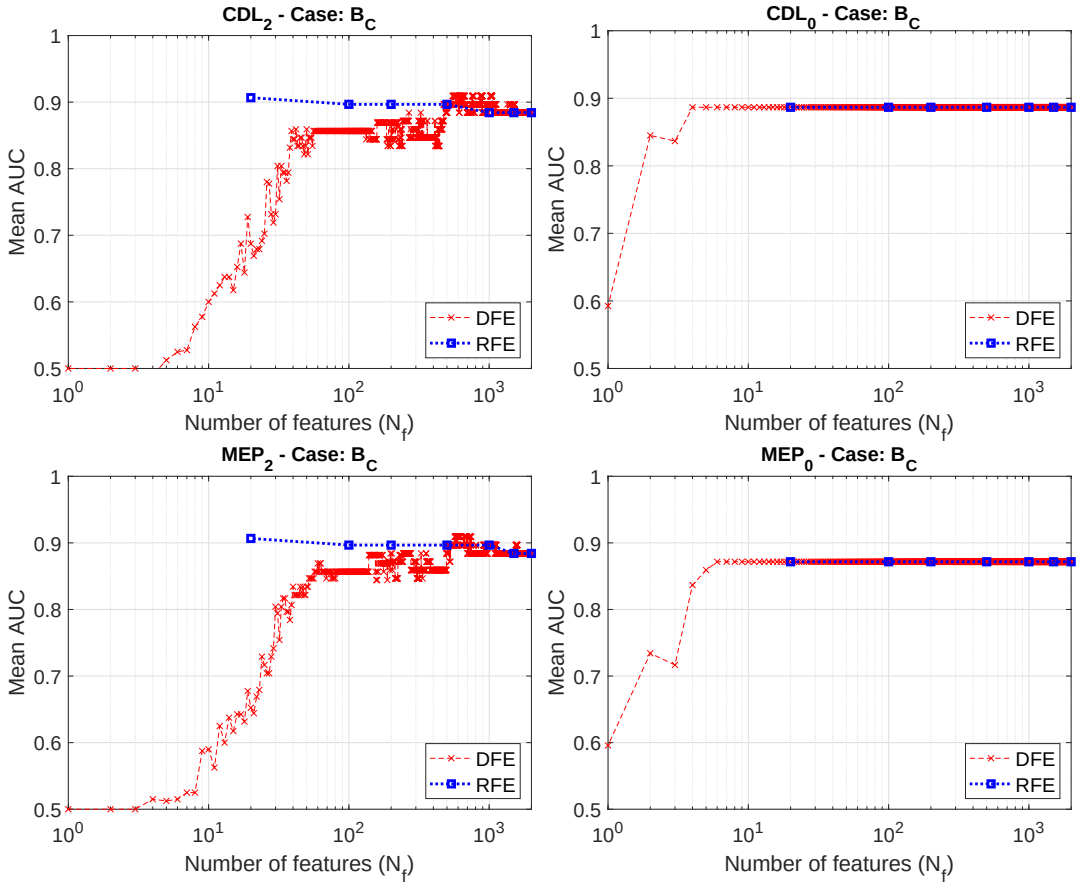

Figure 1: Mean AUC values obtained for the benchmark case $\mathbf{B}_C$ and the algorithms $\mathbf{CDL}_2$, $\mathbf{CDL}_0$, $\mathbf{MEP}_2$ and $\mathbf{MEP}_0$ vs. N_f (in log-scale).

5. Conclusion

In this study, we present robust classification models that integrate feature selection by focusing on ℓ_0-norm regularization methods within SVM-type frameworks. We propose a diagonal algorithm for the numerical solution of these models. By combining this approach with feature elimination techniques such as Recursive Feature Elimination (RFE) and Direct Feature Elimination (DFE), we benchmarked the

models on six well-known datasets. Comparing the results with those obtained from models based on the ℓ_2-norm, we illustrate the advantages and trade-offs associated with both regularization approaches in terms of feature selection and computational efficiency.

