

Optimal arrangements of hyperplanes for multiclass classification

Víctor Blanco[†], Alberto Japón[‡] and Justo Puerto[‡]

[†]IEMath-GR, Universidad de Granada

[‡]IMUS, Universidad de Sevilla

ABSTRACT. In this paper, we present a novel approach to construct multiclass classifiers by means of arrangements of hyperplanes. We propose different mixed integer (linear and non linear) programming formulations for the problem using extensions of widely used measures for misclassifying observations where the *kernel trick* can be adapted to be applicable. Some dimensionality reductions and variable fixing strategies are also developed for these models. An extensive battery of experiments has been run which reveal the powerfulness of our proposal as compared with other previously proposed methodologies.

1. INTRODUCTION

Support Vector Machine (SVM) is a widely-used methodology in supervised binary classification, firstly proposed by Cortes and Vapnik [12]. Given a number of observations with their corresponding labels, the SVM technique consists of finding a strip in the feature space so that each class is included in a different semispace maximizing the separation between classes (in a training sample) and minimizing some measure of the misclassification errors. This problem can be cast within the class of convex optimization and its dual enjoys very good properties. Actually, one can project the original data out onto a higher dimensional space where the separation of the classes can be more adequately performed, and still keeping the same computational effort that was required in the original problem. This fact is the so-called *kernel trick*, and very likely this is one of the reasons that has motivated the successful use of this tool in a wide range of applications [4, 19, 23, 30, 38].

Most of the SVM literature concentrates on binary classification where several extensions are available. One can use different measures for the separation between classes [9, 21, 22], select important features [29], apply regularization strategies [28], etc. However, the analysis of SVM-based methods for datasets with more than two classes has been, from our point of view, only partially investigated. The k -label ($k > 2$) SVM consists of the following. Given a *training sample* of observations $\{x_1, \dots, x_n\} \subseteq \mathbb{R}^p$ with their labels $(y_1, \dots, y_n) \in \{1, \dots, k\}^n$, the goal is to construct a decision rule able to classify out-of-sample observations learning from the training sample.

The most common techniques applied to supervised multiclass classification are based on natural extensions of the tools valid for the binary case: Deep Learning [1], k -Nearest Neighborhoods [13, 39] or Naïve Bayes [26], among others.

In addition, one can also find some techniques for multiclass classification that take advantage of the SVM methods for binary classification. The most popular multiclass SVM-based approaches are One-Versus-All (OVA) and One-Versus-One (OVO). The former, namely OVA, computes, for each class $r \in \{1, \dots, k\}$, a binary SVM classifier labeling the observations as 1, if the observation is in the class r and -1 otherwise. The process is repeated for all classes (k times), and then each observation is classified into the class whose constructed hyperplane is the furthest from it in the positive halfspace. In the OVO approach, classes are separated with $\binom{k}{2}$ hyperplanes using one hyperplane for each pair of classes, where the decision rule comes from a voting strategy in which the most represented class among votes becomes the class predicted. OVA and OVO inherit most of the good properties of binary SVM. In spite of that, they are not able to correctly classify datasets where separated clouds of observations may belong to the same class (and thus are given the same label) when a linear kernel is used. Another popular method is the directed acyclic graph SVM (DAGSVM) [37]. In this technique, although the decision rule involves the same hyperplanes built with the OVO approach, it is not given by a unique voting strategy but for a sequential number of voting in which the most unlikely class is removed until only one class remains. In addition, apart from OVA and OVO, there are other methods based on decomposing the multiclass problem into several binary classification problems. In particular, in [2, 15], this decomposition is based on the construction of a coding matrix that determines the pairs of classes that will be used to build the separating hyperplanes. Alternatively, other methods such as Cramer-Singer (CS) [14], Weston-Watkins (WW) [41] or Lee-Lin-Wahba (LLW) [25], do not address the classification problem sequentially but as a whole considering all the classes within the same optimization model. Obviously, this seems to be the correct approach. In particular, in WW, k hyperplanes are used to separate the k classes, each hyperplane separating one class from the others, using $k - 1$ misclassification errors for each observation. The same separating idea, is applied in CS but reducing the number of misclassification errors for each observation to a unique value. In LLW, a *sum-to-zero constraint* is used to reduce the dimensionality of the problem. We can also find a quadratic extension based on LLW proposed in [18]. Finally, in [11], the authors propose a multiclass SVM-based approach, *GenSVM*, in which the classification boundaries for a problem with k classes are obtained in a $(k - 1)$ -dimensional space using a simplex encoding. Some of these methods have become popular and are implemented in most software packages in machine learning as `e1071` [34], `scikit-learn` [36] or `MSVMpack` [24]. Nevertheless, as far as we are concerned, none of the existing multiclass SVM methods keeps the essence of binary SVM which stems from finding a globally optimal partition of the feature space.

This paper proposes a novel approach to handle multiclass classification extending the paradigm of binary SVM classifiers. In particular, our method finds a polyhedral partition of the feature space and an assignment of classes to the *cells* of the partition, by maximizing the separation between classes and minimizing two intuitive misclassification errors. Obviously, as in standard SVM, we can also account in different ways the misclassification errors (hinge or ramp-based losses). For bi-class instances, and using a single separating hyperplane, the method coincides with the standard SVM. Nevertheless, even for 2-classes

datasets, new alternatives appear if more than one hyperplane to separate the data is permitted. In particular, our approach allows one to generalize the polyhedral conic classifiers presented in [3].

Apart from justifying the rationale of our method, we also propose different mathematical programming formulations in order to solve the resulting optimization problems. These formulations belong to the family of Mixed Integer (Linear and Non Linear) Programming (MILP and MINLP) problems, in which the nonlinearities come from the representation of the Euclidean distance margin between classes, that can be modeled as a set of second order cone constraints [7]. This type of constraints can be handled nowadays by any of the most popular off-the-shelf optimization solvers (CPLEX, Gurobi, XPress, SCIP, ...).

These models also have a combinatorial nature induced by the correct allocation of labels to cells. Therefore, they require using some binary variables. This approach is not new and recently, a few attempts have been proposed for different classification problems using discrete optimization tools. For instance, in [40] the authors construct classification hyperboxes for multiclass classification, in [6] the authors provide formulations for SVM with unlabelled data (semi-supervised SVM), and in [17, 29, 31] mixed integer linear programming tools are provided for feature selection in SVM. Handling a large number of binary variables in the models may become an inconvenient when trying to compute classifiers for medium to large size instances. This inconvenience is alleviated with some preprocessing and dimensionality reduction techniques that are also introduced.

In case the data are, by nature, nonlinearly separable, in classical SVM one can apply the so-called kernel trick to project the data out onto a higher dimensional space where the linear separation has a better performance. The key point is that one does not need to know neither the dimension of the final space nor the specific transformation that is applied to the data: the resulting mathematical programming problem is in the same space as the original one. Here, we show that the kernel trick can be extended to our framework and therefore, it also allows us to find nonlinear classifiers with this methodology.

To assess the validity of our method we have performed a battery of computational tests on two different families of data. We have tested our method against some well-known multiclass SVM classifiers (OVO, CS, WW and LLW) on 6 databases from the UCI repository. Moreover, we also report results on synthetic datasets specially tailored to capture the difficulty of multiclass supervised classification. In all cases, our methods give results similar or superior to those provided for the other methods. In particular, for the synthetic data instances the improvement in accuracy on the test samples are remarkable (see Table 3).

The rest of the paper is organized as follows. In sections 2 and 3 we describe and set up the elements of the problem to be considered. Afterward, we introduce a MINLP formulation for our model. Alternatively, we also present a linear version, which is obtained whenever we measure the margins with the ℓ_1 -norm. A discussion on the extension, with very few modifications, of the previous models to the Ramp Loss versions is included as well. In Subsection 3.2 we show how an analogous to the kernel trick can be extended to be applied in this model. Section 4 describes some heuristic strategies, preprocessing and dimensionality

reductions to obtain good quality initial solutions of the MINLP. Finally, in section 5 we report our computational results on different real and synthetic datasets, and compare our method with the most classical ones for multiclass SVM.

2. MULTICLASS SUPPORT VECTOR MACHINES

In this section, we introduce the problem under study and set the notation used through this paper.

Given a training sample $\{(x_1, y_1), \dots, (x_n, y_n)\} \subseteq \mathbb{R}^p \times \{1, \dots, k\}$ the goal of supervised classification is to find a decision rule to assign labels (y) to data (x), in order to be applied to out-of-sample data. We assume that a given number, m , of hyperplanes in \mathbb{R}^p have to be built to obtain a subdivision of this space into full dimension polyhedral regions that we shall denote as *cells*. (Here, we would like to mention that the term *cell* stands for a nonempty intersection of the semispaces induced by the hyperplanes in the considered family). Let us denote by $\mathcal{H}_1, \dots, \mathcal{H}_m$ the hyperplanes to be found, which are in the form $\mathcal{H}_r = \{z \in \mathbb{R}^p : \omega_r^t z + \omega_{r0} = 0\}$ for $r = 1, \dots, m$ (here v^t stands for the transpose operator applied to the vector $v \in \mathbb{R}^p$). Each cell induced with such an arrangement of hyperplanes will be then assigned to a label in $\{1, \dots, k\}$. In Figure 1 we illustrate a subdivision of \mathbb{R}^2 induced by 2 hyperplanes and the labels assigned to each cell. In the left figure, we represent the observations, highlighting the classes with different symbols (stars, circles and squares). In the right figure, two hyperplanes which induce 4 cells are constructed to separate the three classes. Each cell is assigned to a class (north \rightarrow circles, south \rightarrow stars, east \rightarrow stars and west \rightarrow squares). In this example the subdivision in cells and the assignment of labels reaches a perfect classification on the given observations.

