Strongly agree or strongly disagree?: Rating features in Support Vector Machines

Emilio Carrizosa, Amaya Nogales-Gómez, and Dolores Romero Morales

Journal article (Post print version)

This article was originally published in *Information Sciences*

First published online: 26 September 2015

DOI: 10.1016/j.ins.2015.09.031

Uploaded to Research@CBS: October 2015

Available at: http://research.cbs.dk/da/publications/strongly-agree-or-strongly-disagree%288b1b9406-b65e-4364-865f-fa918176c844%29.html

Available online at www.sciencedirect.com
ScienceDirect
Information Sciences 00 (2015) 1–17
Information Sciences

Strongly agree or strongly disagree?: Rating features in Support Vector Machines☆

Emilio Carrizosa[1], Amaya Nogales-Gómez[2,*], Dolores Romero Morales[3]

Abstract

In linear classifiers, such as the Support Vector Machine (SVM), a score is associated with each feature and objects are assigned to classes based on the linear combination of the scores and the values of the features. Inspired by discrete psychometric scales, which measure the extent to which a factor is in agreement with a statement, we propose the Discrete Level Support Vector Machine (DILSVM) where the feature scores can only take on a discrete number of values, defined by the so-called feature rating levels. The DILSVM classifier benefits from interpretability and it has visual appeal, since it can be represented as a collection of Likert scales, one for each feature, where we rate the level of agreement with the positive class. To construct the DILSVM classifier, we propose a Mixed Integer Linear Programming approach, as well as a collection of strategies to reduce computational cost. Our numerical experiments show that the 3-point and the 5-point DILSVM classifiers have comparable accuracy to the SVM with a substantial gain in interpretability and visual appeal, but also in sparsity, thanks to the appropriate choice of the feature rating levels.

Keywords: **Keywords:** Support Vector Machines, Mixed Integer Linear Programming, Likert scale, interpretability, feature rating level.

1. Introduction

Supervised Classification, [1, 27, 52], plays an important role in many industries, with notable examples being credit scoring [3], fraud detection [14], customer targeting [2, 20], and surveillance [21]. In Supervised Classification, we are given a set of objects Ω partitioned into classes and the aim is to build a procedure for classifying new objects. In its simplest form, each object $i \in \Omega$ has associated a pair (x_i, y_i), where the feature vector x_i takes values on a set $X \subseteq \mathbb{R}^d$ and $y_i \in \{-1, +1\}$ is the class membership of object i.

There are several desirable managerial properties in Supervised Classification methods. Incorporating domain knowledge at the time of constructing the classifier is a very appealing property [14]. One may also like to keep the costs of learning the classifier low, where these costs arise from retrieving the features [9, 47], obtaining feature

☆This work has been partially supported by projects MTM2012-36163, Ministerio de Economía y Competitividad, Spain, and FQM-329 of Junta de Andalucía, Spain, both with EU ERD Funds.

*Most of this work was done when the author was at Departamento de Estadística e Investigación Operativa, Facultad de Matemáticas, Universidad de Sevilla.

Email addresses: ecarrizosa@us.es (Emilio Carrizosa), amaya.nogales.gomez@huawei.com (Amaya Nogales-Gómez), drm.eco@cbs.dk (Dolores Romero Morales)

[1]Departamento de Estadística e Investigación Operativa, Facultad de Matemáticas, Universidad de Sevilla, Sevilla, Spain

[2]Mathematical and Algorithmic Sciences Lab, Huawei France R&D, Paris, France

[3]Department of Economics, Copenhagen Business School, Frederiksberg, Denmark

information from the classifier [46], or refreshing the classifier to include new data [21]. Another desirable property is the comprehensibility/interpretability of the classifier. Classifier interpretability is a major challenge in datasets that contain a huge number of features, where most of these are irrelevant. Since conciseness of the classifier is closely related with its interpretability, much effort has been devoted to increasing its sparsity, i.e., to reducing the number of active features in the classifier [26, 35], as is done with the Lasso, see [32] and [45]. However, constructing comprehensible/interpretable classifiers goes beyond imposing sparsity, [23]. Alternative approaches focus on discretizing the features to detect active ranges of the features [34, 43] or relevant thresholds for the features [10]. When trading off between accuracy and interpretability, another popular approach has been to extract easy–to–understand structures, such as if–then rules, decision trees and decision tables, from powerful black–box type classifiers [3, 5, 37, 38, 40, 41]. Yet another way of achieving interpretability is by constructing discrete linear classifiers [17, 25, 48, 53].

In Supervised Classification, linear classifiers are based on score functions. For instance, the CHADS_2 score [24, 31] is widely used in medicine to predict the risk of stroke in patients with atrial fibrillation based on five different symptoms of the patient. A score is associated with each feature, and objects are assigned to classes based on the linear combination of the scores and the values of the features. The role each feature plays in the classifier is related to the magnitude of the corresponding score, while the sign gives information on how the feature points towards a given class.

A state-of-the-art method in Supervised Classification using a score function is the Support Vector Machine (SVM), [19, 49, 50]. The SVM aims at separating both classes by means of a hyperplane, $\omega^\top x + b = 0$ with $\omega = (\omega_1, \omega_2, \ldots, \omega_d)$, found by solving a Quadratic Programming problem with linear constraints. The classifier obtained by solving the SVM, like any linear classifier, provides valuable information on the role of each feature in the classifier. Indeed, the set of features can be partitioned into three clusters, namely those with positive ω_j, those with negative ω_j, and the ones with ω_j equal to zero. We can say that features in the first group have a positive contribution in the classifier, and thus such features point towards class $+1$, the positive class; similarly, features in the second group have a negative contribution, and thus such features point towards class -1, the negative class; and those features with $\omega_j = 0$ have no contribution in the classifier and are therefore irrelevant. However, the construction of the SVM classifier is aimed at margin maximization (a surrogate of classification accuracy), and interpretability issues are fully disregarded in the formulation.

In this paper, we propose the Discrete Level Support Vector Machine (DILSVM) where the feature scores can only take on a discrete number of levels, defined by the so-called feature rating levels. This is inspired by discrete psychometric scales, [30], and in particular, by Likert scales [33], which measure the extent to which a factor is in agreement with a statement. The DILSVM classifier benefits from interpretability and enjoys the visual appeal of Likert scales. In terms of interpretability, the rating levels can be seen as the intensity of agreement of a feature with the positive class. In terms of visualization, the DILSVM classifier can be represented as a collection of Likert scales, one for each feature, where we rate the level of agreement with the positive class. For feature scores equal to zero, the corresponding Likert scales are redundant and can be eliminated, enhancing the visual appeal of DILSVM. Thus, the performance of DILSVM should be measured using accuracy, but also the number of feature scores equal to zero, i.e., sparsity.

We formulate the DILSVM as a Mixed Integer Linear Programming (MILP) problem. The parameters of our model, namely, the usual tradeoff parameter in SVM and the newly introduced rating levels, may heavily influence the performance of the DILSVM, and therefore, they have to be carefully chosen. Thus, constructing the DILSVM classifier involves solving a series of MILP problems, which, if solved to optimality, may make the overall computational cost high for large datasets. For this reason we propose three strategies to alleviate the computational burden, with different tradeoffs between accuracy and sparsity performance and reduction in computational cost. Such strategies use the guidance of related but simpler optimization models, and reduce the computational cost associated with each parameter vector and/or the number of parameter vectors to be inspected.

In our numerical experiments, we compare the DILSVM against the SVM classifier using ten real-world datasets. Our numerical results show that the appeal of the 3-point and 5-point DILSVM classifiers is threefold. First, we show that the 3-point and 5-point DILSVM classifiers, built using the MILP approach, have comparable accuracy to the SVM. Second, they achieve a substantial gain in sparsity over the SVM due to the small number of rating levels. Third, by the very nature of our procedure, interpretability is definitely improved allowing us to visualize the classification process via a collection of a few Likert scales. The experiments illustrating the performance of the reduction strategies reveal a clear competitiveness in terms of sparsity, while the accuracy depends on the magnitude

of the reduction, being close to the SVM accuracies. In our numerical experiments, we also compare the DILSVM against the 3-point DILSVM classifier with fixed parameters using ten real-world datasets. The 3-point DILSVM with fixed parameters is inspired by the model in [17]. The results illustrate the relevance of tuning parameters, built in our approach, to ensure that accuracy and sparsity are not compromised.

The remainder of this paper is organized as follows. In Section 2 we introduce the DILSVM classifier and give examples that illustrate its visual appeal. In Section 3 we discuss procedures for constructing the DILSVM classifier. In Section 4 we report our numerical experiments using real-world datasets. We end the paper in Section 5 with some conclusions and directions for future research.

2. DILSVM

2.1. The idea

The SVM aims at separating both classes by means of a hyperplane, $\omega^\top x + b = 0$, found by solving the following Quadratic Programming problem with linear constraints:

$$\min_{\omega,b,\xi} \frac{1}{2}\sum_{j=1}^{d}\omega_j^2 + \frac{C}{n}\sum_{i=1}^{n}\xi_i$$

s.t. (SVM)

$$\begin{aligned} y_i(\omega^\top x_i + b) &\geq 1-\xi_i \qquad \forall i = 1,\dots,n \\ \omega &\in \mathbb{R}^d \\ b &\in \mathbb{R} \\ \xi_i &\geq 0 \qquad \forall i = 1,\dots,n, \end{aligned}$$

where n is the size of the sample used to construct the classifier, C is a nonnegative tradeoff parameter, (ξ_i) is the vector of deviation variables, and $\sum_{i=1}^{n}\xi_i$ is the so-called *hinge loss* function. See [12] for a recent review on Mathematical Optimization and the SVM, and [4, 7, 14, 16, 26, 36, 37, 44] for successful applications of the SVM.

The Discrete Level Support Vector Machine (DILSVM) is a variant of the SVM classifier where for each feature j, the score ω_j can only take on a discrete number of values. Let $\mathcal{A} \subset \mathbb{R}$ be a finite set that includes the value 0, which models the marginal impact of the feature on the classifier. We can formulate the DILSVM as the following Discrete Quadratic Programming problem:

$$\min_{\omega,b,\xi} \frac{1}{2}\sum_{j=1}^{d}\omega_j^2 + \frac{C}{n}\sum_{i=1}^{n}\xi_i \tag{1}$$

s.t.

$$y_i(\omega^\top x_i + b) \geq 1-\xi_i \qquad \forall i = 1,\dots,n \tag{2}$$

$$\xi_i \geq 0 \qquad \forall i = 1,\dots,n \tag{3}$$

$$\omega_j \in \mathcal{A} \qquad \forall j = 1,\dots,d \tag{4}$$

$$b \in \mathbb{R}. \tag{5}$$

See [8] for a recent review of algorithmic tools for Mixed Integer Nonlinear Programming.

For adequate choices of $\mathcal{A}$, this model gains in interpretability and visualization. For instance, let us consider $\mathcal{A} = \{-a, 0, a\}$, $a > 0$. In the DILSVM classifier, some of the ω_j will be equal to a or $-a$, while the rest will be equal to zero. Features for which $\omega_j = a$ will have a positive impact on the classifier and therefore will point towards the positive class, those for which $\omega_j = -a$ will point towards the negative class, while the rest will have no impact. We can view this in an alternative way: for a given rating level a, the DILSVM detects those features which *strongly agree* with the positive class, those which *strongly disagree* (and therefore strongly agree with the negative class), and those which are irrelevant to the classifier. This DILSVM classifier can be represented as a collection of 3-point Likert

scales, one for each feature, measuring the extent to which the feature is in agreement with the positive class. When looking for more granularity of the scale, we can increase the size of $\mathcal{A}$. For $\mathcal{A} = \{-a_1, -a_2, 0, a_2, a_1\}$, $a_1 > a_2 > 0$, the DILSVM classifier can be seen as a collection of 5-point Likert scales where features j with $\omega_j = a_1$ are seen to *strongly agree* with the positive class, those with $\omega_j = a_2$ *agree* (but not so strongly), while $\omega_j = -a_1$ ($-a_2$) *strongly disagree* (*disagree*).

2.2. Demonstrations

To illustrate the interpretability and visual appeal of DILSVM, let us consider three datasets which are also used in our numerical experiments in Section 4. The values of the DILSVM parameters are also taken from that section.