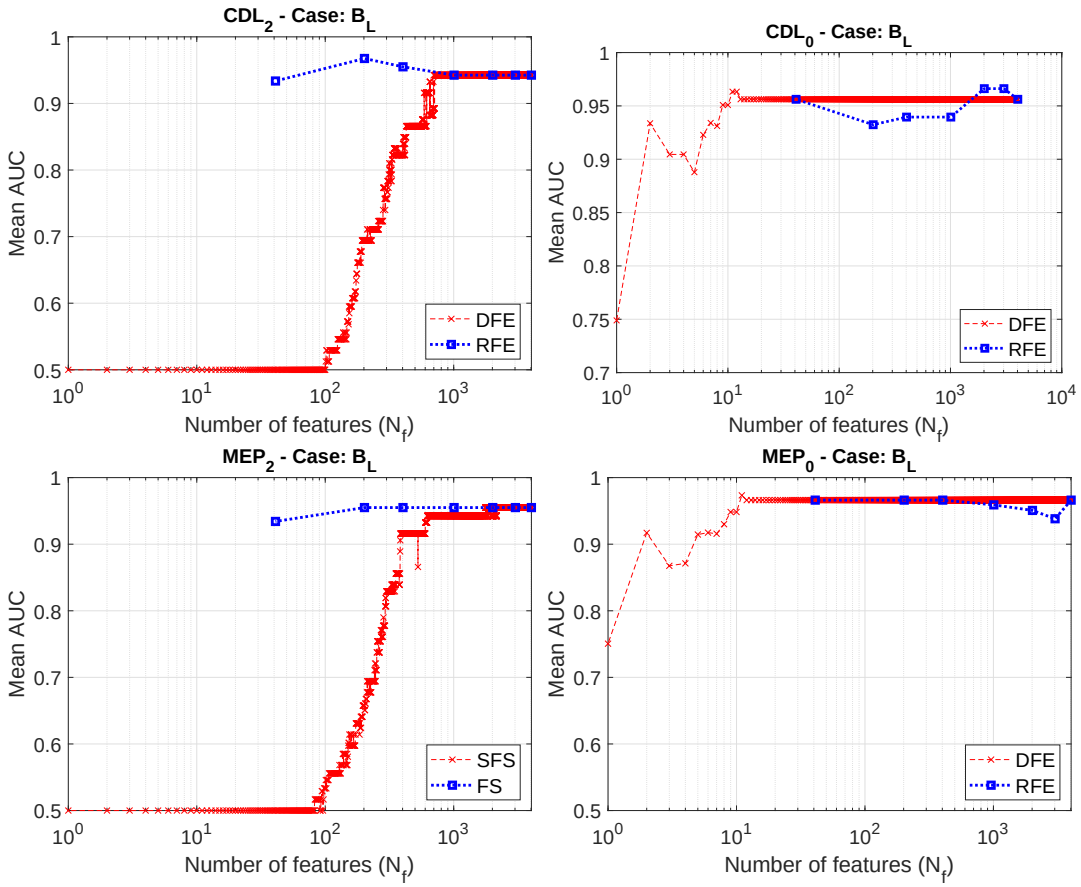

Figure 2: Mean AUC values obtained for the benchmark case $\mathbf{B}_L$ and the algorithms $\mathbf{CDL}_2$, $\mathbf{CDL}_0$, $\mathbf{MEP}_2$ and $\mathbf{MEP}_0$ vs. N_f (in log-scale).

The ℓ_0-norm models, especially when combined with the DFE method, demonstrated superior performance in reducing the number of features without significantly sacrificing classification accuracy. Notably, in most cases, the ℓ_0 models proved more resilient to feature reduction. In contrast, the ℓ_2-norm models tended to retain a larger number of features, resulting in lower efficiency of the DFE method compared to RFE.