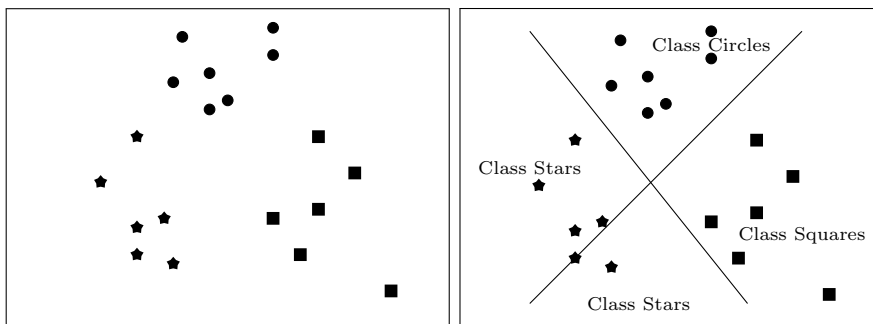


FIGURE 1. Illustration of a subdivision induced by 2 hyperplanes in \mathbb{R}^2 .

From the above, we would like to construct an arrangement of m hyperplanes, $\mathbb{H} = \{\mathcal{H}_1, \dots, \mathcal{H}_m\}$, determined by $\omega_1, \dots, \omega_m \in \mathbb{R}^{p+1}$ (the first component of each vector accounts for the intercept) and a decision rule that assigns a single label to each one of the cells in the subdivision of the space induced by such an arrangement. We would like to point out that each cell in the subdivision can be univocally identified with a $\{-1, +1\}$ -vector in \mathbb{R}^m : the ℓ -component of that

vector represents the side (positive or negative) with respect to the hyperplane \mathcal{H}_ℓ where that cell lies in.

Definition 2.1 (Suitable Assignment). *Given a subdivision \mathcal{C} of \mathbb{R}^p into cells induced by the arrangement of hyperplanes $\mathbb{H} = \{\mathcal{H}_1, \dots, \mathcal{H}_m\}$ in \mathbb{R}^p , a function $g : \{-1, 1\}^m \rightarrow \{1, \dots, k\}$ is said a suitable assignment, if g univocally maps cells (equivalently, sign-patterns) to labels in $\{1, \dots, k\}$.*

Observe that a suitable assignment, g , allows us to classify any observation $x \in \mathbb{R}^p$ within the set of classes $\{1, \dots, k\}$, as follows:

- (1) Identify x with a sign-pattern: $s(x) = (s_1(x), \dots, s_m(x)) \in \{-1, +1\}^m$, where $s_r(x) = \text{sign}(\omega_r^t x + \omega_{r0})$ for $r = 1, \dots, m$.
- (2) Apply the function g to the sign-patterns: $\hat{y}(x) = g(s(x)) \in \{1, \dots, k\}$, is the *predicted* label of x .

The quality of the decision rule is based, on comparing predictions and actual labels on a training sample, but also on maximally separating the classes in order to find good predictions and avoid undesired overfitting.

In binary classification datasets, SVM is a particular case of our approach if $m = 1$, i.e., a single hyperplane to subdivide the feature space is used. In such a case, signs are in $\{-1, 1\}$ and classes in $\{1, 2\}$, so whenever there are observations in both classes, the assignment is one-to-one. However, even for biclass instances, if more than one hyperplane is used, one may find better classifiers (we illustrate this behavior with the dataset 2C4N of our computational experiments in Table 3). In Figure 2, left-and-right, we draw the same dataset of labeled (red and blue) observations and the result of applying a standard SVM (left) and our method with 2 hyperplanes. In that picture one may see that not only the misclassification errors are smaller with two hyperplanes, as expected, but also the separation between classes is larger, improving the predictive power of the classifier.

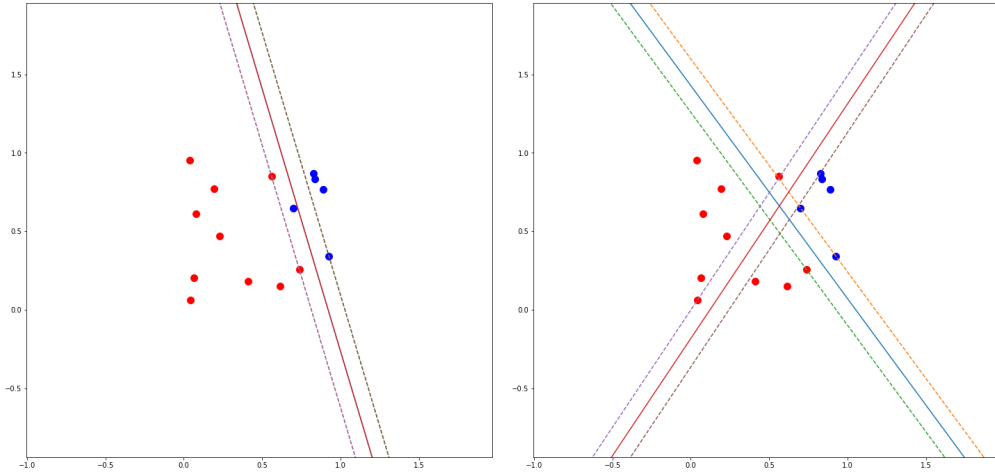


FIGURE 2. Standard SVM (left) and our approach with 2 hyperplanes (right).

The rationale of our approach is particularly adequate for datasets in which there are several separated “clouds” of observations that belong to the same

class. In Figure 3, we show two different instances in which, again, the colors indicate the class of the observations. The classes in both instances cannot be appropriately separated using any of the available linear SVM-based methods in the literature since they are based on subdividing the space on class-connected regions. However, we are able to perfectly separate the classes using 5 hyperplanes.

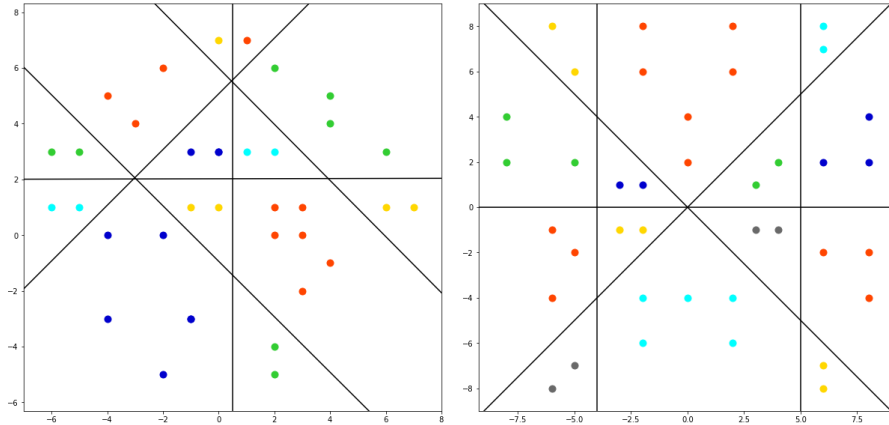


FIGURE 3. A 5-classes instance classified with our approach using 5 hyperplanes (left) and a 6-classes instance classified with our approach using 5 hyperplanes (right).

In Figure 4 we compare our approach and the One-versus-One (OVO) approach in an instance with 24 observations. In the left figure we show the result of separating the classes with four hyperplanes, reaching a perfect classification on the training sample. In the right figure we show the best linear OVO classifier, in which only 66% of the data were correctly classified. We would like also to highlight that, although nonlinear SVM-approaches may separate the data more conveniently, our approach may help to avoid using kernels and ease the interpretation of the results.

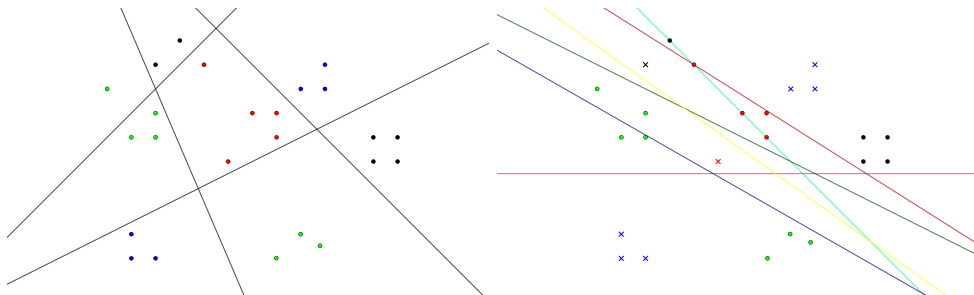


FIGURE 4. A 4-classes instance classified with our approach using 4 hyperplanes (left) and the same instance classified using the OVO SVM approach (right).