The `abalone` dataset has its origins in a Biology study of the abalone population in Tasmania. This is a dataset with a small number of features, 10 in total, used to predict whether an abalone is old. Figure 1 displays a DILSVM classifier as a collection of 5-point Likert scales, where 4 features are irrelevant. Thus, even though the number of features is small, the number of relevant features present in the DILSVM classifier is even smaller. In addition, the DILSVM has the visual appeal of being represented as a collection of a few Likert scales, where the user does not need to get involved with the actual values of the feature scores, just the level of agreement/disagreement with the positive class.

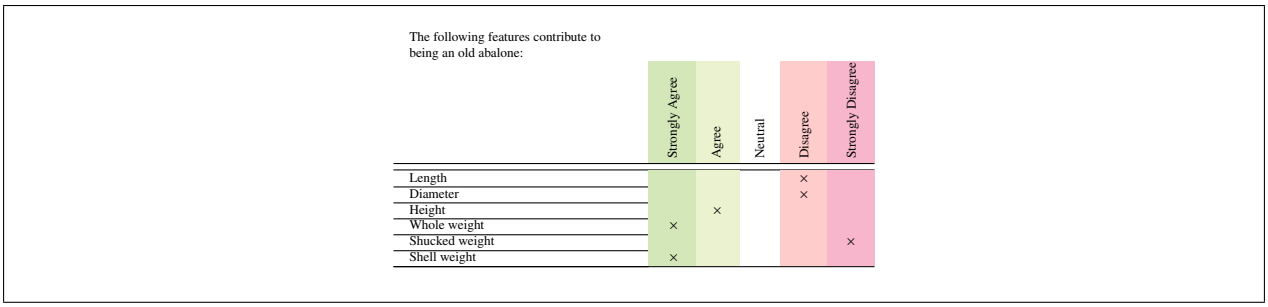

Figure 1: Collection of 5-point Likert scales for `abalone` dataset with the DILSVM; $\mathcal{A} = \{-8, -4, 0, 4, 8\}$ and $C/n = 10^6$; the accuracy in the training set is 78.1%, in the testing set 76.9%, and in the validation set 79.0%; the time to construct the classifier is less than 2 seconds; this model only uses 6 out of 10 features.

The `careval` dataset has 21 features in total, used to predict whether a car is unacceptable. Figure 2 displays a DILSVM classifier as a collection of 3-point Likert scales. Let us focus on three of the features, related to 'Buying price', namely 'very high', 'high', 'medium' and 'low'. The only relevant feature is 'very high', strongly contributing to being an unacceptable car, while for the other three features the score is equal to zero. This is a pattern that can be extended to the overall classifier, where more than half of the features are irrelevant (13 out of 21). The remaining features are equally split between the left and right side of the scale. Thus, in addition to the gain in interpretability and visual appeal, this DILSVM classifier gains in sparsity too.

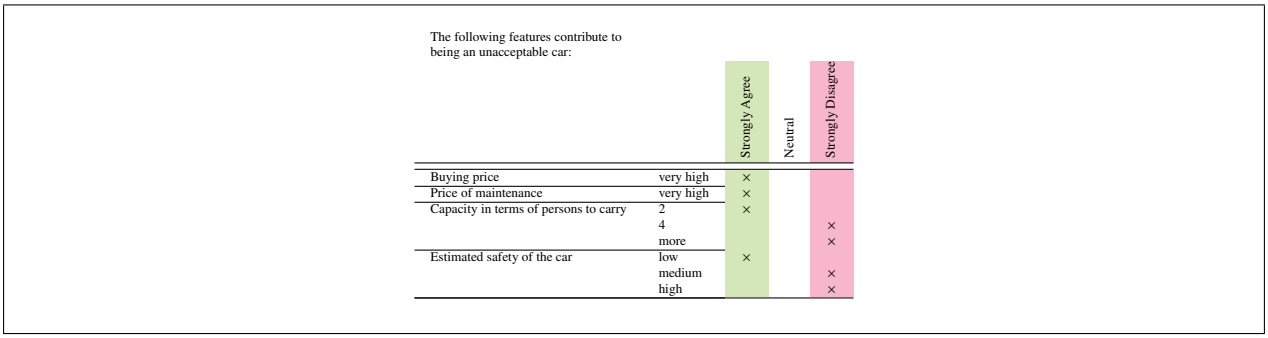

Figure 2: Collection of 3-point Likert scales for `careval` dataset with the DILSVM; $\mathcal{A} = \{-1, 0, 1\}$ and $C/n = 10^{-1}$; the accuracy in the training set is 96.6%, in the testing set 95.9%, and in the validation set 97.5%; the time to construct the classifier is less than 0.1 seconds; this model only uses 8 out of 21 features.

The following features contribute to being a good customer:

		Strongly Agree	Neutral	Strongly Disagree
Status of existing checking account	... ≥ 200 DM	×		
	no checking account	×		
Credit duration				×
Credit history	no credits taken/all credits paid back duly			×
	all credits at this bank paid back duly			×
	critical account/other credits existing (not at this bank)	×		
Purpose	new car			×
	domestic appliances			×
	education			×
	retraining	×		
Credit amount				×
Saving accounts	500 ≤ ... < 1000	×		
	... ≥ 1000	×		
	unknown/no saving accounts	×		
Installment rate in % of disposable income				×
Personal status and sex	male (divorced/separated)			×
Other debtors/guarantors	guarantor	×		
Property	unknown/no property			×
Age		×		
Housing	for free	×		
Number of existing credits at this bank				×
Foreign worker	no	×		

Figure 3: Collection of 3-point Likert scales for `german` dataset with the DILSVM; $\mathcal{A} = \{-1, 0, 1\}$ and $C/n = 10^6$; the accuracy in the training set is 79.0%, in the testing set 71.6%, and in the validation set 75.6%; the optimization model to construct the classifier was aborted after 300 seconds (the time limit); this model only uses 22 out of 63 features.

The well-known German credit scoring dataset, `german` has been used in the context of Supervised Classification and interpretability, such as in [3] where rule extraction techniques are studied. This dataset has 63 features in total, used to predict whether a customer will be *good*. Figure 3 displays a DILSVM classifier as a collection of 3-point Likert scales. Let us focus on three of the features, related to 'Other debtors/guarantors', namely 'having no debtor', 'having a co-applicant' and 'having a guarantor'. The only relevant feature is 'having a guarantor', strongly contributing to being a good customer, while for the other two features the score is equal to zero. This is a pattern that can be extended to the overall classifier, where more than half of the features are irrelevant (41 out of 63). The remaining features are roughly equally split between the left and right side of the scale. Thus, in addition to the gain in interpretability, this DILSVM classifier gains in sparsity too.

Apart from the gain in interpretability, visual appeal and sparsity, once the DILSVM classifier has been obtained, its evaluation (i.e., classifying new objects) is as inexpensive as for the SVM. However, these advantages come with an increased computational burden, related to the choice of the set $\mathcal{A}$, to construct the classifier. We analyze this issue in the next section.

3. Constructing the DILSVM classifier

Inspired by Likert scales, we assume that the set $\mathcal{A}$ is symmetric and defined as $\mathcal{A} = \{-a_1, \ldots, -a_K, 0, a_K, \ldots, a_1\}$, where $a_1 > \ldots > a_K > 0$ are the so-called rating levels telling us about the extent to which each feature is in agreement with the positive class. We denote this model by DILSVM$^{(K)}$. Please note that $\mathcal{A}$ could be considered asymmetric without loss of generality.

Our model involves $K + 1$ parameters, namely the K rating levels as well as the tradeoff parameter C. In the following, we formulate the DILSVM$^{(K)}$, when the $K+1$ parameters are fixed, as an MILP problem. Tuning efficiently the parameters is a challenging problem, [11]. In order to alleviate this computational burden, a collection of strategies is proposed.

3.1. MILP formulation

In this section we formulate (1)–(5) with $\mathcal{A} = \{-a_1, \ldots, -a_K, 0, a_K, \ldots, a_1\}$ as an MILP problem. For each feature j and each rating level a_k, let α_{jk} be equal to either $-1, 1$ or 0, indicating whether ω_j is equal to $-a_k$, a_k or none of

these two. For each feature j, at most one α_{jk} variable can be different from zero. We can now rewrite

$$\begin{aligned} \omega_j &= \sum_{k=1}^{K} a_k \alpha_{jk} \\ \sum_{j=1}^{d} \omega_j^2 &= \sum_{j=1}^{d} \sum_{k=1}^{K} a_k^2 \alpha_{jk}^2 = \sum_{j=1}^{d} \sum_{k=1}^{K} a_k^2 |\alpha_{jk}|, \end{aligned}$$

where the latter follows from the fact that $\alpha_{jk}\alpha_{jk'} = 0$ for $k \neq k'$. It is straightforward to see that by making these substitutions in (1)–(5) and adding the constraints relating to α_{jk}, the DILSVM$^{(K)}$ can be formulated as:

$$\min_{\alpha,b,\xi} \frac{1}{2} \sum_{j=1}^{d} \sum_{k=1}^{K} a_k^2 |\alpha_{jk}| + \frac{C}{n} \sum_{i=1}^{n} \xi_i$$

s.t.

$$\begin{aligned} y_i \left(\sum_{j=1}^{d} \sum_{k=1}^{K} a_k \alpha_{jk} x_{ij} + b \right) &\geq 1 - \xi_i && \forall i = 1, \ldots, n \\ \xi_i &\geq 0 && \forall i = 1, \ldots, n \\ \sum_{k=1}^{K} |\alpha_{jk}| &\leq 1 && \forall j = 1, \ldots, d \\ \alpha_{jk} &\in \{-1,\ 0,\ 1\} && \forall j = 1, \ldots, d,\ \forall k = 1, \ldots, K \\ b &\in \mathbb{R}. \end{aligned}$$

Using the usual approach to transform an absolute value into linear constraints, i.e., $\alpha_{jk} = \alpha_{jk}^{+} - \alpha_{jk}^{-}$, and $|\alpha_{jk}| = \alpha_{jk}^{+} + \alpha_{jk}^{-}$, with $\alpha_{jk}^{+}, \alpha_{jk}^{-} \in \{0, 1\}$, we can reformulate the DILSVM$^{(K)}$ as an MILP problem:

$$\min_{\alpha^{+},\alpha^{-},b,\xi} \frac{1}{2} \sum_{j=1}^{d} \sum_{k=1}^{K} a_k^2 (\alpha_{jk}^{+} + \alpha_{jk}^{-}) + \frac{C}{n} \sum_{i=1}^{n} \xi_i \tag{6}$$

s.t. (DILSVM$^{(K)}$)

$$y_i \left(\sum_{j=1}^{d} \sum_{k=1}^{K} a_k (\alpha_{jk}^{+} - \alpha_{jk}^{-}) x_{ij} + b \right) \geq 1 - \xi_i \qquad \forall i = 1, \ldots, n \tag{7}$$

$$\xi_i \geq 0 \qquad \forall i = 1, \ldots, n \tag{8}$$

$$\sum_{k=1}^{K} (\alpha_{jk}^{+} + \alpha_{jk}^{-}) \leq 1 \qquad \forall j = 1, \ldots, d \tag{9}$$

$$\alpha_{jk}^{+}, \alpha_{jk}^{-} \in \{0,\ 1\} \qquad \forall j = 1, \ldots, d,\ \forall k = 1, \ldots, K \tag{10}$$

$$b \in \mathbb{R}. \tag{11}$$

This MILP formulation has $n + d$ constraints, $2dK$ binary variables, n nonnegative variables and 1 free variable, while the SVM formulation has a quadratic objective function but all the variables are continuous.

In terms of the choice of the number of rating levels, the DILSVM$^{(K)}$ may lead to a higher classification accuracy than the DILSVM$^{(K')}$, with $K > K'$, but it is computationally more expensive as the number of binary decision variables increases and the gain in interpretability, visual appeal and sparsity are less dramatic. Similar observations can be made when comparing the SVM and the DILSVM$^{(K)}$, since at the end of the spectrum is the SVM, which can be seen as DILSVM$^{(\infty)}$. In the tradeoff between accuracy and interpretability, we recommend the $K = 1$ and $K = 2$

Step 1. For each C,

(i) Solve the SVM and obtain the (partial) optimal solution (ω, b).