Additionally, our results indicate that the ℓ_0-norm models are computationally more expensive, particularly when using the RFE method. However, this computational cost is significantly reduced when employing DFE, which decreases computational time by a factor of five. Among the ℓ_0-norm models, $\mathbf{MEP}_0$ consistently outperformed $\mathbf{CDL}_0$ in both mean AUC values and computational time, establishing it as the most efficient model for feature selection.

Compared to the ℓ_1-norm models discussed in previous works [7], the ℓ_0-norm results are generally less favorable, possibly due to challenges in achieving good convergence in the optimization algorithms. This suggests that further exploration of alternative approaches to approximate the ℓ_0-norm could enhance the applicability of this

model. Nonetheless, our results underline the potential of ℓ_0-norm models as robust and effective tools for feature selection in high-dimensional datasets.

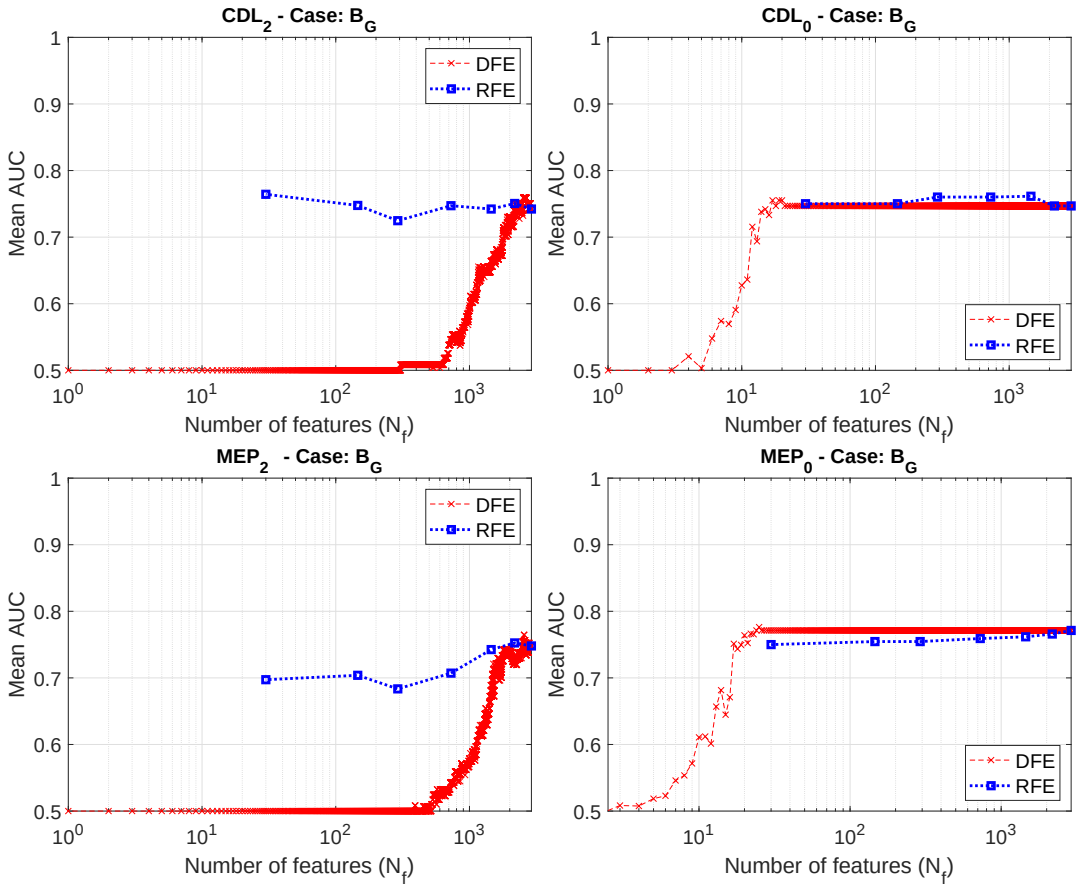

Figure 3: Mean AUC values obtained for the benchmark case $\mathbf{B}_G$ and the algorithms $\mathbf{CDL}_2$, $\mathbf{CDL}_0$, $\mathbf{MEP}_2$ and $\mathbf{MEP}_0$ vs. N_f (in log-scale).