Different alternatives could be admissible to justify the rationale of the multiclass classifiers in our framework. To simplify the presentation, we will concentrate on two different models which share the same paradigm but differ in the way they account for misclassification errors. Recall that in SVM-based methods, two criteria are *simultaneously* optimized when constructing a classifier. On the one hand, a measure of the quality of the decision rule on out-of-sample observations, based on finding a maximum separation between classes; and on the other hand a measure of the misclassification errors for the training set of observations. Both criteria are adequately weighted in order to find a good compromise between the two goals.

In what follows we describe how similar measures can be defined in our multiclass classification framework and the way we account them for.

2.1. Separation between classes . Separation between classes will be measured as it is usual in SVM-based methods. Let $(\omega_1; \omega_{10}), \dots, (\omega_m; \omega_{m0}) \in \mathbb{R}^p \times \mathbb{R}$ be the coefficients and intercepts of a set of hyperplanes. The distance induced by a norm $\|\cdot\|$ between the shifted hyperplanes $\mathcal{H}_r^+ = \{z \in \mathbb{R}^p : \omega_r^t z + \omega_{r0} = 1\}$ and $\mathcal{H}_r^- = \{z \in \mathbb{R}^p : \omega_r^t z + \omega_{r0} = -1\}$ is given by $\frac{2}{\|\omega_r\|^*}$, where $\|\cdot\|$ is a given norm in \mathbb{R}^p and $\|\cdot\|^*$ is its dual norm (see [32]). Unless explicitly mentioned, we will consider that $\|\cdot\|$ is the Euclidean norm which dual is also the Euclidean norm.

Hence, in order to find globally optimal hyperplanes with maximum separation, we maximize the minimum separation between classes, that is $\min \left\{ \frac{2}{\|\omega_1\|}, \dots, \frac{2}{\|\omega_m\|} \right\}$. This measure will conveniently keep the minimum separation between classes as largest as possible. Observe that finding the maximum min-separation is equivalent to minimize $\max \left\{ \frac{1}{2} \|\omega_1\|^2, \dots, \frac{1}{2} \|\omega_m\|^2 \right\}$. For a given arrangement of hyperplanes, $\mathbb{H} = \{\mathcal{H}_1, \dots, \mathcal{H}_m\}$, we will denote by $h_H(\mathcal{H}_1, \dots, \mathcal{H}_m) = \max \left\{ \frac{1}{2} \|\omega_1\|^2, \dots, \frac{1}{2} \|\omega_m\|^2 \right\}$.

We note in passing that different criteria could have been used to model the separation between classes. For instance, one may consider to maximize the summation of all separations namely $\sum_{r=1}^m \frac{2}{\|\omega_r\|}$. However, although mathematically possible, this approach does not capture the original concept in classical SVM and we have left it to be developed by the interested reader.

2.2. Misclassification errors. The performance of a classifier on the training set is usually measured with some function of the misclassification errors. Classical SVMs with hinge-loss errors use, for non well-classified observations, a penalty proportional to the distance to the side in which they would have been well-classified. Then the overall sum of these errors is minimized. We extend the notion of hinge-loss errors to the multiclass setting as follows.

Let $\mathbb{H} = \{\mathcal{H}_1, \dots, \mathcal{H}_m\}$ be an arrangement of hyperplanes and (x, y) a pair observation (x) , label (y) , with $s(x) = (s_1(x), \dots, s_m(x))$ being the sign-pattern of x with respect to the hyperplanes in \mathbb{H} . Let $g : \{-1, 1\}^m \rightarrow \{1, \dots, k\}$ be a suitable assignment. We denote by $t(x) = (t_1(x), \dots, t_m(x))$ the signs of the closest cell to x whose class by g is y . We will say that (x, y) is *wrong-classified* with respect to \mathcal{H}_r if $s_r(x) \neq t_r(x)$, otherwise it is said that (x, y) is *well-classified*.

In what follows we describe the different error measures (misclassification errors due to different causes) that will be considered for x in order to construct an optimal decision rule.

Definition 2.2 (Multiclass In-Margin Hinge-Loss). *The multiclass in-margin hinge-loss for (x, y) with respect to the hyperplane \mathcal{H}_r is given as:*

$$h_I(x, y, \mathcal{H}_r) = \begin{cases} \max\{0, 1 - s_r(x) \cdot (\omega_r^t x + \omega_{r0})\} & \text{if } x \text{ is well classified through } \mathcal{H}_r, \\ 0 & \text{otherwise.} \end{cases}$$

Observe that h_I models the error due to observations that although adequately classified with respect to \mathcal{H}_r , belong to the margin between the shifted hyperplanes \mathcal{H}_r^+ and \mathcal{H}_r^- . These errors will be zero if the observation is wrong-classified, or if it is well-classified and does not belong to the margin induced by the r -th hyperplane.

Definition 2.3 (Multiclass Out-Margin Hinge-Loss). *The multiclass out-margin hinge-loss for (\bar{x}, \bar{y}) with respect to the hyperplane \mathcal{H}_r is given as:*

$$h_O(x, y, \mathcal{H}_r) = \begin{cases} 1 - t_r(x) \cdot (\omega_r^t x + \omega_{r0}) & \text{if } x \text{ is not well classified through } \mathcal{H}_r, \\ 0 & \text{otherwise.} \end{cases}$$

h_O measures, for wrong-classified observations, how far is from being well-classified. This error is zero whenever an observation is well-classified. Note that if an observation, besides being wrong-classified, belongs to the margin between \mathcal{H}_r^+ and \mathcal{H}_r^- , then only h_O should be accounted for. In Figure 5 we illustrate the differences between the two types of losses.

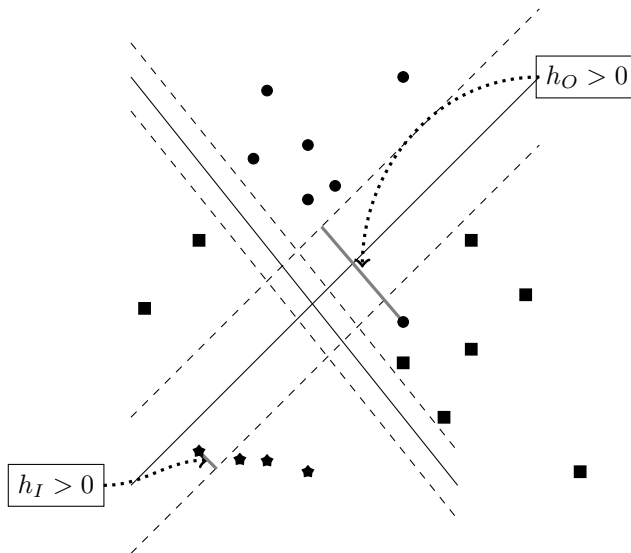


FIGURE 5. Illustration of the error measures considered in our approach.

3. MIXED INTEGER NON LINEAR PROGRAMMING FORMULATIONS

In this section we describe the two mathematical optimization models that we propose for the multiclass classification problem. Using the notation introduced

in previous sections, the problem can be mathematically stated as follows:

$$\min h_H(\mathcal{H}_1, \dots, \mathcal{H}_m) + C_1 \sum_{i=1}^n \sum_{r=1}^m h_I(x_i, y_i, \mathcal{H}_r) + C_2 \sum_{i=1}^n \sum_{r=1}^m h_O(x_i, y_i, \mathcal{H}_r) \quad (1)$$

s.t. \mathcal{H}_r is a hyperplane in \mathbb{R}^p , for $r = 1, \dots, m$.

C_1 and C_2 are parameters which model the *cost* of misclassified and strip-related errors. Usually these constants will be considered equal, nevertheless, in practice analyzing different values for them might lead to better results on predictions. A case of interest results considering $C_2 = mC_1$, i.e., the unitary cost of misclassification errors caused by out-margin observations is m times the unitary cost caused by in-margin observations, giving a larger penalty to wrongly classified observations, avoiding the calibration of a larger number of parameters.

Observe that the problem above consists of finding the arrangement of hyperplanes minimizing a combination of the three quality measures described in the previous section: 1) the maximum margin between classes, 2) the overall sums of the in-margin errors and 3) the out-margin misclassification errors. In what follows, we describe how the above problem can be re-written as a mixed integer non linear programming problem by means of adequate decision variables and constraints. Furthermore, the proposed model will consist of a set of continuous and binary variables, a linear objective function, and a set of linear and second order cone constraints. It will allow us to push the model to a commercial solver in order to easily solve, at least, small to medium instances.

First, we describe the variables and constraints needed to model the first term in the objective function. We consider the continuous variables $\omega_r \in \mathbb{R}^p$ and $\omega_{r0} \in \mathbb{R}$ to represent the coefficients and intercept of hyperplane \mathcal{H}_r , for $r = 1, \dots, m$. Since there is no distinction between hyperplanes, we can assume, without loss of generality that they are non-decreasingly sorted with respect to the norms of their coefficients, i.e., $\|\omega_1\| \geq \|\omega_2\| \geq \dots \geq \|\omega_m\|$. Then, it is straightforward to see that the term $h_H(\mathcal{H}_1, \dots, \mathcal{H}_m)$ can be replaced in the objective function by $\frac{1}{2}\|\omega_1\|^2$, once the following set of constraints is included in the model:

$$\frac{1}{2}\|\omega_{r-1}\|^2 \geq \frac{1}{2}\|\omega_r\|^2, \forall r = 2, \dots, m. \quad (2)$$

As already applied in multivariate linear regression [8] or binary SVM [9], other norms can also be used to measure the margin.