(ii) For each $(a_1, \ldots, a_K)$,

For $j = 1, \ldots, d$

For $k = 1, \ldots, K$, set $\beta^+_{jk} = \beta^-_{jk} = 0$

For $k = 1, \ldots, K$

If $a_k \leq \omega_j < a_{k-1}$ then $\beta^+_{jk} = 1$

ElseIf $-a_{k-1} < \omega_j \leq -a_k$ then $\beta^-_{jk} = 1$

end

end

Return the classifier $\sum_{k=1}^{K} a_k(\beta^+_k - \beta^-_k)^\top x + b$

end

Step 2. Choose the optimal parameter vector using $\sum_{k=1}^{K} a_k(\beta^+_k - \beta^-_k)^\top x + b$.

Figure 4: Pseudocode for the *SVM rounding* strategy.

versions of the model, namely the DILSVM$^{(1)}$ and the DILSVM$^{(2)}$.

3.2. Reduction strategies

We have formulated the DILSVM$^{(K)}$ as an MILP problem. Solving this MILP formulation to optimality is an $\mathcal{NP}$-Hard task even if we only consider one single rating level, namely $K = 1$ and $a_1 = 1$, and $C = \infty$. This $\mathcal{NP}$-Hardness result has been shown in [17], where a Structural Risk Minimization analysis, similar to the one available for the SVM, is also provided. In addition, the performance of the DILSVM$^{(K)}$ classifier may be strongly influenced by the choice of the tradeoff parameter C as well as the K rating levels. Thus, constructing the DILSVM classifier involves solving a series of MILPs, which, if solved to optimality, may make the overall computational cost high for large datasets. In this section we present three different strategies to alleviate the computational burden of constructing the DILSVM$^{(K)}$ classifier.

The proposed strategies use the guidance of related but simpler optimization models, and reduce the computational cost associated with each parameter vector $(C, a_1, \ldots, a_K)$ and/or the number of parameter vectors to be inspected. The first strategy is based on rounding the SVM classifier using each vector of rating levels. The second strategy proposes, for each parameter vector, the randomized rounding, [42], of the Linear Programming (LP) relaxation of the DILSVM$^{(K)}$. The third strategy aims at speeding up the process for the DILSVM$^{(K)}$ based on information readily available for the DILSVM$^{(K')}$, $K' < K$. Clearly, these strategies are of diverse nature and, as the numerical experiments will illustrate, offer different tradeoffs between performance and reduction in computational cost.

The first and second strategies will be presented for general K, while, and for the sake of clarity, the third strategy will be described for $K = 2$, though the process gracefully extends to arbitrary K. Apart from a grid of parameter vectors, we assume that we are given a selection criterion, [12], to choose the optimal classifier among those generated for each vector of the grid. This is usually the accuracy on an independent sample different from the one used to construct the classifier, see Section 4.1 for more details.

The first strategy, the *SVM rounding* (*RSVM*), uses the guidance of the SVM formulation. It is based on rounding the feature scores of the SVM classifier using the rating levels. For each value of C, the SVM classifier is obtained, and the rounding procedure is performed using each vector of rating levels. The pseudocode of this reduction strategy is given in Figure 4, where $a_0 = \infty$. While this strategy examines all the parameter vectors in the grid, it is appealing since it is cheaper to train an SVM than a DILSVM. In addition, for a given value of C, once the SVM classifier has been obtained, the rounding procedure takes $O(dK)$ time for each vector of rating levels.

Step 1. For each $(C, a_1, \ldots, a_K)$,

(i) Solve the LP relaxation of DILSVM$^{(K)}$ and obtain the (partial) optimal solution (α^+, α^-).

(ii) For $j = 1, \ldots, d$

For $k = 1, \ldots, K$, set $\beta^+_{jk} = \beta^-_{jk} = 0$

Set $\mathcal{K} = \{1, \ldots, K\}$

while $(\mathcal{K} \neq \emptyset)$

Let $\bar{k}$ such that $\max\{\alpha^+_{j\bar{k}}, \alpha^-_{j\bar{k}}\} \geq \max\{\alpha^+_{jk}, \alpha^-_{jk}\},\ \forall k \in \mathcal{K}$

Set $\beta^+_{j\bar{k}} = \mathtt{rand}(\alpha^+_{j\bar{k}})$

If $\beta^+_{j\bar{k}} = 1$, set $\mathcal{K} = \emptyset$

Else

Set $\beta^-_{j\bar{k}} = \mathtt{rand}(\alpha^-_{j\bar{k}})$

If $\beta^+_{j\bar{k}} = \beta^-_{j\bar{k}} = 0$, set $\mathcal{K} = \mathcal{K} \setminus \{\bar{k}\}$

Else $\mathcal{K} = \emptyset$

end

end

(iii) Return the classifier $\sum_{k=1}^{K} a_k(\beta^+_k - \beta^-_k)^\top x + b$.

Step 2. Choose the optimal parameter vector using $\sum_{k=1}^{K} a_k(\beta^+_k - \beta^-_k)^\top x + b$.

Figure 5: Pseudocode for the *randomized rounding* strategy.

The second strategy, the *randomized rounding* (*RR*) strategy, uses the guidance of the LP relaxation of the DILSVM$^{(K)}$. It performs a randomized rounding procedure using a partial solution to the LP relaxation of the DILSVM$^{(K)}$. For each parameter vector $(C, a_1, \ldots, a_K)$, the *RR* strategy solves the LP relaxation of the DILSVM$^{(K)}$, where constraints (10) are relaxed to $\alpha^+_{jk}, \alpha^-_{jk} \in [0, 1]$. Let (α^+, α^-) be the (partial) optimal solution obtained. Without loss of optimality, $\alpha^+_{jk} \cdot \alpha^-_{jk} = 0$. Thus, for a given feature j, α^+_{jk} (α^-_{jk}) can be seen as the desirability of setting the score of feature j to value a_k ($-a_k$). Noting that the assignment constraints (9) are satisfied, a randomized rounding procedure can be applied to derive the DILSVM classifier. In order to ensure feasibility with respect to the assignment constraints, the rounding needs to take place in a predetermined order of the rating levels, such that once a rating level and a sign has been assigned to a feature, we move to the next feature. In the current version of the *RR* strategy, the rating levels are arranged in decreasing order of $\max\{\alpha^+_{jk}, \alpha^-_{jk}\}$. The pseudocode of this reduction strategy can be found in Figure 5, where $\mathtt{rand}(p)$ is a subroutine of random numbers generation, returning the value 1 with probability p and 0 otherwise.

In the third strategy, the *fixing* strategy, we use the output of the DILSVM$^{(1)}$ classifier to alleviate the burden of constructing the DILSVM$^{(2)}$ classifier. In this case, the reduction is twofold. First, the size of the parameter space is narrowed down. Second, some of the decision variables in the MILP formulation (6)–(11) are fixed in advanced, and therefore eliminated. The pseudocode for this reduction strategy can be found in Figure 6. Using the grid corresponding to parameters C and a_1, we first construct the DILSVM$^{(1)}$ classifier yielding optimal parameters values $C^{(1)}$ and $a^{(1)}_1$. We then construct the DILSVM$^{(2)}$ classifier using the MILP formulation (6)–(11) for $K = 2$, with $C = C^{(1)}$, $a_1 = a^{(1)}_1$ and the values of a_2 in the grid. In addition, we reduce the number of binary decision variables in the MILP formulation. In the current version of the *fixing* strategy, feature j will have a strong positive rating in the DILSVM$^{(2)}$ classifier if that was the case in the DILSVM$^{(1)}$ classifier, and similarly for a strong negative rating. For the rest of features, β^+_{jk} and β^-_{jk}, $k = 1, 2$, must be optimized.

As already mentioned, the *fixing* strategy can be extended to an arbitrary K, where we use the output of the

DILSVM$^{(K')}$ classifier with $K' < K$.

Step 1. For each (C, a_1),

(i) Solve the DILSVM$^{(1)}$ and obtain the (partial) optimal solution (α^+, α^-, b).

(ii) Choose the optimal parameter vector $(C^{(1)}, a_1^{(1)})$ using $a_1(\alpha^+ - \alpha^-)^\top x + b$, and obtain $(\bar{\alpha}^+, \bar{\alpha}^-)$.

Step 2. For $C^{(1)}, a_1^{(1)}$, and for each a_2,

(i) For $j = 1, \ldots, d$

If $\bar{\alpha}^+_{j1} = 1$ then $\beta^+_{j1} = 1$ and $\beta^-_{j1} = \beta^+_{j2} = \beta^-_{j2} = 0$

ElseIf $\bar{\alpha}^-_{j1} = 1$ then $\beta^-_{j1} = 1$ and $\beta^+_{j1} = \beta^+_{j2} = \beta^-_{j2} = 0$

end

(ii) Solve the DILSVM$^{(2)}$ and return the classifier $\sum_{k=1}^{2} a_k(\beta_k^+ - \beta_k^-)^\top x + b$.

Step 3. Choose the optimal parameter vector using $\sum_{k=1}^{2} a_k(\beta_k^+ - \beta_k^-)^\top x + b$.

Figure 6: Pseudocode for the *fixing* strategy.

4. Numerical experiments

In this section we illustrate the performance of the DILSVM classifier against that of SVM. For feature scores equal to zero, the corresponding Likert scales in the DILSVM classifier are redundant and can be eliminated, enhancing its visual appeal. Thus, we use accuracy and sparsity as performance criteria, where the latter is measured as the percentage of features with zero score. Although there are other modifications of SVM pursuing interpretability and sparsity, see e.g. [10, 13, 26, 35, 37, 48], as far as the authors are aware, none of them pursue the important topic of visualization.

To construct the DILSVM classifier, we use four approaches, namely the MILP approach and the three reduction strategies. We will show that, with small values of K ($K = 1, 2$) to ensure interpretability and visual appeal, the DILSVM$^{(K)}$ is competitive against the SVM in terms of accuracy, being substantially sparser than the latter. Inspired by the model in [17], we show results for the DILSVM$^{(1)}$ with fixed parameters $C = \infty$ and $a_1 = 1$. Comparing the DILSVM$^{(1)}$ with it shows the relevance of tuning parameters. The accuracy results for the SVM and the MILP approach for DILSVM$^{(1)}$ and DILSVM$^{(2)}$ are reported in Table 2, while the sparsity results are reported in Table 3. For each dataset and each performance criterion, we underline the best results across the three approaches. The accuracy performance of the reduction strategies to construct the DILSVM classifier is reported in Table 4, while the sparsity performance can be found in Table 5. (Note that the *fixing* strategy only applies to $K \geq 2$, and therefore it appears only once in Tables 4 and 5.) Again, we underline the best results across the reduction strategies for each dataset and each performance criterion. Figures 8–17 help to visualize the results presented in this section.

Our experiments have been conducted on a PC with an Intel® Core™ i7 processor, 16 Gb of RAM. We use the optimization engine CPLEX v12.4, [18], for solving all optimization problems. We have set the time limit to 300 seconds, which is enough for most of the problems to be solved to optimality. For the remaining ones, since the optimization process is prematurely aborted, the so-obtained solution is heuristic. The rest of this section is structured as follows. The tuning procedure used to choose the tradeoff parameter C and the rating levels a_k, $k = 1, \ldots, K$, is given in Section 4.1. The datasets used to compare the models are described in Section 4.2. The numerical experiments are presented in Sections 4.3 and 4.4.

Step 1. For each value of the parameter vector $(C, a_1, \ldots, a_K)$ in the grid,

(i) Obtain the corresponding classifier $\sum_{k=1}^{K} a_k(\alpha_k^+ - \alpha_k^-)^\top x + b$ using the training set.

(ii) Let $\pi^{\text{test}}(C, a_1, \ldots, a_K)$ be the accuracy of $\sum_{k=1}^{K} a_k(\alpha_k^+ - \alpha_k^-)^\top x + b$ in the testing set.