Computing the robustness metric N_{rob} shows that the proposed models $\mathbf{CDL}_0$ and $\mathbf{MEP}_0$ demonstrate enhanced robustness against input data perturbations compared to $\mathbf{SVM}_0$. Notably, $\mathbf{CDL}_0$ consistently achieves the lowest N_{rob} values. These findings suggest that our models perform robustly in the presence of input perturbations, aligning with similar observations in [7].

In conclusion, our study highlights the significance of integrating robust classification models with embedded feature selection using ℓ_0-norm regularization. The proposed diagonal algorithm has been validated through numerical experiments, demonstrating its effectiveness in developing sparse classifiers. Future work could focus on improving optimization techniques for ℓ_0-norm models to achieve better convergence and further enhance their applicability in various domains. Additionally, exploring mixed-norm approaches (combining the ℓ_0-norm with ℓ_1- or ℓ_2-norms) could be highly beneficial for developing models that leverage the strengths of each method.

Acknowledgements. This work was supported by FONDECYT ANID Chile through grants 1230694 and 1251860; by the projects PID2019-106337GB-I00 and PID2023-146754NB-I00, funded by MCIU/AEI/10.13039/501100011033 and FEDER, EU; and by the European M-ERA.Net under project PCI2024-153478. Addi-

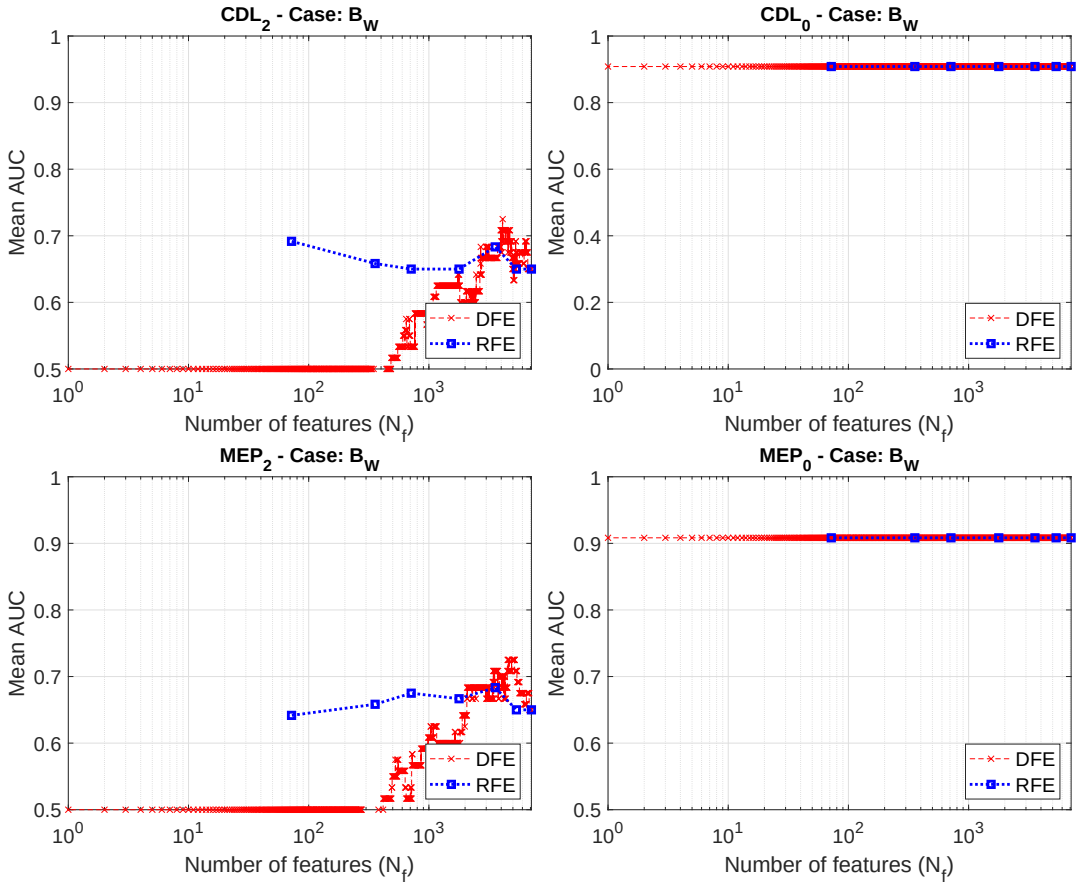