For the second term, the in-margin misclassification error, $h_I(x_i, y_i, \mathcal{H}_r)$, corresponding to the observation (x_i, y_i) will be identified with the continuous variable $e_{ir} \geq 0$, for $i = 1, \dots, n$, $r = 1, \dots, m$. Observe that to properly determine each of these errors, one has to determine whether the observation x_i is well-classified or not with respect to the r th hyperplane. In order to do that we need to introduce some binary variables. First, we consider the following two sets of binary variables:

$$t_{ir} = \begin{cases} 1 & \text{if } \omega_r^t x_i + \omega_{r0} \geq 0, \\ 0 & \text{otherwise.} \end{cases} \quad \text{and} \quad z_{is} = \begin{cases} 1 & \text{if } i \text{ is assigned to class } s, \\ 0 & \text{otherwise.} \end{cases}$$

for $i = 1, \dots, n$, $r = 1, \dots, m$, $s = 1, \dots, k$. The t -variables model the sign-pattern of the observations, while the z -variables give the allocation profile of observations to classes. As mentioned above, the classification rule is based on assigning sign-patterns to classes.

The adequate definition of the t -variables is assured with the following constraints:

$$\omega_r^t x_i + w_{r_0} \geq -T(1 - t_{ir}), \quad \forall i \in N, r \in M \quad (3)$$

$$\omega_r^t x_i + w_{r_0} \leq T t_{ir} \quad \forall i \in N, r \in M \quad (4)$$

where T is a big enough constant. Observe that T can be accurately estimated based on the data set under consideration.

The following constraints assure the adequate relationships between the variables:

$$\sum_{s=1}^k z_{is} = 1, \quad \forall i \in N, \quad (5)$$

$$\|z_i - z_j\|_1 \leq 2\|t_i - t_j\|_1, \quad \forall i, j \in N, \quad (6)$$

Observe that (17) enforce that a single class is assigned to each observation while (18) assure that the assignments of two observations must coincide if their sign-patterns are the same. Additionally, the set of z -variables determines whether an observation is well-classified. Indeed, let $\delta_i \in \{0, 1\}^k$ be defined as $\delta_{is} = 1$ if $y_i = s$ and 0 otherwise. (Observe that δ_i is the binary encoding of the class of the i th observation.) Then, $\xi_i = \frac{1}{2}\|z_i - \delta_i\|_1 \in \{0, 1\}$ assumes the value zero if and only if the observation i is well-classified, i.e.,

$$\xi_i = \begin{cases} 1 & \text{if } i \text{ is well-classified,} \\ 0 & \text{otherwise.} \end{cases}$$

Now, we will model whether the i th observation is well-classified or not, with respect to the r th hyperplane. Observe that the measure of how far is a wrong-classified observation from being well-classified, needs a further analysis. One may has a wrong-classified observation and several training observations in its same class. We assume that the error for this observation is the misclassification error with respect to the closest cell for which there are well-classified observations in its class. Thus, we need to model the decision on the well-classified *representative* observation for a wrong-classified observation. In Figure 6, we illustrate this type of misclassification errors. The observation x_i is wrong-classified but the misclassification error of x_i , in case x_j is chosen as its representative (well-classified) observation, is 0 with respect to hyperplane \mathcal{H}_1 (note that both x_i and x_j are in the same side of \mathcal{H}_1), whereas the misclassification error with respect to \mathcal{H}_2 is h . Observe h is the distance between x_i and the shifted hyperplane defining the halfspace where x_j lies in. We consider the following set of binary variables:

$$h_{ij} = \begin{cases} 1 & \text{if } x_j, \text{ which is well classified and verifies } y_j = y_i, \text{ is the representative} \\ & \text{of } x_i \text{ in its closest cell through hyperplanes,} \\ 0 & \text{otherwise} \end{cases}$$

These variables require to impose the following constraints:

$$\sum_{\substack{j \in N: \\ y_i = y_j}} h_{ij} = 1, \quad \forall i \in N, \quad (7)$$

$$\xi_j + h_{ij} \leq 1 \quad \forall i, j \in N(y_i = y_j), \quad (8)$$

$$h_{ii} = 1 - \xi_i \quad \forall i \in N, \quad (9)$$

The first set of constraints, (20), impose a single assignment between observations belonging to the same class. Constraints (21) avoid choosing wrong-classified representative observations. The set of constraints (22) enforces well-classified observations to be represented by themselves.

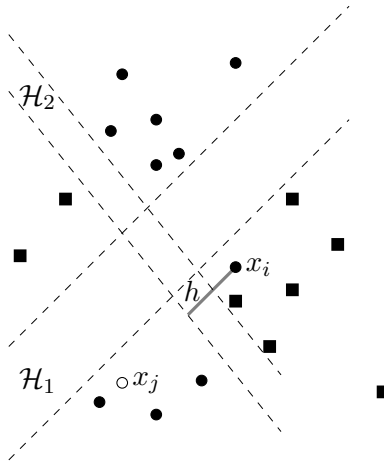


FIGURE 6. Illustration of the wrong-classification errors.

With these variables, we can model the in-margin errors by means of the following constraints:

$$\omega_r^t x_i + \omega_{r0} \geq 1 - e_{ir} - T(3 - t_{ir} - t_{jr} - h_{ij}), \quad \forall r \in M, \quad (10)$$

$$\omega_r^t x_i + \omega_{r0} \leq -1 + e_{ir} + T(1 + t_{ir} + t_{jr} - h_{ij}), \quad \forall r \in M, \quad (11)$$

These constraints model, by using the sign-patterns given by t , that, $e_{ir} = \max\{0, \min\{1, 1 - s_r(x)(\omega_r^t x_i + \omega_{r0})\}\}$. Note that the constraints are active if either $t_{ir} = t_{jr} = h_{ij} = 1$, i.e., if the well-classified observation x_j is the representative observation for x_i and both are in the positive side of the r th-hyperplane; or $t_{ir} = t_{jr} = 0$ and $h_{ij} = 1$, i.e., if the well-classified observation x_j is the representative observation for x_i and both are in the negative side of the r th-hyperplane. Thus, constraints (23) and (24) adequately model the in-margin errors for all observations. Furthermore, because of (15) and (16), and those described above, the variables e_{ir} always take values smaller than or equal to 1.

Finally, the third addend, the out-margin errors, will be modeled through the continuous variables $d_{ir} \geq 0$, for $i = 1, \dots, n$, $r = 1, \dots, m$. With the set of variables described above, the out-margin misclassification errors can be adequately modeled through the following constraints:

$$d_{ir} \geq 1 - \omega_r^t x_i - \omega_{r0} - T(2 + t_{ir} - t_{jr} - h_{ij}), \quad \forall i, j \in N(y_i = y_j), r \in M, \quad (12)$$

$$d_{ir} \geq 1 + \omega_r^t x_i + \omega_{r0} - T(2 - t_{ir} + t_{jr} - h_{ij}), \quad \forall i, j \in N(y_i = y_j), r \in M, \quad (13)$$

Constraints (25) are active only if $t_{ir} = 0$ and $t_{jr} = h_{ij} = 1$, that is, if x_j is a well-classified observation in the positive side of \mathcal{H}_r , while x_i is wrong-classified in the negative side of \mathcal{H}_r being x_j the representative observation for x_i (note that if x_i is well-classified then $h_{ii} = 1$ by (22) and then, the constraint cannot be activated). The second set of constraints, namely (26), can be analogously justified in terms of the negative side of \mathcal{H}_r . The main difference of these constraints with respect to (23) and (24) is that (25) and (26) are active only if x_i is wrong-classified.

According to the above constraints, a misclassified observation x_i is penalized in two ways with respect to each hyperplane \mathcal{H}_r . In case that x_i is well-classified with respect to \mathcal{H}_r , but it belongs to the margin, then $e_{ir} = 1 - \text{sign}(\omega_r^t x_i + \omega_{r0})(\omega_r^t x_i + \omega_{r0}) \leq 1$ and $d_{ir} = 0$ ($t_{ir} = t_{jr}$). Otherwise, if x_i is wrong-classified with respect to \mathcal{H}_r , then $d_{ir} = 1 - \text{sign}(\omega_r^t x_i + \omega_{r0})(\omega_r^t x_i + \omega_{r0}) \geq 1$ and $e_{ir} = 0$ ($h_{ij} = 1$ and $t_{ir} \neq t_{jr}$).

We illustrate the rationale of the proposed constraints on the data drawn in Figure 7. Observe that A is not correctly classified since it lies within a cell in which the blue-class is not assigned. Suppose that B, a well-classified observation, is the representative of A ($h_{AB} = 1$), then the model would have to penalize two types of errors. The first one with respect to \mathcal{H}_2 . If we suppose $t_{B2} = 1$, then $t_{A2} = 0$, leading to an activation on constraint (25) being $d_{A2} > 0$. On the other hand, even though A is well-classified with respect to \mathcal{H}_1 , we also have to penalize its margin violation. Again, if we assume $t_{B1} = 1$, then $t_{A1} = 1$, what would activate the constraint (23) being $e_{A1} > 0$.