Step 2. Let $(\bar{C}, \bar{a}_1, \ldots, \bar{a}_K) \in \arg\max_{(C,a_1,\ldots,a_K)} \pi^{\text{test}}(C, a_1, \ldots, a_K)$ and $\sum_{k=1}^{K} \bar{a}_k(\bar{\alpha}_k^+ - \bar{\alpha}_k^-)x^\top + b$ be the corresponding classifier using the training set.

(i) Report the performance of $\sum_{k=1}^{K} \bar{a}_k(\bar{\alpha}_k^+ - \bar{\alpha}_k^-)^\top x + b$ in the validation set.

Figure 7: Pseudocode of the tuning procedure for the DILSVM$^{(K)}$.

4.1. Parameter settings

As customary in Supervised Classification, the construction of the DILSVM classifier calls for tuning some parameters, namely the tradeoff parameter C as well as the rating levels. The tuning procedure works as follows, [11, 12]. The dataset is split into three sets, the so-called training, testing and validation sets. For each vector of parameters, the DILSVM is run on the training set, which has size n. The different classifiers built in this way are compared according to their accuracy on the testing set. The vector of parameters with the highest accuracy on the testing set is chosen, and its accuracy and sparsity on the validation set are reported, see Figure 7.

Following the usual approach, for the DILSVM$^{(1)}$, parameters C and a_1 are tuned by inspecting a grid of the form $\frac{C}{n} \in \{10^{-6}, \ldots, 10^6\}$ and of the form $a_1 \in \{2^0, \ldots, 2^{10}\}$. For the DILSVM$^{(2)}$, C and a_1 are tuned with the same grid, and $a_2 \in \{\frac{a_1}{2}, \frac{a_1}{2^2}\}$. Due to the low dimensionality of the parameter space, such straightforward strategy was sufficient, and we did not need to use more sophisticated tuning procedures, as those described in [11].

4.2. Datasets

The performance, in terms of accuracy and sparsity, of our model is illustrated using 10 real-world large datasets: `calhous` from the StatLib repository [51], `ijcnn1` and `cod-rna` from the LIBSVM repository [15], and the remaining ones from the UCI repository [6]. Datasets containing categorical features (`mushroom`, `german`) have been transformed splitting the categories into binary features. Regression datasets (`abalone`, `calhous`) are transformed into 2-class datasets using the median, and multi-class datasets (`careval`, `shuttle`) are transformed into 2-class, treating the largest class as class +1 and the remaining classes as class −1. As customary, the features have been normalized. With this, we can safely assume that there is one single rating scale for all the features, and thus for the scores, although this means that DILSVM is built on the normalized, as opposed to the original, data.

A description of these datasets can be found in Table 1, whose first four columns report the dataset name, full name given in the repository, number of features (d) and total size of the dataset ($|\Omega|$). The size of the training set (n) is set as the closest $5 \cdot 10^{s-1}$ multiple to $|\Omega|/2$ where $|\Omega|$ is of 10^s order, see fifth column of Table 1, and the remaining records in the dataset are equally split between the testing and validation sets. The last column reports the class split in the training set.

To obtain sharp estimates for the accuracy and the sparsity, repeated random subsampling is used, where ten instances are run for each dataset. The ten instances differ in the seed used to reshuffle the dataset and then obtain different training, testing and validation sets. As customary in the literature, for each dataset, we report the mean accuracy in the validation set across the ten instances, as well as the standard deviation. For each dataset and each criteria, we report the mean performance in the validation set across the ten instances, as well as the standard deviation.

Table 1. Datasets.

Name	Name in Repository	d	$\lvert\Omega\rvert$	n	Class split (in %)
adult	Adult	123	30956	15000	24/76
mushroom	Mushroom	119	8124	4000	48/52
german	Statlog (German Credit Data)	63	1000	500	30/70
ijcnn1	ijcnn1	22	35000	20000	9/91
careval	Car Evaluation	21	1728	1000	30/70
gamma	MAGIC Gamma Telescope	10	19020	10000	32/68
abalone	Abalone	10	4177	2000	50/50
shuttle	Statlog (Shuttle)	9	58000	30000	20/80
cod-rna	cod-rna	8	59535	30000	33/67
calhous	California Housing	8	20460	10000	50/50

4.3. Results for the MILP approach

In this section we compare the performance of the DILSVM$^{(1)}$ and the DILSVM$^{(2)}$ against that of the SVM, where the DILSVM results are generated using the MILP formulation in Section 3.1. Performance will be measured in terms of two criteria, accuracy and sparsity. The ideal model would be that one achieving the highest values in both criteria. When, for a given criterion, the difference in performance between two approaches is 1 percentage point (p.p.) or below, we will say that both approaches are comparable under that criterion.

Recall that Table 2 reports the accuracy performance of the SVM, and the MILP approach results for the DILSVM$^{(1)}$ and DILSVM$^{(2)}$. Similarly, Table 3 reports their sparsity performance.

4.3.1. Accuracy

We start with the analysis of the accuracy. Figure 8 shows that the DILSVM is competitive against the SVM and that the tuning of parameters is an essential task. We first compare with the accuracy of the SVM. For three datasets, adult, german and careval, the DILSVM$^{(1)}$ outperforms the SVM by 5.5, 2.3 and 1.1 p.p., respectively. For three datasets, mushroom, gamma and shuttle, the DILSVM$^{(1)}$ and the SVM are comparable. In three datasets, ijcnn1, abalone and calhous, the SVM outperforms the DILSVM$^{(1)}$ by 1.2, 1.2 and 1.6 p.p., respectively. In cod-rna, the DILSVM$^{(1)}$ is clearly inefficient compared to the SVM. For datasets such as cod-rna, the DILSVM$^{(1)}$ is too restrictive, and the accuracy may benefit from the additional flexibility built in by the DILSVM$^{(2)}$. An improvement of more than 1 p.p. can be observed from the DILSVM$^{(1)}$ to the DILSVM$^{(2)}$ in three datasets. The first one is careval, for which the DILSVM$^{(1)}$ is already better than the SVM, and the improvement on the SVM increases from 1.1 to 3.6 p.p. The second one is cod-rna, where now the difference in accuracy between the SVM and the DILSVM has been reduced from 15.7 to 3.3 p.p. The third one is calhous, where the SVM is better than the DILSVM$^{(1)}$, but now the DILSVM$^{(2)}$ has a comparable performance to the SVM. Across all datasets, the DILSVM$^{(2)}$ outperforms the SVM in three datasets, they are comparable in five datasets, while the SVM outperforms the DILSVM$^{(2)}$ in two datasets. Thus, the DILSVM$^{(2)}$ is competitive against the SVM in terms of accuracy. We now show the relevance of tuning parameters by comparing accuracy results between the DILSVM$^{(1)}$ and the DILSVM$^{(1)}$ with fixed parameters $C = \infty$ and $a_1 = 1$. Using this criterion, the DILSVM$^{(1)}$ with fixed parameters is not competitive and is outperformed by the DILSVM$^{(1)}$ in eight datasets, where the increase in accuracy achieved ranges between 2.8 and 18.4 p.p., while the accuracy of both methods is the same for mushroom and ijcnn1. This shows that a fine tuning of the parameters in the DILSVM$^{(1)}$, as proposed in this paper, strongly improves the performance of the so-obtained classifier.

4.3.2. Sparsity

We now focus on the second criterion under consideration, namely, sparsity. Figure 9 shows that the DILSVM is the best in terms of sparsity. Indeed, for each dataset, the best model is the DILSVM$^{(1)}$, followed by the DILSVM$^{(2)}$, and the SVM being the worse. Note that the sparsity performance of the SVM is rather poor, being always fully dense (except for german), i.e., all features have nonzero coefficients, and therefore play a role in the classifier. The DILSVM$^{(1)}$ with fixed parameters is clearly outperformed by the DILSVM$^{(1)}$, except for cod-rna, in which the sparsity is exactly the same for both methods. We now take a closer look at the sparsity of our model and show how

Table 2. Accuracy in the validation set in % with $C/n \in \{10^{-6}, \ldots, 10^{6}\}$: the SVM and the MILP approach for the DILSVM$^{(1)}$ and DILSVM$^{(2)}$.

Name	*SVM*		DILSVM$^{(1)}$				DILSVM$^{(2)}$	
			MILP $C = \infty, a_1 = 1$		*MILP* $a_1 \in \{2^0, \ldots, 2^{10}\}$		*MILP* $a_1 \in \{2^0, \ldots, 2^{10}\}$ $a_2 \in \{\frac{a_1}{2}, \frac{a_1}{2^2}\}$	
	mean	std	mean	std	mean	std	mean	std
adult	84.9	0.3	82.9	0.5	90.4	0.6	90.6	0.5
mushroom	100.0	0.0	100.0	0.0	100.0	0.0	100.0	0.0
german	73.0	2.4	72.5	2.1	75.3	5.7	76.0	5.3
ijcnn1	91.4	0.4	90.2	0.4	90.2	0.4	90.2	0.4
careval	95.1	0.9	86.9	2.5	96.2	2.0	98.7	0.4
gamma	79.3	0.6	70.4	0.6	79.1	0.6	79.3	0.7
abalone	79.5	1.1	74.3	0.8	78.3	1.0	79.1	1.1
shuttle	98.2	0.1	80.8	1.3	97.3	0.1	97.6	0.3
cod-rna	93.9	0.1	66.6	0.4	78.2	0.3	90.6	0.9
calhous	83.6	0.3	63.6	2.5	82.0	0.5	83.0	0.2

Table 3. Sparsity in the validation set in % with $C/n \in \{10^{-6}, \ldots, 10^{6}\}$: the SVM and the MILP approach for the DILSVM$^{(1)}$ and DILSVM$^{(2)}$.

Name	*SVM*		DILSVM$^{(1)}$				DILSVM$^{(2)}$	
			MILP $C = \infty, a_1 = 1$		*MILP* $a_1 \in \{2^0, \ldots, 2^{10}\}$		*MILP* $a_1 \in \{2^0, \ldots, 2^{10}\}$ $a_2 \in \{\frac{a_1}{2}, \frac{a_1}{2^2}\}$	
	mean	std	mean	std	mean	std	mean	std
adult	0.0	0.0	55.9	9.7	79.3	8.8	77.4	11.6
mushroom	0.0	0.0	49.1	3.3	91.6	0.0	91.6	0.0
german	4.1	1.1	51.4	7.3	71.0	12.2	55.9	22.1
ijcnn1	0.0	0.0	81.8	22.3	100.0	0.0	95.5	13.6
careval	0.0	0.0	30.0	15.2	59.5	5.7	35.7	2.4
gamma	0.0	0.0	31.0	5.4	50.0	11.0	50.0	16.7
abalone	0.0	0.0	15.0	5.0	70.0	8.9	55.0	9.2
shuttle	0.0	0.0	15.6	7.4	63.3	5.1	45.6	13.6
cod-rna	0.0	0.0	87.5	0.0	87.5	0.0	42.5	11.5
calhous	0.0	0.0	0.0	0.0	31.3	8.4	17.5	6.1

useful the MILP approach is to determine a high number of irrelevant features which may make the classifier harder to interpret and also may negatively affect accuracy due to overfitting. The sparsity for the DILSVM$^{(1)}$ is always 50% or above except for calhous, with 31.3%, while for the DILSVM$^{(2)}$ the sparsity is always above 35% except for calhous, with 17.5%. Thus, except for calhous, at least half of the features are irrelevant in the DILSVM$^{(1)}$, while this becomes at least one third in the DILSVM$^{(2)}$. The results are even more encouraging for five of these datasets, where the sparsity of the DILSVM$^{(1)}$ is at least 70%, while the sparsity of the DILSVM$^{(2)}$ is at least 50%. In addition, in the mushroom dataset, both the DILSVM$^{(1)}$ and the DILSVM$^{(2)}$ report a 100% accuracy with a 91.6% sparsity.

4.3.3. Computational performance

The model we propose, the DILSVM, is competitive against the SVM in terms of accuracy, being substantially sparser than the latter. When faced with the choice between one or two rating levels, we should observe that the DILSVM$^{(1)}$ is, in general, more appealing: it has comparable accuracies to those of the DILSVM$^{(2)}$ in seven datasets, while the DILSVM$^{(1)}$ is at least as sparse as the DILSVM$^{(2)}$. Thus, one rating level will, in general, suffice. In the remaining three datasets, including cod-rna, the DILSVM$^{(1)}$ underperforms in terms of accuracy, while the additional rating level available in the DILSVM$^{(2)}$ boosts the accuracy to a competitive level, at the cost of a lower sparsity.