Figure 4: Mean AUC values obtained for the benchmark case $\mathbf{B}_W$ and the algorithms $\mathbf{CDL}_2$, $\mathbf{CDL}_0$, $\mathbf{MEP}_2$ and $\mathbf{MEP}_0$ vs. N_f (in log-scale).

tional support was provided by the 23-MATH-09, MORA-DataS project, and by CONCYTEC through the PROCIENCIA program as part of the "Proyectos de Investigación Básica" competition, under contract number N° PE501087025-2024-PROCIENCIA.

We would like to honor Hédy Attouch, who passed away recently. He was an incredible professor to us and opened pathways toward research in mathematics. In particular, he brought us together, and since then, we have passionately continued our research work. This work is dedicated to him, with our warmest thanks.

References

[1] A. Alizadeh et al.: *Distinct types of diffuse large B-cell lymphoma identified by gene expression profiling*, Nature 403 (2000) 503–511.

[2] F. Alizadeh, D. Goldfarb: *Second-order cone programming*, Math. Program. Ser. B 95/1 (2003) 3–51.

[3] U. Alon, N. Barkai, D. A. Notterman, K. Gish, S. Ybarra, D. Mack, A. J. Levine: *Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays*, Proc. Natl. Acad. Sci. USA 96/12 (1999) 6745–6750.

[4] A. Beck, A. Ben-Tal, L. Tetruashvili: *A sequential parametric convex approximation method with applications to non-convex truss topology design problems*, J. Global Optim. 47/1 (2010) 29–51.