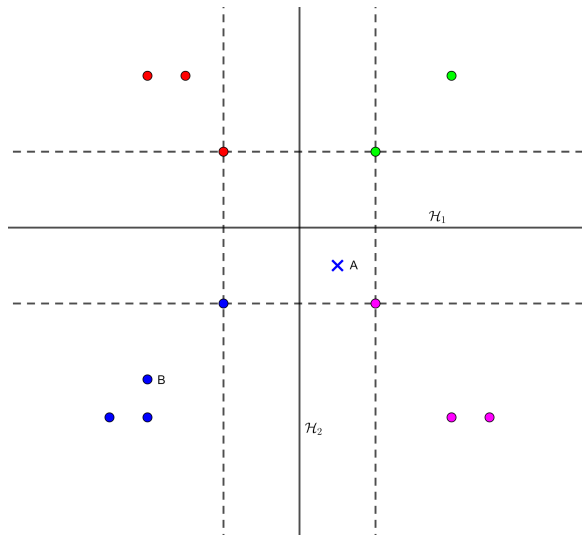


FIGURE 7. Illustration of the in-margin and out-margin constraints of our model.

The above comments can be summarized in the following mathematical programming formulation for the problem:

$$\begin{aligned}
 \min \quad & \|\omega_1\|^2 + C_1 \sum_{i=1}^n \sum_{r=1}^m e_{ir} + C_2 \sum_{i=1}^n \sum_{r=1}^m d_{ir} & (\text{MCSVM}) \\
 \text{s.t.} \quad & (14) - (26) \\
 & \omega_r \in \mathbb{R}^p, \omega_{r0} \in \mathbb{R}, & \forall r \in M, \\
 & d_{ir}, e_{ir} \geq 0, t_{ir} \in \{0, 1\} & \forall i \in N, r \in M, \\
 & h_{ij} \in \{0, 1\}, & \forall i, j \in N, \\
 & z_{is} \in \{0, 1\}, & \forall i \in N, s \in K, \\
 & \xi_i \in \{0, 1\}, & \forall i \in N.
 \end{aligned}$$

(MCSVM) is a mixed integer non linear programming model, whose non-linear terms come from the norm minimization in the objective function and constraints (14), so that they are second order cone representable. In case one chooses the ℓ_1 -norm instead of the Euclidean norm, the model becomes a mixed integer linear programming problem. Therefore, the model is suitable to be solved using any of the available commercial solvers, as Gurobi, CPLEX, etc. The main bottleneck of the above formulation relies on the number $O(n^2)$ of binary variables.

Remark 3.1 (Ramp Loss misclassification errors). *An alternative measure of misclassification training errors is the ramp loss. The ramp loss version of the model is interesting for certain instances since it allows one to improve the robustness against potential outliers. Instead of using out of margin hinge loss errors h_O , the ramp-loss measure consists of penalizing wrong-classified observations by a constant, independently on how far they are from being well-classified. Given an observation/label, (\bar{x}, \bar{y}) , the ramp-loss with respect to \mathbb{H} , is defined as:*

$$\text{RL}((\bar{x}, \bar{y}), \mathbb{H}) = \begin{cases} 0 & \text{if } \bar{x} \text{ is well-classified} \\ 1 & \text{otherwise} \end{cases}$$

Note that, for the training sample, the ramp-loss is represented in our model through the ξ -variables. More specifically, $\text{RL}((x_i, y_i), \mathbb{H}) = \xi_i$ for all $i \in N$. In order to do that we just need to introduce the following modifications on the MINLP problem:

$$\begin{aligned}
 \min \quad & \|\omega_1\|^2 + C_1 \sum_{i=1}^n \sum_{r=1}^m e_{ir} + C_2 \sum_{i=1}^n \xi_i & (\text{MCSVM}_{\text{RL}}) \\
 \text{s.t.} \quad & (14) - (24) \\
 & \omega_r \in \mathbb{R}^p, \omega_{r0} \in \mathbb{R}, & \forall r \in M, \\
 & e_{ir} \geq 0, & \forall i \in N, r \in M, \\
 & h_{ij} \in \{0, 1\}, & \forall i, j \in N, \\
 & z_{is} \in \{0, 1\}, & \forall i \in N, s \in K, \\
 & t_{ir} \in \{0, 1\}, & \forall i \in N, r \in M, \\
 & \xi_i \in \{0, 1\}, & \forall i \in N.
 \end{aligned}$$

3.1. Building the classification rule. Recall that the main goal of multiclass classification is to determine a decision rule such that, given any observation, it is able to assign it a class, i.e., to determine the optimal suitable assignment. Hence, once the solution of (MCSVM) is obtained, the decision rule has to be derived. Given $x \in \mathbb{R}^p$, two different situations are possible: (a) x belongs to a cell with an assigned class; and (b) x belongs to a cell with no training observations inside, so with non assigned class. For the first case, x is assigned to its cell's class. In the second case, different strategies to determine a class for x are possible.

We propose the following assignment rule based on the same allocation methods used in (MCSVM): observations are assigned to their closest well-classified representatives. More specifically, let $s(x)$ be the sign-pattern of x with respect to the optimal arrangement of hyperplanes $\mathbb{H}^* = \{(\omega_1^*, \omega_{10}^*), \dots, (\omega_m^*, \omega_{m0}^*)\}$ obtained from (MCSVM), and let $J = \{j \in \{1, \dots, n\} : \xi_j^* = 0\}$ (here ξ^* stand for the optimal vector obtained by solving (MCSVM)). Then, among all the well-classified observations in the training sample, J , we assign to x the class of the one whose cell is the *closest* (less separated from x). Such a classification of x can be obtained by enumerating all the possible assignments, $O(|J|)$ and computing the distance measure over all of them. Equivalently, one can solve the following mathematical programming problem:

$$\begin{aligned} \min \sum_{j \in J}^n \sum_{\substack{r=1 \\ s(x_j)_r + s(x)_r = 0}}^m \gamma_j |(\omega_r^*)^t x + \omega_{r0}^*| \\ \text{s.t. } \sum_{j \in J}^n \gamma_j = 1, \\ \gamma_j \in \{0, 1\}, \forall j \in J \end{aligned}$$

where $\gamma_j = \begin{cases} 1 & \text{if } x \text{ is assigned to the same cell as } x_j, \\ 0 & \text{otherwise.} \end{cases}$

The integrality condition in the problem above can be relaxed, since the unique constraint in the problem is totally unimodular and thus, the problem is a linear programming problem. Clearly, the solution of the above problem gives the optimal labelling of x with respect to the existing cells in the arrangement.

One could also consider other robust measures for such an assignment following the same paradigm, as min-max error or the like.

3.2. Nonlinear Multiclass Classification. Finally, we analyze a crucial question in any SVM-based methodology, which is whether one can apply the Theory of Kernels in our framework. Using kernels means been able to map the observations (via some transformation $\varphi : \mathbb{R}^p \rightarrow \mathbb{R}^P$) to a higher dimensional space, where the separation of the data is more adequately performed. If the desired transformation, φ , is known, one could transform the data and solve the problem (MCSVM) with a higher number of variables. However, in binary SVMs, formulating the dual of the classification problem, one can observe that it only depends on the original data via the inner products of each pair of observations (originally in \mathbb{R}^p), i.e., through the amounts $x_i^t x_j$ for $i, j = 1, \dots, n$. If the transformation φ is applied to the data, the observations only appear in the (classical

SVM) problem as $\varphi(x_i)^t \varphi(x_j)$ for $j = 1, \dots, n$. Thus, kernels are defined as generalized inner products as $K(a, b) = \varphi(a)^t \varphi(b)$ for each $a, b \in \mathbb{R}^p$, and they can be introduced using any of the well-known families of kernel functions (see e.g., [20]). Moreover, Mercer's theorem gives sufficient conditions for a function $K : \mathbb{R}^p \times \mathbb{R}^p \rightarrow \mathbb{R}$ to be a kernel function (one which is constructed as the inner product of a transformation of the features) what allows one to construct kernel measures that induce transformations. The main advantage of using kernels, apart from a probably better separation in the projected space, is that in binary SVM, the complexity of the *transformed problem is the same as the original one*. More specifically, the dual problems have the same structure and the same number of variables.

Although problem (MCSVM) is a MINLP, and then, duality results do not hold, one can apply decomposition techniques to separate the binary and the continuous variables and then, iterate over the binary variables by recursively solving certain continuous and easier problems (see e.g. Benders decomposition[5, 16]...). The following result, whose proof can be found in the extended version of this paper (see [10]), states that our approach also allows us to find nonlinear classifiers via the kernel tools.

This result is interesting by itself since links the general theory of nonlinear classifiers, very well-known for the standard SVM theory with Euclidean distance, to our multiclass framework. It is worth noting that for a function $h_H(\mathcal{H}_1, \dots, \mathcal{H}_m) = \sum_{r=1}^m \|\omega_r\|^2$ the usual kernel trick construction applies *mutatis-mutandis*. Nevertheless, as pointed out in Section 2.1, we elaborate our approach based on the natural measure of margin that maximizes the minimum separation between classes, namely $h_H(\mathcal{H}_1, \dots, \mathcal{H}_m) = \max\{\|\omega_1\|^2, \dots, \|\omega_m\|^2\}$. This change implies that the mathematical development known for the standard kernel trick does not carry over our new approach without a further analysis. We prove below that in this new framework one can also find nonlinear multiclass classifiers that, as in the standard SVM case, only depend on the transformation by means of inner products of the original data. Hence, extending the kernel trick to this multiclass framework.