Table 4. Accuracy in the validation set in % with $C/n \in \{10^{-6}, \ldots, 10^{6}\}$: the reduction strategies for the DILSVM.

Name	DILSVM$^{(1)}$				DILSVM$^{(2)}$					
	RSVM $a_1 \in \{2^0, \ldots, 2^{10}\}$		*RR* $a_1 \in \{2^0, \ldots, 2^{10}\}$		*RSVM* $a_1 \in \{2^0, \ldots, 2^{10}\}$ $a_2 \in \{\frac{a_1}{2}, \frac{a_1}{2^2}\}$		*RR* $a_1 \in \{2^0, \ldots, 2^{10}\}$ $a_2 \in \{\frac{a_1}{2}, \frac{a_1}{2^2}\}$		*fixing* $C^{(1)}, a_1^{(1)}$ $a_2 \in \{\frac{a_1^{(1)}}{2}, \frac{a_1^{(1)}}{2^2}\}$	
	mean	std	mean	std	mean	std	mean	std	mean	std
adult	75.8	0.3	82.3	0.4	77.9	2.4	89.0	2.0	89.1	1.6
mushroom	52.0	1.1	99.5	0.2	99.4	0.1	100.0	0.0	100.0	0.0
german	69.2	3.0	69.3	2.7	70.9	3.4	73.0	4.5	71.6	2.3
ijcnn1	90.2	0.4	90.2	0.4	90.5	0.5	90.3	0.3	90.2	0.4
careval	82.0	5.3	87.6	1.6	89.8	3.2	98.8	0.4	96.8	0.8
gamma	78.1	0.7	78.3	0.5	78.3	0.4	79.2	0.8	79.4	0.6
abalone	77.2	1.9	76.9	1.3	78.3	0.8	77.6	1.4	78.4	1.1
shuttle	84.3	5.1	93.8	1.8	92.2	3.8	96.3	0.8	97.3	0.1
cod-rna	71.1	3.1	76.1	1.5	73.9	5.9	80.1	4.6	81.3	2.2
calhous	78.6	2.0	78.2	1.2	80.1	0.8	80.5	0.6	82.3	0.4

The MILP approach to construct the DILSVM yields attractive results in terms of both criteria against both benchmarks. That comes with a high computational cost though, since an MILP problem needs to be solved for each parameter vector. This can be illustrated by the median ratio across the ten datasets between the computational time of the DILSVM and that of the SVM, which is equal to 210.9 for $K = 1$ and 429.6 for $K = 2$. When comparing with the fixed DILSVM$^{(1)}$, the median ratio is 361.2 for $K = 1$ and 1054.5 for $K = 2$. We should emphasize that this high computational effort only affects the phase of constructing the classifier (off-line). The evaluation phase (on-line), when new objects are to be classified, is even faster than with the SVM, since, due to the high sparsity of the classifier, fewer terms are computed. In addition, as shown in the next section, the three strategies proposed in Section 3.2 are able to reduce the time to construct the DILSVM classifier. In terms of performance, while the sparsity is not compromised, the accuracy will depend on the magnitude of the time reduction.

4.4. Results for the reduction strategies

In this section we illustrate the performance of the three reduction strategies proposed in Section 3.2, as well as their computational times. The accuracy results can be found in Table 4, and plotted in Figures 10–13. Clearly, there is a drop in accuracy with respect to the corresponding MILP approach. The results for the sparsity criterion can be found in Table 5, and plotted in Figures 14–17, showing similar performance to the MILP approach. Below we show that these strategies behave well against the benchmarks.

4.4.1. Accuracy

As before, we start with the accuracy, see Table 4. In the *RSVM* strategy, we round the feature scores of the SVM classifier using the rating levels in the grid. This is a cheap option, in which the optimization problems involved have only continuous decision variables. We now analyze the accuracy, and show that the *RSVM* strategy is dominated by the SVM. For $K = 1$, the *RSVM* strategy is clearly inefficient against the SVM, where the loss in accuracy is at least 1 p.p. in all datasets, while this becomes 48.0 p.p. in `mushroom` and 22.8 p.p. in `cod-rna`. The improvement of $K = 2$ is dramatic for `mushroom`, where the *RSVM* strategy becomes comparable to the SVM, also in `ijcnn1`, while in the remaining eight datasets this strategy is still not competitive against the SVM.

In the *RR* strategy, we reduce the computational cost of solving an MILP problem for each parameter vector. Instead, the *RR* strategy solves the LP relaxation of the DILSVM$^{(K)}$ and applies a randomized rounding to its optimal solution. Below, we show that the *RR* strategy yields accuracies close to those of the SVM. For $K = 1$, the loss in accuracy of the *RR* strategy compared to the SVM ranges from 0.5 to 17.8 p.p., and therefore, using this criterion, the *RR* strategy is dominated by the SVM. For $K = 2$, the *RR* strategy outperforms the SVM in `adult` by 4.1 p.p. and in `careval` by 3.7 p.p., for three datasets both methods are comparable, while in the remaining five the losses in accuracy (in p.p.) with respect to the SVM are 1.1 (`ijcnn1`), 1.9 (`shuttle`), 1.9 (`abalone`), 3.1 (`calhous`) and 13.8

Table 5. Sparsity in the validation set in % with $C/n \in \{10^{-6}, \ldots, 10^{6}\}$: the reduction strategies for the DILSVM.

Name	DILSVM$^{(1)}$				DILSVM$^{(2)}$					
	RSVM $a_1 \in \{2^0, \ldots, 2^{10}\}$		*RR* $a_1 \in \{2^0, \ldots, 2^{10}\}$		*RSVM* $a_1 \in \{2^0, \ldots, 2^{10}\}$ $a_2 \in \{\frac{a_1}{2}, \frac{a_1}{2^2}\}$		*RR* $a_1 \in \{2^0, \ldots, 2^{10}\}$ $a_2 \in \{\frac{a_1}{2}, \frac{a_1}{2^2}\}$		*fixing* $C^{(1)}, a_1^{(1)}$ $a_2 \in \{\frac{a_1^{(1)}}{2}, \frac{a_1^{(1)}}{2^2}\}$	
	mean	std	mean	std	mean	std	mean	std	mean	std
adult	100.0	0.0	97.4	0.9	90.3	7.0	93.3	8.4	67.1	19.1
mushroom	100.0	0.0	97.6	2.3	95.8	0.0	100.0	0.0	91.6	0.0
german	98.9	2.8	93.0	6.5	57.8	20.2	54.8	20.6	35.6	10.6
ijcnn1	98.2	3.6	96.6	4.2	65.9	7.7	70.5	4.7	100.0	0.0
careval	62.9	15.2	81.2	7.2	43.3	6.9	53.8	30.6	51.9	12.7
gamma	80.0	0.0	73.0	4.0	52.0	12.5	57.0	14.9	32.0	7.5
abalone	72.0	6.0	75.5	5.2	57.0	14.9	65.0	19.6	61.0	13.8
shuttle	75.6	9.7	68.3	7.9	43.3	20.8	43.3	27.9	47.8	15.8
cod-rna	88.8	3.8	93.8	0.0	82.5	8.3	62.5	12.5	46.3	11.3
calhous	50.0	0.0	65.0	5.7	25.0	19.4	17.5	6.1	20.0	6.1

(cod-rna). Except for cod-rna, we can conclude that the *RR* strategy with $K = 2$ and the SVM are comparable in terms of accuracy.

In the *fixing* strategy, we first construct the DILSVM$^{(1)}$, and use its classifier to reduce both the number of parameter vectors to be inspected as well as the number of binary decision variables in the MILP. For each dataset, this strategy reduces the number of MILP problems to be solved from 286 (full grid inspected) to 2 (only grid for a_2 inspected), after solving the 143 MILPs associated with the DILSVM$^{(1)}$. Roughly speaking, this halves the number of MILPs to be solved. With respect to the accuracy, the *fixing* strategy yields results close to those of the SVM. The *fixing* strategy outperforms the SVM in adult by 4.2 p.p. and in careval by 1.7 p.p., for three datasets both methods are comparable, while in the remaining five the SVM performs better. If we ignore cod-rna with a 12.6 p.p. loss, one can see that the improvement of the SVM for the remaining four datasets is below 1.4 p.p. Except for cod-rna, we can conclude that the *fixing* strategy has a comparable behavior to the SVM.

4.4.2. Sparsity

In terms of sparsity, the reduction strategies clearly dominate the SVM, as the latter is always fully dense (except for ijcnn1). We now take a closer look at the sparsity of each strategy. For $K = 1$, the sparsity of the *RSVM* strategy is at least 50% for all datasets with eight datasets above 70%, while for $K = 2$ the sparsity is above 25% for all datasets with seven above 50%. For $K = 1$, the sparsity of the *RR* strategy is at least 65% for all datasets with eight datasets above 70%, while for $K = 2$ the sparsity is above 15% for all datasets with eight datasets above 50%. Finally, the sparsity of the *fixing* strategy is at least 20% for all datasets with five datasets above 50%.

4.4.3. Computational performance

We now illustrate the reduction in computational time achieved by the strategies. As before, we measure the ratio between the computational time of a given strategy and that of the SVM, and report the median ratio across the ten datasets. For the *RSVM* strategy, the rounding time is negligible for small values of K. Thus, for $K = 1, 2$, the ratio between the computational time of the *RSVM* and the SVM is roughly equal to 1 for each dataset. For the *RR* strategy, when comparing with the SVM, the median ratio is equal to 10.3 for $K = 1$ and 34.4 for $K = 2$. Finally, the *fixing* strategy is designed to reduce the computational time when $K = 2$ using the DILSVM$^{(1)}$ classifier, and the median ratios are very similar to the ones reported for the MILP approach with $K = 1$, namely 211.6 against the SVM.

In conclusion, we have presented three strategies of very diverse nature that are able to reduce the computational time of the DILSVM classifier. When compared to the SVM, the *RR* and the *fixing* strategies are competitive in terms of accuracy, but less so is the *RSVM*, while the dominance of the three strategies in terms of sparsity is overwhelming.

5. Conclusions

In this paper we propose the DILSVM$^{(K)}$ classifier, an SVM-type classifier in which there are K possible levels of agreement of each feature with the positive class. We recommend small values of K, such as $K = 1$ and $K = 2$. In this case, our classifier enjoys two properties. First, it can be visualized as a collection of Likert scales, and therefore, since K is small, the DILSVM$^{(K)}$ classifier gains in interpretability. Second, once the classifier has been built, the evaluation of the DILSVM$^{(K)}$ (i.e., classifying new objects) is at least as inexpensive as for the SVM: classifying new objects amounts to evaluating a linear function. In addition, as shown by our numerical experiments, many coefficients are set to zero in both the DILSVM$^{(1)}$ and the DILSVM$^{(2)}$, while the SVM is fully dense.

The gain in visualization and sparsity achieved by the DILSVM is made without paying any price in accuracy. As our numerical experiments show, the DILSVM is competitive against the SVM in terms of accuracy. Our classifier is much harder to obtain than the SVM, since an MILP problem is to be solved for each parameter vector, and we have illustrated that parameter tuning is crucial if we do not want to compromise accuracy. In terms of the number of parameters, there are now one ($K = 1$) or two ($K = 2$) more parameters than in the standard SVM to tune. In order to alleviate the computational burden, three reduction strategies have been proposed. Our numerical experiments show that we are able to preserve an accuracy comparable to the SVM and significantly better sparsities. This means that, at the expense of an increase in off-line computational cost, the DILSVM is able to extract easy-to-interpret information from datasets without sacrificing classification accuracy.