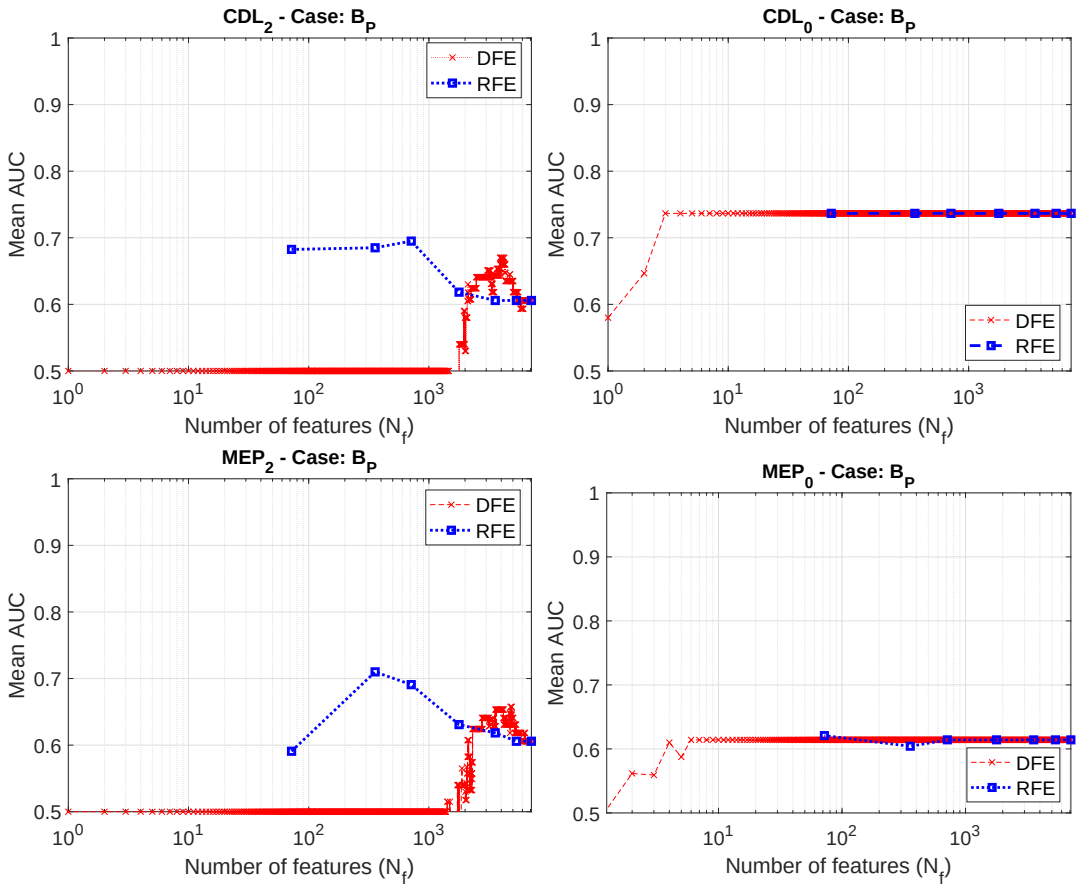

Figure 5: Mean AUC values obtained for the benchmark case $\mathbf{B}_P$ and the algorithms $\mathbf{CDL}_2$, $\mathbf{CDL}_0$, $\mathbf{MEP}_2$ and $\mathbf{MEP}_0$ vs. N_f (in log-scale).

[5] P. Bradley, O. Mangasarian: *Feature selection via concave minimization and support vector machines*, in: Proc. 15th Int. Conf. on Machine Learning (ICML '98), Morgan Kaufmann, San Francisco (1998) 82–90.

[6] A. Canelas, M. Carrasco, J. López: *Application of the sequential parametric convex approximation method to the design of robust trusses*, J. Global Optim. 68/1 (2017) 169–187.

[7] M. Carrasco, B. Ivorra, J. López, A. M. Ramos: *Embedded feature selection for robust probability learning machines*, Pattern Recogn. 159 (2025), art.no. 111157.

[8] W. J. Chen, Y. J. Tian: *L_p-norm proximal support vector machine and its applications*, Procedia Comput. Sci. 1/1 (2010) 2417–2423.

[9] A. Christou, A. Artemiou: *Adaptive ℓ_0 regularization for sparse support vector regression*, Mathematics 11/13 (2023), art.no. 2808.

[10] C. Cortes, V. Vapnik: *Support-vector networks*, Mach. Learn. 20 (1995) 273–297.

[11] M. A. Davenport, M. F. Duarte, Y. C. Eldar, G. Kutyniok: *Introduction to compressed sensing*, in: *Compressed Sensing: Theory and Applications*, Y. C. Eldar et al. (eds.), Cambridge Univ. Press, Cambridge (2012) 1–64.

[12] P. D. Tao, E. B. Souad: *Algorithms for solving a class of non-convex optimization problems. Methods of subgradients*, in: *Fermat Days 85: Mathematics for Optimization*, J.-B. Hiriart-Urruty (ed.), North-Holland Mathematics Studies Vol. 129, North-Holland, Amsterdam (1986) 249–271.