Theorem 3.1. *Let $\varphi : \mathbb{R}^p \rightarrow \mathbb{R}^P$ be a transformation of the feature space. Then, one can obtain a multiclass classifier which only depends on the original data by means of the inner products $\varphi(x_i)^t \varphi(x_j)$, for $i, j = 1, \dots, n$.*

Proof. See Appendix 6. □

4. A MATH-HEURISTIC ALGORITHM

As mentioned above, the computational burden for solving (MCSVM), that is a mixed integer non linear programming problem (in which the nonlinearities come from the norm minimization in the objective function), is the combination of the discrete aspects and the non-linearities in the model. In this section we provide some heuristic strategies that allow us to cut down the computational effort by fixing some of the variables. It will also provide good-quality initial feasible solutions when solving, exactly, (MCSVM) using a commercial solver. Two different strategies are provided. The first one consists of applying a variable fixing strategy to reduce the number of h -variables in the model. Note that in principle, n^2 variables of this type are considered in the model. The second

approach consists of fixing to zero some of the z -variables. These nk variables allow us to model assignments between observations and classes. The proposed method is a math-heuristic approach, since after applying the adequate dimensionality reductions, Problem (MCSVM) (or (MCSVM_{RL})) has to be solved. Also, although our strategies do not ensure any kind of optimality certificate, they produce a very good performance as will be shown in our computational experiments. Observe that when classifying datasets, the measure of the efficiency of a decision rule, as ours, is usually assessed by means of the accuracy of the classification on out-of-sample data, whereas the objective value of the proposed model is just an approximated measure of such an accuracy which cannot be computed only with the training data.

Algorithm 1: A math-heuristic approach.

- (1) Apply dimensionality reductions test based on algorithms 2 and 3.
 - (2) Find an initial solution generating k separating hyperplanes.
 - (3) Solve problem (MCSVM) (or (MCSVM_{RL})) up to a prescribed accuracy for the train data.
-

In what follows we describe two strategies to reduce the dimensionality of the problem. These approaches are based on applying clustering techniques to the data. The methods are sensible to the number of clusters. For determining this parameter, we run a hierarchical clustering method, using as termination criterion a given squared Euclidean distance between the observations and their centroids.

4.1. Reducing the h -variables. Our first strategy comes from the fact that for a given observation x_i , there may be several possible choices for h_{ij} to assume the value one with the same final result. Recall that h_{ij} could be equal to one whenever x_j is a well-classified observation in the same class as x_i . The errors e_{ir} and d_{ir} are then computed by using the class of x_j but not the observation x_j itself. Thus, if a set of well-classified observations of the same class is close enough, only one of them can be the representative element of the group. In order to illustrate the procedure, we show in Figure 8 (left) a 4-classes and 24-points instance in which the classes are easily identified by applying any clustering strategy. In such a case (MCSVM) has $(24 \times 24 =)$ 576 h -variables, but if we allow h only to take value 1 at a single point in each cluster, we obtain the same result but reducing to 144 $(24 \times 6 + 18)$, where the 18 comes from the observation mentioned in the formulation in which each well-classified observation can be a representative element of itself) the number of variables. In Figure 8 (right), we show the some clusters based on the data, and a (random) selection of a unique point at each cluster for which the h -values are allowed to be one.

This strategy is summarized in Algorithm 2.

Algorithm 2: Strategy to reduce h -variables.

- (1) Cluster the dataset by *approximated* classes: $\mathcal{C}_1, \dots, \mathcal{C}_c$.
 - (2) Randomly choose a single point at each cluster, $x_{i_j} \in \mathcal{C}_j$, for $j = 1, \dots, c$.
 - (3) Set $h_{ij} = 0$ for $j \notin \{i_1, \dots, i_c\}$.
-

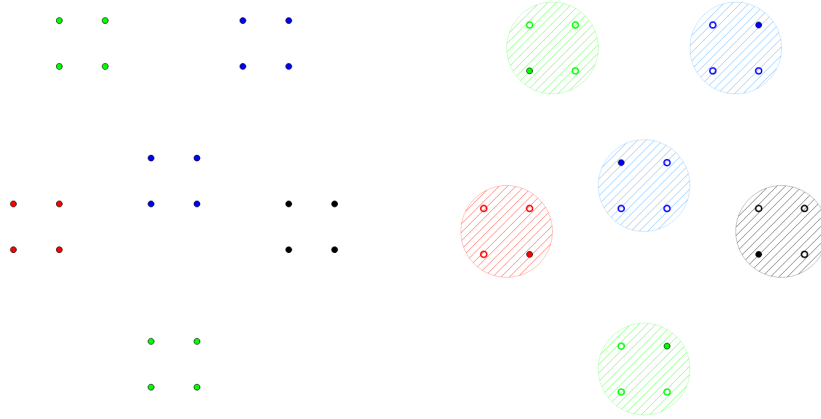


FIGURE 8. Clustering observations for reducing h -variables.

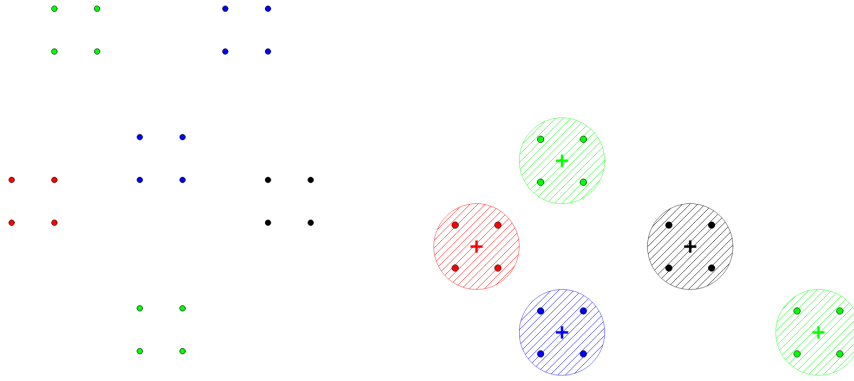


FIGURE 9. Illustration of the strategy to reduce the z -variables.

4.2. Reducing the z -variables. The second strategy consists of fixing to zero some of the point-to-class assignments (z -variables). In the picture shown in Figure 9 (left), one can see a set of points which seems reasonable to group in 5 clusters. One may notice that assignments from the red class to the black class (and vice versa) are rarely going to occur following our approach. This is due to the fact that given this configuration of points, our model would provide a cell for red points located far from a black cell (otherwise it would probably not be maximizing the distance between classes). Following this idea, we derive a procedure to fix some of the z -variables to zero. Another observation that comes out from Figure 9, is that with respect to the red cluster we obtain the following sorting on the set of distances: $d_{green}^1 \leq d_{blue} \leq d_{black} \leq d_{green}^2$. Then, since $d_{green}^1 < d_{green}^2$, we may not take into account the distance to the green cluster on the very right. Thus, we would fix to zero all z_{is} variables that relate the red cluster with the maximum of their minimum distance set, that is, in this case we would fix to zero the z_{is} -variables associated to the black cluster with the red cluster ($d_{green}^1 < d_{black}$ and $d_{blue} < d_{black}$) and vice versa.

The above observations lead us to some strategies for fixing z -variables to zero that we summarize in Algorithm 3.

Algorithm 3: Strategy to reduce z -variables.

- (1) Group the observations in L clusters, being each cluster formed by points of the same class. Let $a_{\ell s}$ be the centroids of the cluster, $\ell = 1, \dots, L$, $s \in \{1, \dots, k\}$.
 - (2) Compute the squared Euclidean distance matrix between centroids:
 $\mathcal{D} = \left(\|a_{\ell s} - a_{q s'}\|^2 \right)$.
 - (3) For each cluster ℓ , $\ell = 1, \dots, L$, assigned to class s , compute the cluster q with class $s' \neq s$ such that $\|a_{\ell s} - a_{q s'}\|^2$ is maximum and greater than a given threshold. For each observation i in cluster q , set $z_{is} = 0$. For each observation \hat{i} in cluster ℓ , set $z_{\hat{i}s'} = 0$.
-

5. EXPERIMENTS

5.1. Real Datasets. In this section we report the results of our computational experience. We have run a series of experiments to analyze the performance of our model in some real datasets widely used in the classification literature, and that are available in the UCI machine learning repository [27]. Summarized information about the datasets is detailed in Table 1. In such a table we report, for each dataset, the number of observations considered in the training sample (n_{Tr}) and test sample (n_{Te}), the number of features (p), the number of classes (k), the number of hyperplanes used in our separation (m), and the number of hyperplanes required by the OVO methodology (m_{OVO}).

Dataset	n_{Tr}	n_{Te}	p	k	m	m_{OVO}
Forest	75	448	28	4	3	6
Glass	75	139	10	6	6	15
Iris	75	75	4	3	2	3
Seeds	75	135	7	3	2	3
Wine	75	103	13	3	2	3
Zoo	75	26	17	7	4	21

TABLE 1. Data sets used in our computational experiments.