We conclude with three promising extensions of our approach. First, domain knowledge can also be incorporated into the model. Given a family of features, constraints of the type "at least one feature of this family should be selected" or "no more than one feature of this family should be chosen" simply lead to new linear constraints in the MILP formulation. Second, accuracy and interpretability are usually contradicting objectives. In this paper we fix the number of rating levels, and aim at optimizing accuracy. It is natural to consider the problem of simultaneous optimization of both accuracy and interpretability, see for instance [29, 39]. This biobjective model deserves further analysis and testing. Third, our approach can also be extended to other linear classifiers, such as the classical Linear Discriminant Analysis [22] and the Logistic Regression [28], where constructing the new classifier yields Nonlinear Integer Programming problems [8]. Solving these nonlinear problems efficiently remains an important future challenge.

References

[1] C. Apte. The big (data) dig. *OR/MS Today*, 30(1):24–29, February 2003.

[2] N. Archak, A. Ghose, and P.G. Ipeirotis. Deriving the pricing power of product features by mining consumer reviews. *Management Science*, 57(8):1485–1509, 2011.

[3] B. Baesens, R. Setiono, C. Mues, and J. Vanthienen. Using neural network rule extraction and decision tables for credit-risk evaluation. *Management Science*, 49(3):312–329, 2003.

[4] D. Bertsimas, M.V. Bjarnadóttir, M.A. Kane, J.Ch. Kryder, R. Pandey, S. Vempala, and G. Wang. Algorithmic prediction of health-care costs. *Operations Research*, 56(6):1382–1392, 2008.

[5] D. Bertsimas, A. Chang, and C. Rudin. ORC: Ordered rules for classification. A discrete optimization approach to associative classification. Technical Report OR 386-11, Massachusetts Institute of Technology, 2011.

[6] C.L. Blake and C.J. Merz. UCI Repository of Machine Learning Databases. http://www.ics.uci.edu/~mlearn/MLRepository.html, 1998. University of California, Irvine, Department of Information and Computer Sciences.

[7] J.P. Brooks. Support vector machines with the ramp loss and the hard margin loss. *Operations Research*, 59(2):467–479, 2011.

[8] S. Burer and A.N. Letchford. Non-convex mixed-integer nonlinear programming: A survey. *Surveys in Operations Research and Management Science*, 17(2):97–106, 2012.

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

[10] E. Carrizosa, B. Martín-Barragán, and D. Romero Morales. Binarized support vector machines. *INFORMS Journal on Computing*, 22(1):154–167, 2010.

[11] E. Carrizosa, B. Martín-Barragán, and D. Romero Morales. A nested heuristic for parameter tuning in Support Vector Machines. *Computers and Operations Research*, 43:328–334, 2014.

[12] E. Carrizosa and D. Romero Morales. Supervised classification and mathematical optimization. *Computers and Operations Research*, 40:150–165, 2013.

[13] E. Carrizosa, A. Nogales-Gómez, and D. Romero Morales. Clustering categories in support vector machines. Technical report, 2014. `http://www.optimization-online.org/DB_HTML/2014/06/4403.html`.

[14] M. Cecchini, H. Aytug, G.J. Koehler, and P. Pathak. Detecting management fraud in public companies. *Management Science*, 56(7):1146–1160, 2010.

[15] C.C. Chang and C.J. Lin. LIBSVM: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:1–27, 2011.
[16] W. A. Chaovalitwongse, Y.-J. Fan, and R. C. Sachdeo. Novel optimization models for abnormal brain activity classification. *Operations Research*, 56(6):1450–1460, 2008.
[17] Y. Chevaleyre, F. Koriche, and J.-D. Zucker. Rounding methods for discrete linear classification. In S. Dasgupta and D. McAllester, editors, *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, volume 28, pages 651–659. JMLR Workshop and Conference Proceedings, 2013.
[18] ILOG CPLEX. www.ilog.com/products/cplex, 2012.
[19] N. Cristianini and J. Shawe-Taylor. *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*. Cambridge University Press, 2000.
[20] G. Cui, M.L. Wong, and H.-K. Lui. Machine learning for direct marketing response models: Bayesian networks with evolutionary programming. *Management Science*, 52(4):597–612, 2006.
[21] X. Fang, O.R. Liu Sheng, and P. Goes. When is the right time to refresh knowledge discovered from data? *Operations Research*, 61(1):32–44, 2013.
[22] R.A. Fisher. The use of multiple measurements in taxonomy problems. *Annals of Eugenics*, 7:179–188, 1936.
[23] A.A. Freitas. Comprehensible classification models: a position paper. *ACM SIGKDD Explorations Newsletter*, 15(1):1–10, 2014.
[24] B. F. Gage, A. D. Waterman, W. Shannon, M. Boechler, M.W. Rich, and M.J. Radford. Validation of clinical classification schemes for predicting stroke: Results from the national registry of atrial fibrillation. *Journal of the American Medical Association*, 285(22):2864–2870, 2001.
[25] M. Golea and M. Marchand. On learning perceptrons with binary weights. *Neural Computation*, 5(5):767–782, 1993.
[26] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik. Gene selection for cancer classification using support vector machines. *Machine Learning*, 46:389–422, 2002.
[27] H. Hand, H. Mannila, and P. Smyth. *Principles of Data Mining*. MIT Press, 2001.
[28] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer, New York, 2001.
[29] J.N. Hooker and H.P. Williams. Combining equity and utilitarianism in a mathematical programming model. *Management Science*, 58(9):1682–1693, 2012.
[30] R.L. Keeney and R.S. Gregory. Selecting attributes to measure the achievement of objectives. *Operations Research*, 53(1):1–11, 2005.
[31] B. Letham, C. Rudin, T.H. McCormick, and D. Madigan. Building interpretable classifiers with rules using bayesian analysis. Technical Report tr609, University of Washington, 2012.
[32] F. Li, Y. Yang, and E. Xing. From Lasso regression to feature vector machine. In Y. Weiss, B. Schölkopf, and J. Platt, editors, *Advances in Neural Information Processing Systems*, volume 18, pages 779–786. MIT Press, Cambridge, MA, 2006.
[33] R. Likert. A technique for the measurement of attitudes. *Archives of Psychology*, 22:1–55, 1932.
[34] H. Liu, F. Hussain, C. Tan, and M. Dash. Discretization: An enabling technique. *Data Mining and Knowledge Discovery*, 6(4):393–423, 2002.
[35] S. Maldonado and R. Weber. A wrapper method for feature selection using support vector machines. *Information Sciences*, 179(13):2208–2217, 2009.
[36] S. Maldonado, R. Montoya, and R. Weber. Advanced conjoint analysis using feature selection via support vector machines. *European Journal of Operational Research*, 241(2):564–574, 2015.
[37] D. Martens, B. Baesens, T.V. Gestel, and J. Vanthienen. Comprehensible credit scoring models using rule extraction from support vector machines. *European Journal of Operational Research*, 183(3):1466–1476, 2007.
[38] D. Martens and F. Provost. Explaining data-driven document classifications. *MIS Quarterly*, 38(1):73–99, 2014.
[39] C. Müssel, L. Lausser, M. Maucher, and H.A. Kestler. Multi-objective parameter selection for classifiers. *Journal of Statistical Software*, 46(5):1–27, 2012.
[40] C. Orsenigo and C. Vercellis. Multivariate classification trees based on minimum features discrete support vector machines. *IMA Journal of Management Mathematics*, 14(3):221–234, 2003.
[41] C. Orsenigo and C. Vercellis. Discrete support vector decision trees via tabu search. *Computational Statistics and Data Analysis*, 47(2):311–322, 2004.
[42] P. Raghavan and C.D. Tompson. Randomized rounding: A technique for provably good algorithms and algorithmic proofs. *Combinatorica*, 7(4):365–374, 1987.
[43] D. Romero Morales and J. Wang. A parallel discretization algorithm for cancellation rate forecasting in revenue management. Working Paper, Saïd Business School, University of Oxford, UK, 2009.
[44] D. Romero Morales and J. Wang. Forecasting cancellation rates for services booking revenue management using data mining. *European Journal of Operational Research*, 202(2):554–562, 2010.
[45] V. Roth. The generalized lasso. *IEEE Transactions on Neural Networks*, 15(1):16–28, 2004.
[46] M. Saar-Tsechansky, P. Melville, and F. Provost. Active feature-value acquisition. *Management Science*, 55(4):664–684, 2009.
[47] P.D. Turney. Cost-sensitive classification: Empirical evaluation of a hybrid genetic decision tree induction algorithm. *Journal of Artificial Intelligence Research*, 2:369–409, 1995.
[48] B. Ustun and C. Rudin. Supersparse linear integer models for optimized medical scoring systems. *arXiv preprint arXiv:1502.04269*, 2015.
[49] V. Vapnik. *The Nature of Statistical Learning Theory*. Springer-Verlag, 1995.
[50] V. Vapnik. *Statistical Learning Theory*. Wiley, 1998.
[51] P. Vlachos and M. Meyer. StatLib, 1989. http://lib.stat.cmu.edu.
[52] X. Wu, V. Kumar, J. Ross Quinlan, J. Ghosh, Q. Yang, H. Motoda, G.J. McLachlan, A. Ng, B. Liu, P.S. Yu, Z.-H. Zhou, M. Steinbach, D.J. Hand, and D. Steinberg. Top 10 algorithms in data mining. *Knowledge and Information Systems*, 14:1–37, 2007.
[53] J.-D. Zucker, Y. Chevaleyre, and D. Van Sang. Experimental analysis of new algorithms for learning ternary classifiers. In *2015 IEEE RIVF International Conference on Computing & Communication Technologies-Research, Innovation, and Vision for the Future (RIVF)*, pages

19–24. IEEE, 2015.

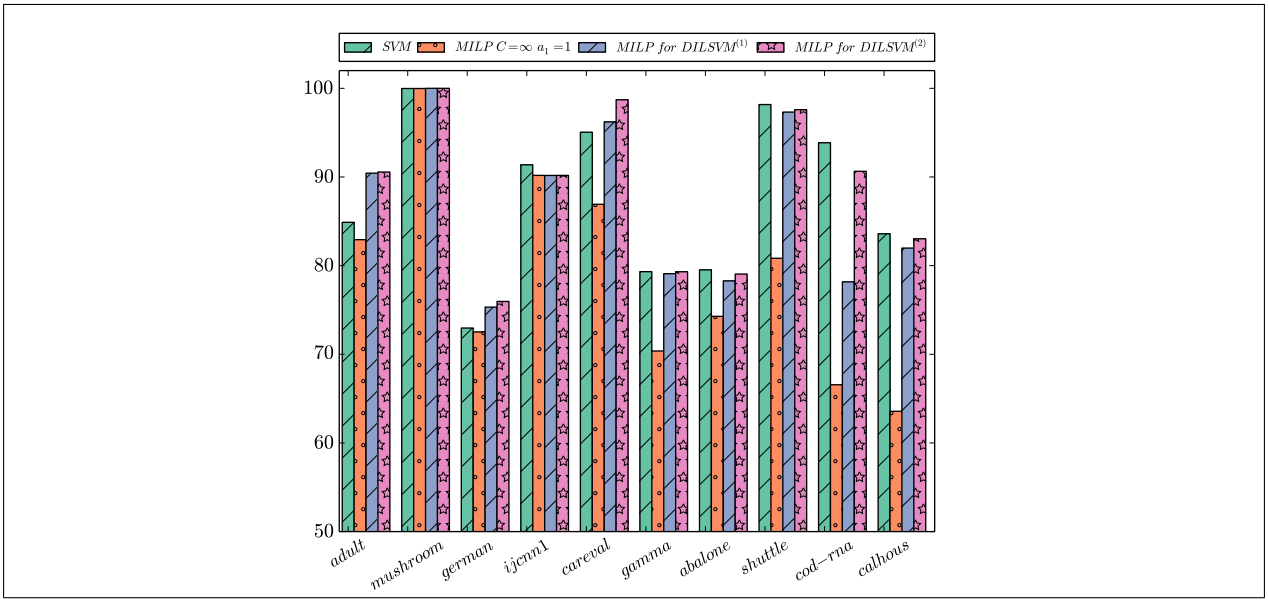

Figure 8: Accuracy of the validation set in % with $C/n \in \{10^{-6}, \ldots, 10^{6}\}$: the SVM and the MILP approach for the DILSVM$^{(1)}$ and DILSVM$^{(2)}$.