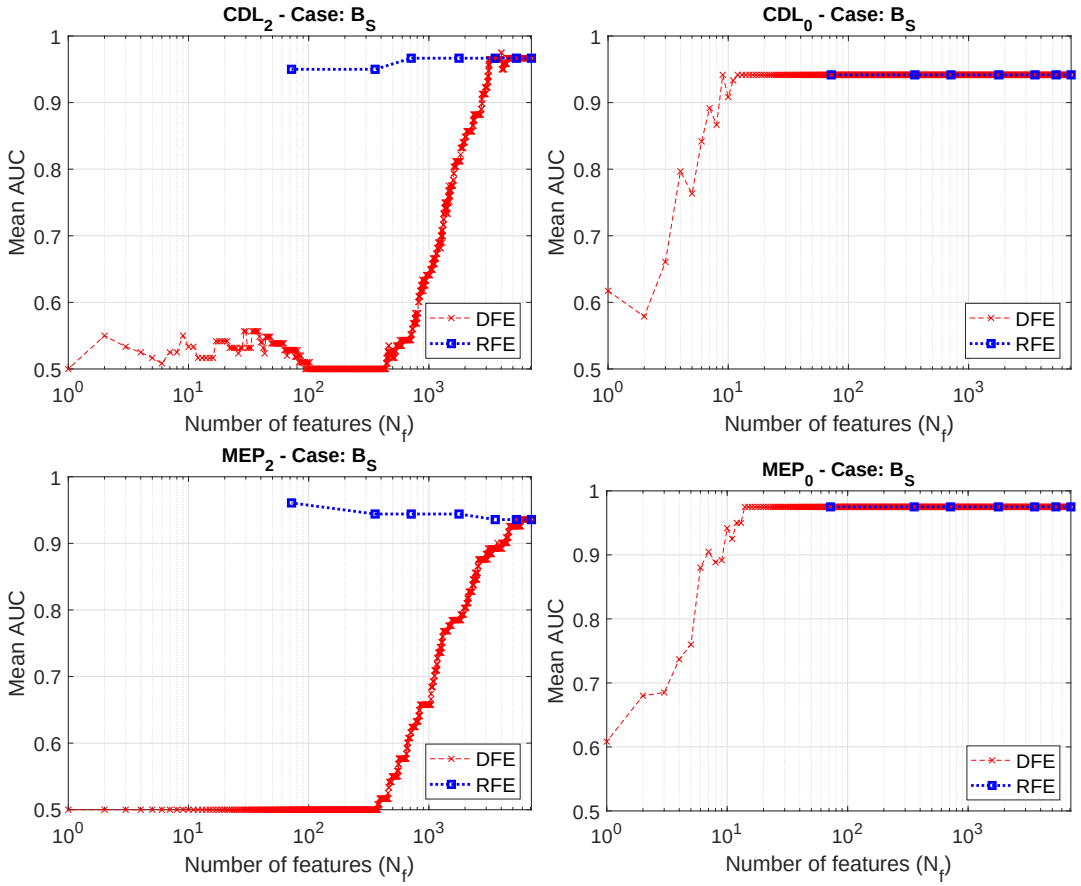

Figure 6: Mean AUC values obtained for the benchmark case $\mathbf{B}_S$ and the algorithms $\mathbf{CDL}_2$, $\mathbf{CDL}_0$, $\mathbf{MEP}_2$ and $\mathbf{MEP}_0$ vs. N_f (in log-scale).

[13] D. L. Donoho, M. Elad, V. N. Temlyakov: *Stable recovery of sparse overcomplete representations in the presence of noise*, IEEE Trans. Inf. Theory 52/1 (2006) 6–18.

[14] R. O. Duda, P. E. Hart, D. G. Stork: *Pattern Classification*, Wiley, New York (2001).

[15] E. Gravier et al.: *A prognostic DNA signature for T1/T2 node-negative breast cancer patients*, Genes Chromosomes Cancer 49/12 (2010) 1125–1134.

[16] I. Guyon, J. Weston, S. Barnhill, V. Vapnik: *Gene selection for cancer classification using support vector machines*, Mach. Learn. 46 (2002) 389–422.

[17] K. Huang, H. Yang, I. King, M. Lyu, L. Chan: *The minimum error minimax probability machine*, J. Mach. Learn. Res. 5 (2004) 1253–1286.