For these datasets, we have run both the hinge-loss (MCSVM) and the ramp-loss (MCSVM_{RL}) models, measuring the margin with the ℓ_1 and the ℓ_2 norms. We have performed a 5-cross validation scheme to test each of the approaches. Thus, the data sets were split into 5 train-test random partitions. Then, the models were solved for the training sample and the resulting classification rule was applied to the test sample. We report the average accuracy, ACC, in percentage, of the 5 repetitions of the experiment on test:

$$\text{ACC} = \frac{\#\text{Well Classified Test Observations}}{n_{\text{Te}}} \cdot 100.$$

The parameters of our models were also chosen after applying a grid-based 4-cross validation scheme. In particular, we calibrate the value of m (number of

hyperplanes to be located) and the misclassification costs C_1 and C_2 in:

$$m \in \{2, \dots, k\}, \quad C_1, C_2 \in \{0.1, 0.5, 1, 5, 10\}.$$

For hinge-loss models $C_1 = C_2$, whereas for ramp-loss models we consider $C_1 < C_2$ to give a high penalty to wrong-classified observations. As a result we obtain a misclassification error for each grid point, and we choose the combination of parameters that provide the lowest error. The same methodology was also applied to the other methods: OVO, Weston-Watkins (WW), Crammer-Singer (CS) and Lee, Lin and Wahba (LLW), calibrating the misclassifying cost C in $\{10^i, i = -6, \dots, 6\}$.

The mathematical programming models solving the MCSVM methods were coded in Python 3.6, and solved using Gurobi 7.5.2 on a PC Intel Core i7-7700 processor at 2.81 GHz and 16GB of RAM. The standard methods (OVO, WW and CS) were applied using R-KernLab. Finally, LLW was applied using the software package MSVMpack provided by [24].

In Table 2 we report the average accuracies obtained with our 4 models: ((MCSVM) and (MCSVM_{RL}) with ℓ_1 and ℓ_2 norms) and those obtained with OVO, WW, CS and LLW. The first two columns (ℓ_1 RL and ℓ_1 HL) show the average accuracies of our two approaches (Ramp Loss - RL- and Hing Loss - HL-) using the ℓ_1 -norm. On the other hand, the third and four columns (ℓ_2 RL and ℓ_2 HL) provide the same results for the ℓ_2 -norm. In the last four columns, we report the average accuracies obtained with the OVO, WW, CS and LLW methods. The best accuracies obtained for each dataset are boldfaced in Table 2.

One can observe that our methods always outperform the results obtained by OVO, WW and CS, although the results are rather similar. Actually, running the two samples proportion test among them, we can not ensure significative differences in all cases. Comparing our methods with LLW the results are different. In three out of the 6 databases (Forest, Glass and Iris) our methods are superior to LLW with up to 10% significative differences with respect to the two samples proportion tests. In the remaining three databases (Seeds, Wine and Zoo) the results are similar with no statistical significative differences with respect to the two samples proportion test.

The results indicate that these UCI databases are friendly for linear classifiers (with the only exception of Glass) and thus all these methods perform reasonably well on test prediction. Thus, it is not possible to establish a clear ranking of these classification methods based only on these databases. In order to asses a more complete comparative of the methods we continue the analysis in the following subsection with a battery of more complex datasets.

5.2. Synthetic Experiments. This section reports extra computational experiments over some synthetic instances that allow us to establish some rank of the methods based on their accuracies. We have generated 6 instances of 750 observations in \mathbb{R}^{10} distributed as multivariate normal distributions separated by a constant factor. The instances are denoted as $XCYN$ where X is the number of classes (ranging in $\{2, 3, 4, 7, 10\}$) and Y the number of different multivariate normal distributions (ranging in $\{4, 6, 8, 15, 20\}$). All the instances are available at http://bit.ly/SynthData_MCSVM for benchmarking purposes. Observe that for each instance, the class labels have been randomly assigned to the normal

Dataset	ℓ_1 RL	ℓ_1 HL	ℓ_2 RL	ℓ_2 HL	OVO	WW	CS	LLW
Forest	80.66	80.12	82.30	81.62	82.10	78.40	78.60	72.54
Glass	64.92	64.92	65.32	65.32	58.76	56.25	59.26	57.04
Iris	95.08	95.40	96.44	96.66	93.80	96.44	96.44	84.17
Seeds	93.66	93.66	93.52	93.52	91.02	93.52	93.52	95.46
Wine	95.20	95.20	96.82	96.82	96.34	96.09	96.17	96.31
Zoo	89.75	89.75	89.75	89.75	87.44	87.68	87.68	91.53

TABLE 2. Average accuracies obtained for the real-world instances

distributions. For illustration purposes, a two-dimensional instance generated in the same way that our 10-dimensional instances is shown in Figure 10: the data are generated according to 20 normal distributions which are assigned to 10 classes.

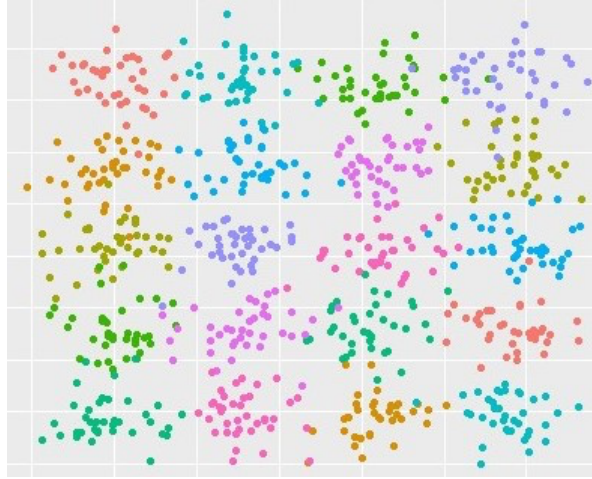


FIGURE 10. A 2-dimensional illustration of our instances.

In Table 3 we report the average accuracies obtained with a 10-fold cross validation experiment, in which 75 observations are taken into the training samples and 675 in the test samples. As before, we have compared our approach (with the Euclidean norm and Hinge-Loss misclassification error) with the existing methodologies: OVO, WW, CS and LLW. The calibration of the parameters was also done as in the previous section.

Dataset	ℓ_2 HL	m	OVO	WW	CS	LLW
2C4N	94.35	2	60.75	60.75	60.75	60.75
3C6N	85.74	3	39.47	41.69	39.03	36.50
4C8N_1	92.76	4	36.46	32.37	29.14	31.86
4C8N_2	91.78	4	48.54	35.14	34.69	39.14
7C15N	88.54	6	27.37	19.64	18.63	20.35
10C20N	85.81	7	29.73	16.17	15.37	15.10

TABLE 3. Average accuracies obtained for the synthetic instances.

One can observe in Table 3 that the results obtained with our approach are much better than those obtained with the other approaches. The generation procedure permits that, in the synthetic instances, separated clouds of points are assigned to the same class. As it can be anticipated, our methodology adapts well to this characteristic whereas the other approaches fail to handle these data. The reader may observe that this type of data are common in real-world datasets. In particular, many diseases are associated to low or high values of certain medical indices thus fitting to this topology in which separated clusters of observations belong to the same class.

Our main conclusion, from the results reported in Table 3, is that our method is adequate for this type of synthetic data, highly outperforming OVO, WW, CS and LLW. Moreover, the accuracy percentages are not only superior but they are also statistically better with respect to the two samples proportion test with a significance level of 1%.

6. CONCLUSIONS

In this paper we propose a novel modeling framework for multiclass classification based on the Support Vector Machine paradigm, in which the different classes are linearly separated and the separation between classes is maximized. We propose two approaches, that depend on the way to account for the misclassification error, to compute an optimal arrangement of hyperplanes subdividing the space into cells, and so that each cell is assigned to a class based on the training data. The models result in Mixed Integer (Linear and Non Linear) Programming problems. Some dimensionality reduction and preprocessing strategies are presented in order to help solvers to find good (optimal) solutions of the corresponding problems. We also prove that an analogous of the kernel trick can be extended to this framework. The performance of this approach is illustrated on some well-known datasets of the multi-category classification literature as well as in some synthetic, but still realistic, examples, in which our approach works remarkably well compared to the existing methodologies. Several extensions of our approach are possible. Among them we would like to mention the use of heuristic algorithms to solve the complex mixed integer nonlinear programs which may alleviate the computational burden of the methodology still keeping high quality solutions. Moreover, our approach could also be extended to the framework of semisupervised learning [6, 35] by assigning unlabelled observations to their closest well-classified cells (which are obtained using the labeled training data). Both research lines seem to be promising and will be the focus of a forthcoming paper.

REFERENCES

- [1] Agarwal, N., Balasubramanian, V.N., Jawahar, C.: Improving multiclass classification by deep networks using dagsvm and triplet loss. *Pattern Recognition Letters* (2018). DOI 10.1016/j.patrec.2018.06.034. URL <http://dx.doi.org/10.1016/j.patrec.2018.06.034>
- [2] Allwein E.L., Schapire R.E., and Singer Y. (2001). Reducing multiclass to binary. Reducing multiclass to binary: a unifying approach for margin classifiers. *The Journal of Machine Learning Research*, 1:113-141.
- [3] Bagirov, A. M., Ugon, J., Webb, D., Ozturk, G., Kasimbeyli, R. (2013). A novel piecewise linear classifier based on polyhedral conic and maxmin separabilities. *TOP* 21(1), 3-24.