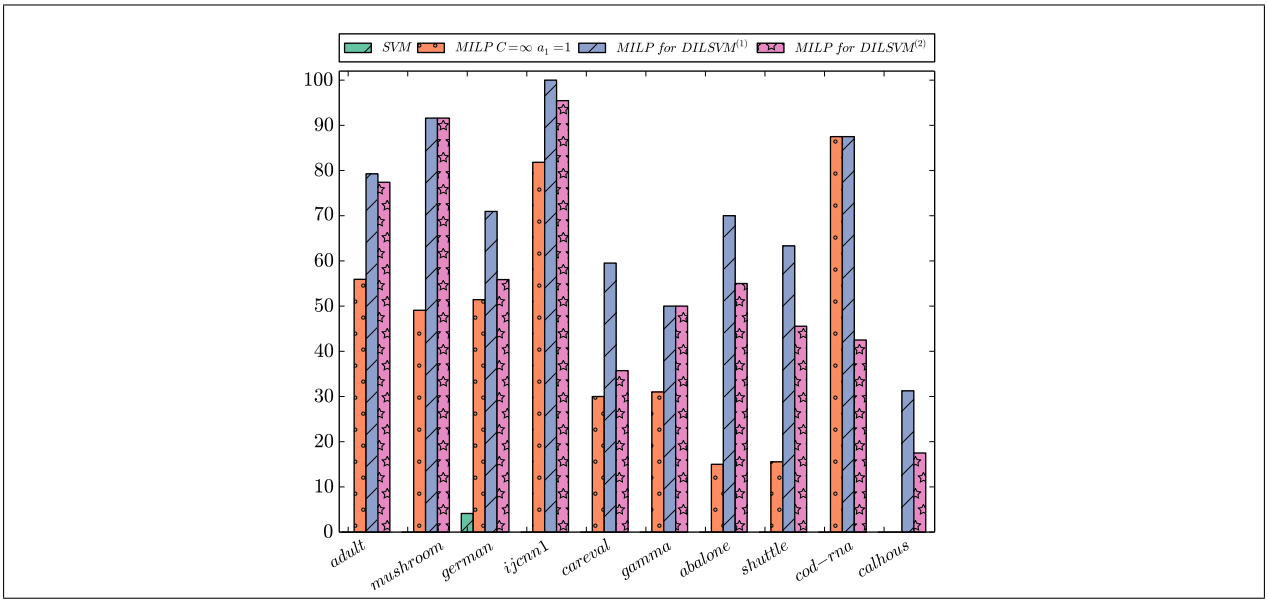

Figure 9: Sparsity (i.e., percentage of features with zero score) of the validation set in % with $C/n \in \{10^{-6}, \ldots, 10^{6}\}$: the SVM and the MILP approach for the DILSVM$^{(1)}$ and DILSVM$^{(2)}$.

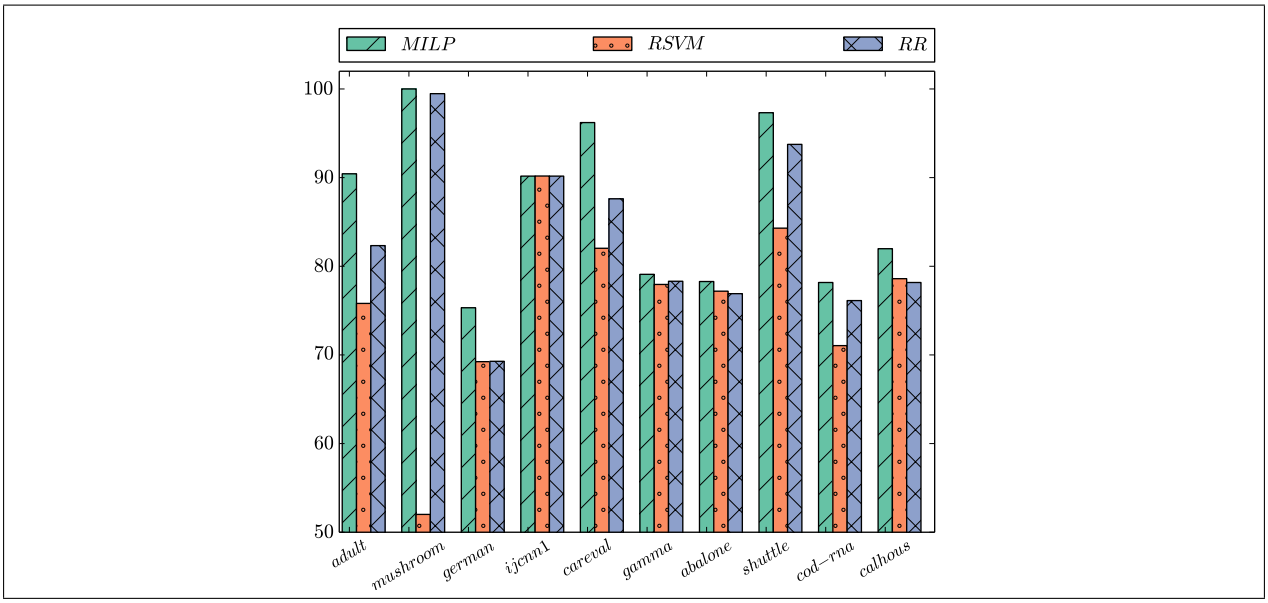

Figure 10: Accuracy of the validation set in % with $C/n \in \{10^{-6}, \ldots, 10^{6}\}$: the reduction strategies for the DILSVM$^{(1)}$ versus the MILP approach.

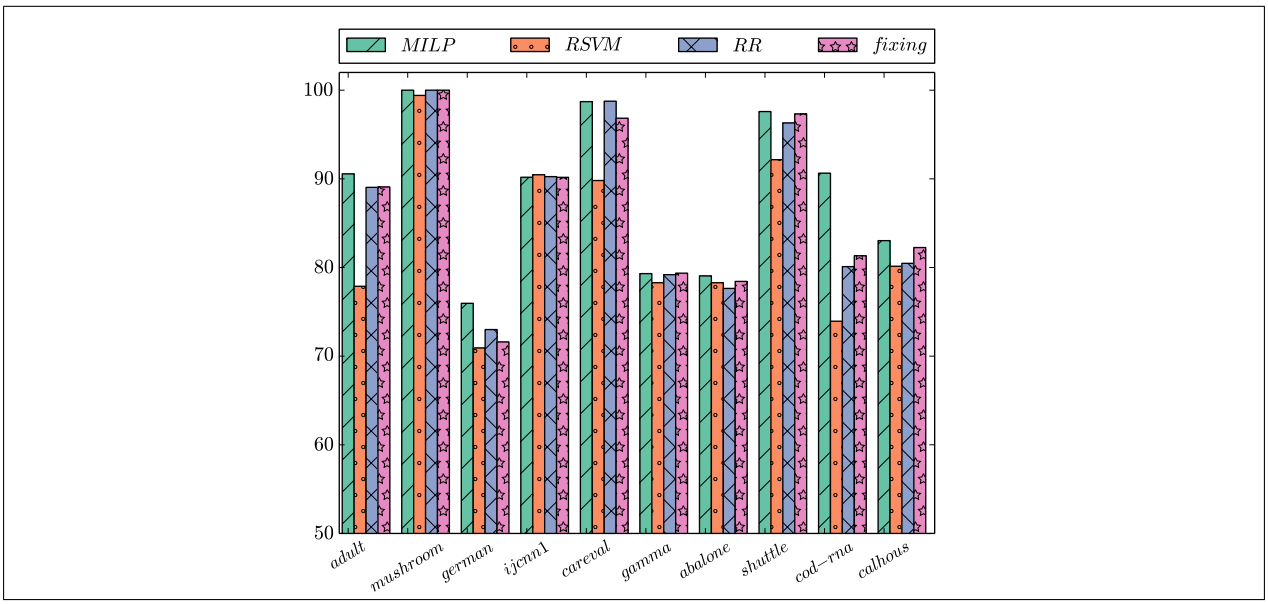

Figure 11: Accuracy of the validation set in % with $C/n \in \{10^{-6}, \ldots, 10^{6}\}$: the reduction strategies for the DILSVM$^{(2)}$ versus the MILP approach.

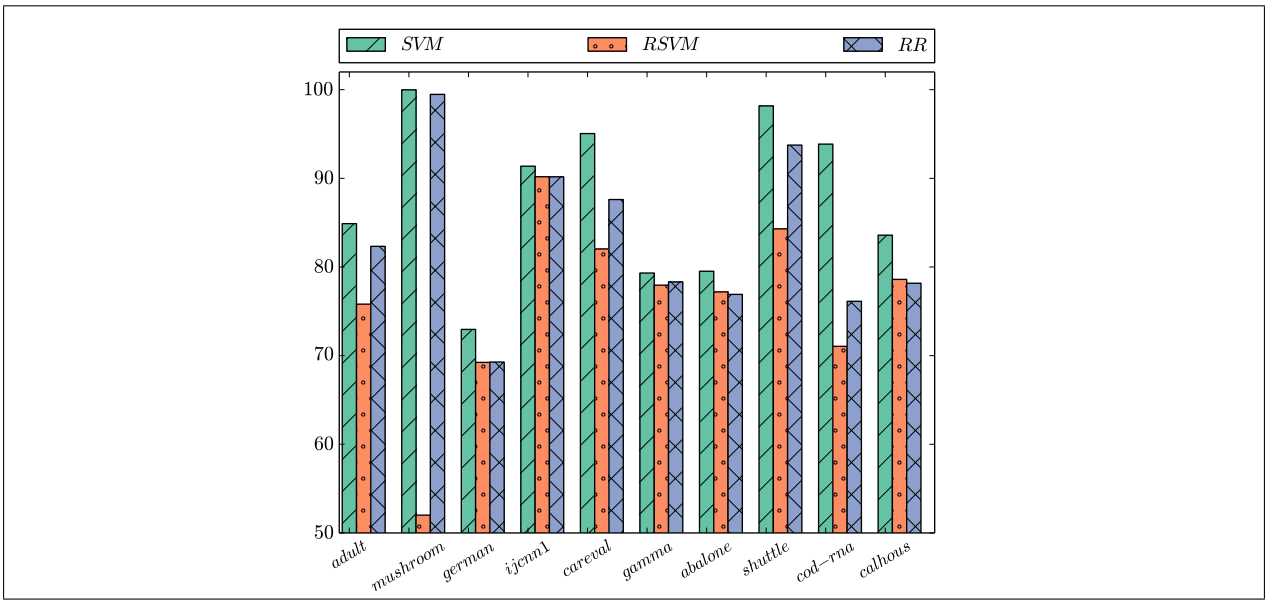

Figure 12: Accuracy of the validation set in % with $C/n \in \{10^{-6}, \ldots, 10^{6}\}$: the reduction strategies for the DILSVM$^{(1)}$ versus the SVM.

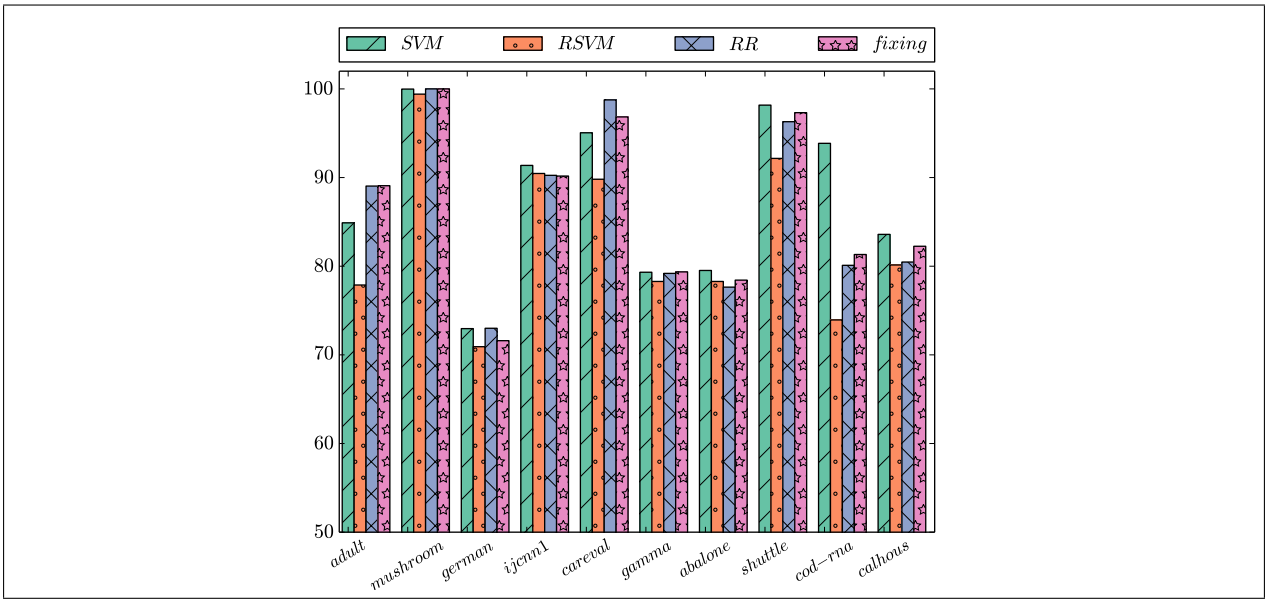

Figure 13: Accuracy of the validation set in % with $C/n \in \{10^{-6}, \ldots, 10^{6}\}$: the reduction strategies for the DILSVM$^{(2)}$ versus the SVM.