[18] Jayadeva, R. Khemchandani, S. Chandra: *Twin support vector machines for pattern classification*, IEEE Trans. Pattern Anal. Mach. Intell. 29/5 (2007) 905–910.

[19] G. Lanckriet, L. El Ghaoui, C. Bhattacharyya, M. Jordan: *A robust minimax approach to classification*, J. Mach. Learn. Res. 3 (2003) 555–582.

[20] H. A. Le Thi, T. Pham Dinh, M. Thiao: *Efficient approaches for l_2–l_0 regularization and applications to feature selection in SVM*, Appl. Intell. 45/2 (2016) 549–565.

[21] J. López, S. Maldonado, M. Carrasco: *Double regularization methods for robust feature selection and SVM classification via DC programming*, Inf. Sci. 429 (2018) 377–389.

[22] S. Maldonado, M. Carrasco, J. López: *Regularized minimax probability machine*, Knowledge-Based Syst. 177 (2019) 127–135.

[23] S. Maldonado, J. López, M. Carrasco: *The Cobb–Douglas learning machine*, Pattern Recogn. 128 (2022), art.no. 108701.

[24] O. L. Mangasarian: *Exact 1-norm support vector machines via unconstrained convex differentiable minimization*, J. Mach. Learn. Res. 7 (2006) 1517–1530.

[25] J. Neumann, C. Schnörr, G. Steidl: *Combined SVM-based feature selection and classification*, Mach. Learn. 61/1–3 (2005) 129–150.

[26] S. L. Pomeroy et al.: *Prediction of central nervous system embryonal tumour outcome based on gene expression*, Nature 415/6870 (2002) 436–442.

[27] F. Rinaldi, F. Schoen, M. Sciandrone: *Concave programming for minimizing the zero-norm over polyhedral sets*, Comput. Optim. Appl. 46/3 (2010) 467–486.

[28] Y. H. Shao, W. J. Chen, N. Y. Deng: *Nonparallel hyperplane support vector machine for binary classification problems*, Inf. Sci. 263 (2014) 22–35.

[29] M. A. Shipp et al.: *Diffuse large B-cell lymphoma outcome prediction by gene-expression profiling and supervised machine learning*, Nat. Med. 8/1 (2002) 68–74.

[30] J. Tan, Z. Zhang, L. Zhen, C. Zhang, N. Deng: *Adaptive feature selection via a new version of support vector machine*, Neural Comput. Appl. 23 (2013) 937–945.

[31] H. A. Le Thi, T. Pham Dinh, H. M. Le, X. T. Vo: *DC approximation approaches for sparse optimization*, Eur. J. Oper. Res. 244/1 (2015) 26–46.

[32] R. Tibshirani: *Regression shrinkage and selection via the Lasso*, J. R. Stat. Soc. B 58/1 (1996) 267–288.

[33] V. N. Vapnik: *Statistical Learning Theory*, John Wiley & Sons, New York (1998).

[34] L. Wang, J. Zhu, H. Zou: *The doubly regularized support vector machine*, Stat. Sinica 16/2 (2006) 589–615.

[35] M. M. West et al.: *Predicting the clinical status of human breast cancer by using gene expression profiles*, Proc. Natl. Acad. Sci. USA 98/20 (2001) 11462–11467.

[36] 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.

[37] L. Yang, R. Ju: *A DC programming approach for feature selection in the minimax probability machine*, Int. J. Comput. Intell. Syst. 7/1 (2014) 12–24.

[38] J. Zhu, S. Rosset, R. Tibshirani, T. J. Hastie: *1-norm support vector machines*, in: Proc. 16th Neural Information Processing Systems (NIPS'03), MIT Press, Cambridge (2003) 49–56.

[39] H. Zou, T. Hastie: *Regularization and variable selection via the elastic net*, J. R. Stat. Soc. B 67/2 (2005) 301–320.