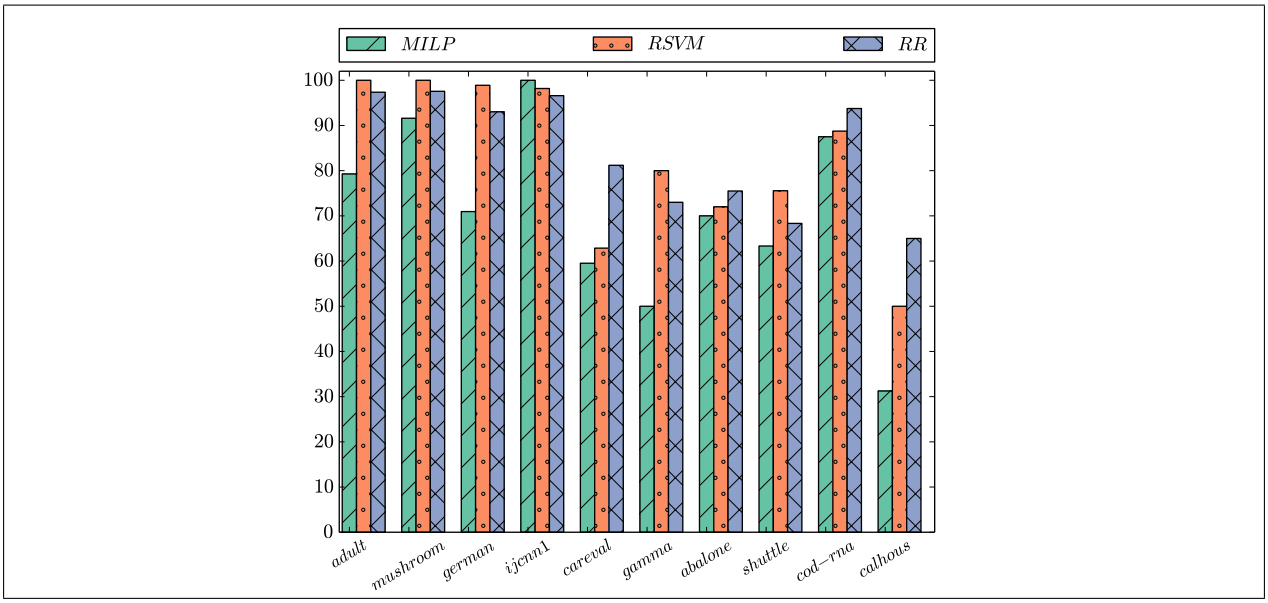

Figure 14: Sparsity (i.e., percentage of features with zero score) of the validation set in % with $C/n \in \{10^{-6}, \ldots, 10^{6}\}$: the reduction strategies for the DILSVM$^{(1)}$ versus the MILP approach.

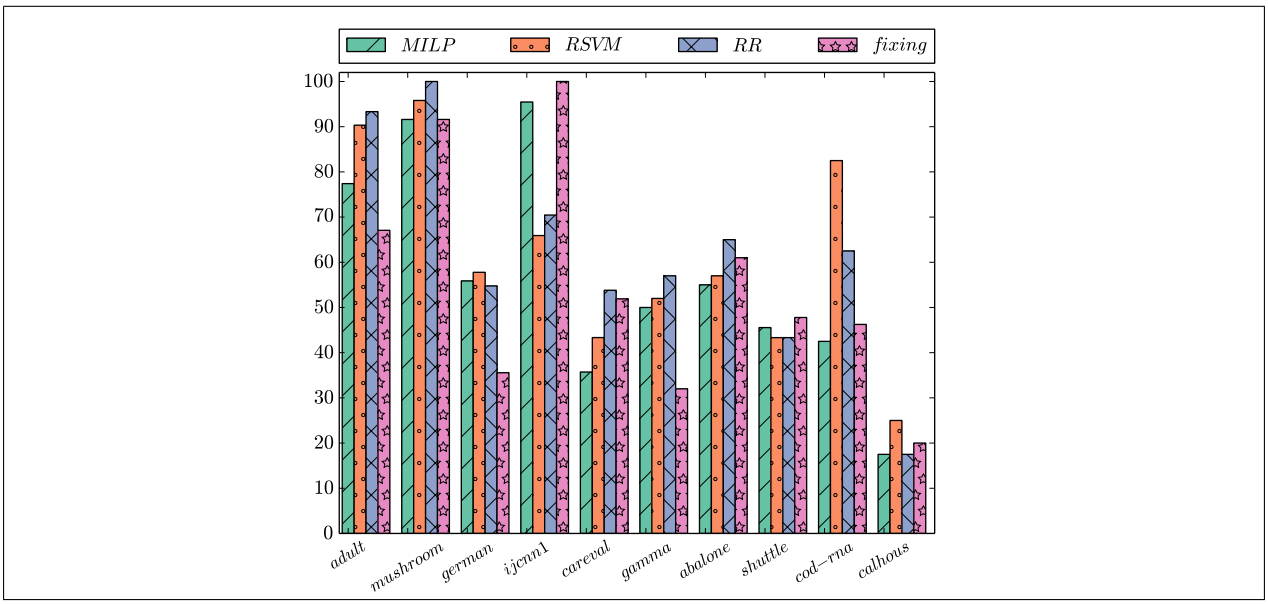

Figure 15: Sparsity (i.e., percentage of features with zero score) of the validation set in % with $C/n \in \{10^{-6}, \ldots, 10^{6}\}$: the reduction strategies for the DILSVM$^{(2)}$ versus the MILP approach.

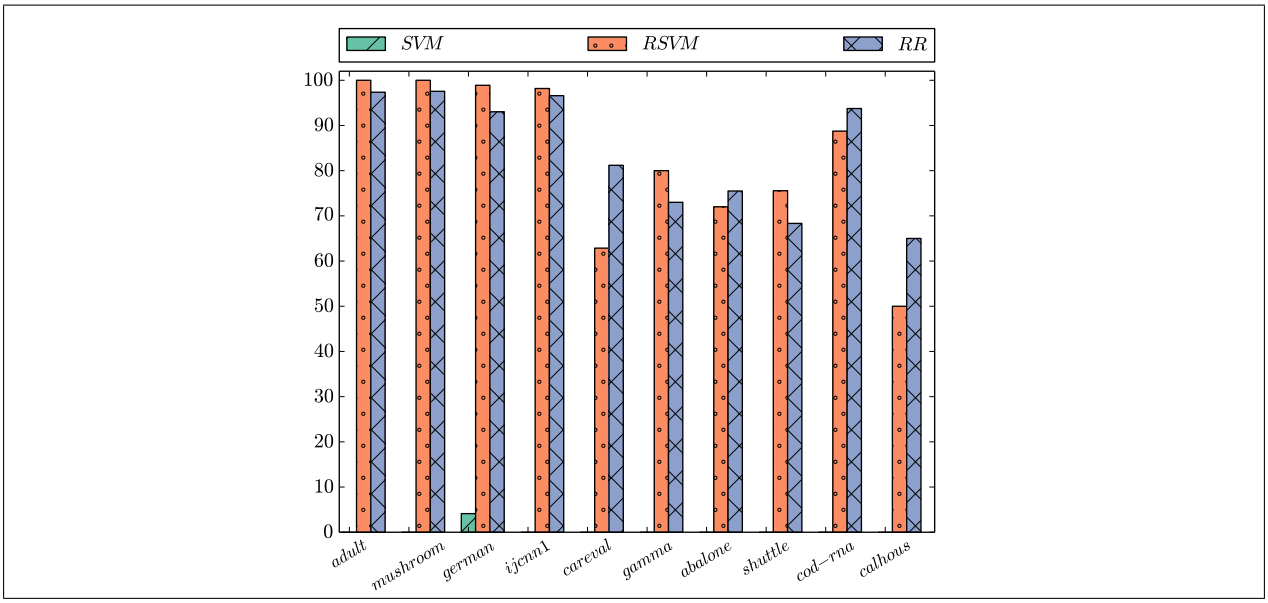

Figure 16: Sparsity (i.e., percentage of features with zero score) of the validation set in % with $C/n \in \{10^{-6}, \ldots, 10^{6}\}$: the reduction strategies for the DILSVM$^{(1)}$ versus the SVM.

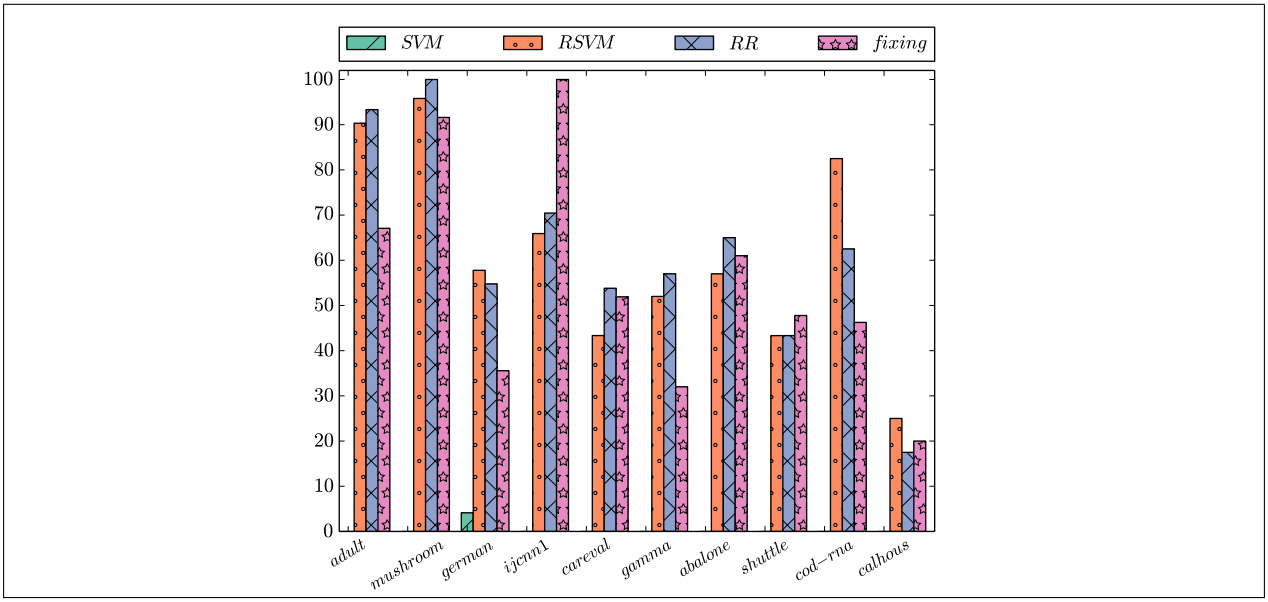

Figure 17: Sparsity (i.e., percentage of features with zero score) of the validation set in % with $C/n \in \{10^{-6}, \ldots, 10^{6}\}$: the reduction strategies for the DILSVM$^{(2)}$ versus the SVM.