- [4] Bahlmann, C., Haasdonk, B., Burkhardt, H.: On-line handwriting recognition with support vector machines ” a kernel approach. In: Proceedings of the Eighth International Workshop on Frontiers in Handwriting Recognition (IWFHR'02), IWFHR '02, pp. 49–. IEEE Computer Society, Washington, DC, USA (2002). URL <http://dl.acm.org/citation.cfm?id=851040.856840>
- [5] Benders, J.F.: Partitioning procedures for solving mixed-variables programming problems. *Numerische mathematik* **4**(1), 238–252 (1962)
- [6] Bennett, K. P., Demiriz, A. (1999). Semi-supervised support vector machines. *Advances in Neural Information processing systems* **11**, 368–374.
- [7] Blanco, V., Ben Ali, S. and Puerto, J. (2014). Revisiting several problems and algorithms in Continuous Location with l_p norms. *Computational Optimization and Applications* **58**(3): 563-595.
- [8] Blanco, V., Puerto, J., Salmern, R. (2018). Locating hyperplanes to fitting set of points: A general framework. *Computers & Operations Research*, **95**, 172-193.
- [9] Blanco, V., Puerto, J., and Rodríguez-Chía, A. M. (2017). On ℓ_p -Support Vector Machines and Multidimensional Kernels. arXiv preprint arXiv:1711.10332.
- [10] Blanco, V., Japn, A., Puerto, J. (2018). Optimal arrangements of hyperplanes for multi-class classification. arXiv preprint arXiv:1810.09167.
- [11] van den Burg G.J.J. and Groenen P.J.F. (2016). GenSVM: A Generalized Multiclass Support Vector Machine. *Journal of Machine Learning Research* **17**(225):1–42, 2016.
- [12] Cortes, C., Vapnik, V.: Support-vector networks. *Machine learning* **20**(3), 273–297 (1995)
- [13] Cover, T., Hart, P.: Nearest neighbor pattern classification. *IEEE transactions on information theory* **13**(1), 21–27 (1967)
- [14] Crammer, K., Singer, Y.: On the algorithmic implementation of multiclass kernel-based vector machines. *Journal of Machine Learning Research* **2**, 265–292 (2001)
- [15] Dietterich T.G. and Bakiri G. (1995). Solving multiclass learning problems via error-correcting output codes. *Journal of Artificial Intelligence Research*, **2**:263-286.
- [16] Geoffrion, A. M. (1972). Generalized Benders decomposition. *Journal of Optimization Theory and Applications*, **10**(4), 237–260.
- [17] Ghaddar, B., and Naoum-Sawaya, J. (2018). High dimensional data classification and feature selection using support vector machines. *European Journal of Operational Research*, **265**(3), 993-1004.
- [18] Y. Guermeur Y. and E. Monfrini E. (2011). A quadratic loss multi-class SVM for which a radius-margin bound applies. *Informatica*, **22**(1):73-96.
- [19] Harris, T.: Quantitative credit risk assessment using support vector machines: Broad versus narrow default definitions. *Expert Systems with Applications* **40**(11), 4404–4413 (2013)
- [20] Horn, D., Demircioglu, A., Bischl, B., Glasmachers, T., and Weihs, C. (2016). *A comparative study on large scale kernelized support vector machines*. *Advances in Data Analysis and Classification*, 1-17.
- [21] K. Ikeda and N. Murata (2005). *Geometrical Properties of Nu Support Vector Machines with Different Norms*. *Neural Computation* **17**(11), 2508-2529.
- [22] K. Ikeda and N. Murata (2005). *Effects of norms on learning properties of support vector machines*. *ICASSP* (5), 241-244
- [23] Kaščélan, V., Kaščélan, L., Novović Burić, M.: A nonparametric data mining approach for risk prediction in car insurance: a case study from the montenegrin market. *Economic research-Ekonomska istraživanja* **29**(1), 545–558 (2016)
- [24] Lauer F., Guermeur Y. (2011). MSVMPack: a Multi-Class Support Vector Machine Package, *Journal of Machine Learning Research*, **12**, 2269–2272.
- [25] Lee, Y., Lin, Y., Wahba, G.: Multicategory support vector machines: Theory and application to the classification of microarray data and satellite radiance data. *Journal of the American Statistical Association* **99**(465), 67–81 (2004)
- [26] Lewis, D.D.: Naive (bayes) at forty: The independence assumption in information retrieval. In: *European conference on machine learning*, pp. 4–15. Springer (1998)
- [27] Lichman, M.: UCI machine learning repository (2013). URL UCIMachineLearningRepository

- [28] López, J., Maldonado, S., and Carrasco, M. (2018). Double regularization methods for robust feature selection and SVM classification via DC programming. *Information Sciences*, 429, 377-389.
- [29] Labbé, M., Martínez-Merino, L. I., and Rodríguez-Chía, A. M. (2018). Mixed Integer Linear Programming for Feature Selection in Support Vector Machine. *Discrete Applied Mathematics*, <https://doi.org/10.1016/j.dam.2018.10.025>.
- [30] Majid, A., Ali, S., Iqbal, M., Kausar, N.: Prediction of human breast and colon cancers from imbalanced data using nearest neighbor and support vector machines. *Computer methods and programs in biomedicine* **113**(3), 792–808 (2014)
- [31] Maldonado, S., Pérez, J., Weber, R., Labbé, M. (2014). Feature selection for support vector machines via mixed integer linear programming. *Information sciences*, 279, 163-175.
- [32] Mangasarian, O.L. *Arbitrary-norm separating plane*. *Oper. Res. Lett.*, 24 (1– 2):15–23 (1999).
- [33] Martínez, D., Millerioux, G. (2000). Support vector committee machines. *European Symposium on Artificial Neural Networks-ESSANN'2000*.
- [34] Meyer, D., Dimitriadou, E., Hornik, K., Weingessel, A. and Leisch, F. e1071: Misc Functions of the Department of Statistics, Probability Theory Group (Formerly: E1071), TU Wien. R package version 1.6-8. <https://CRAN.R-project.org/package=e1071> (2017)
- [35] Ortigosa-Hernández, J., Inza, I., and Lozano, J. A. (2016). Semisupervised multiclass classification problems with scarcity of labeled data: A theoretical study. *IEEE transactions on neural networks and learning systems*, 27(12), 2602-2614.
- [36] Pedregosa, F., Varoquaux, G. , Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M. and Duchesnay, E., *Scikit-learn: Machine Learning in Python*. *Journal of Machine Learning Research* 12, 2825–2830, 2011.
- [37] Platt J.C., Cristianini N., and Shawe-Taylor J. (2000). Large margin DAGs for multi-class classification. In S.A. Solla, T.K. Leen, and K.Mller, editors, *Advances in Neural Information Processing Systems* 12, pages 547-553. MIT Press.
- [38] Radhimeenakshi, S.: Classification and prediction of heart disease risk using data mining techniques of support vector machine and artificial neural network. In: *Computing for Sustainable Global Development (INDIACom)*, 2016 3rd International Conference on, pp. 3107–3111. IEEE (2016)
- [39] Tang, X., Xu, A.: Multi-class classification using kernel density estimation on k-nearest neighbours. *Electronics Letters* **52**(8), 600–602 (2016)
- [40] Üney, F., Türkay, M. (2006). A mixed-integer programming approach to multi-class data classification problem. *European journal of operational research*, 173(3), 910-920.
- [41] Weston, J., Watkins C. (1999). Support vector machines for multi-class pattern recognition. In: *European Symposium on Artificial Neural Networks*, pp. 219–224.

PROOF OF THEOREM 3.1

Let us consider the mathematical programming formulation for our problem:

$$\min \|\omega_1\|^2 + C_1 \sum_{i=1}^n \sum_{r=1}^m e_{ir} + C_2 \sum_{i=1}^n \sum_{r=1}^m d_{ir} \quad (\text{MCSVM})$$

$$\text{s.t. } \frac{1}{2} \|\omega_{r-1}\|^2 \geq \frac{1}{2} \|\omega_r\|^2, \forall r = 2, \dots, m, \quad (14)$$

$$\omega_r^t x_i + w_{r0} \geq -T(1 - t_{ir}), \forall i \in N, r \in M, \quad (15)$$

$$\omega_r^t x_i + w_{r0} \leq T t_{ir}, \forall i \in N, r \in M, \quad (16)$$

$$\sum_{s=1}^k z_{is} = 1, \forall i \in N, \quad (17)$$

$$\|z_i - z_j\|_1 \leq 2\|t_i - t_j\|_1, \forall i, j \in N, \quad (18)$$

$$\xi_i \geq \frac{1}{2} \|z_i - \delta_i\|_1, \forall i \in N, \quad (19)$$

$$\sum_{\substack{j \in N: \\ y_i = y_j}} h_{ij} = 1, \forall i \in N, \quad (20)$$

$$\xi_j + h_{ij} \leq 1, \forall i, j \in N(y_i = y_j), \quad (21)$$

$$h_{ii} = 1 - \xi_i, \forall i \in N, \quad (22)$$

$$\omega_r^t x_i + \omega_{r0} \geq 1 - e_{ir} - T(3 - t_{ir} - t_{jr} - h_{ij}), \forall r \in M, \quad (23)$$

$$\omega_r^t x_i + \omega_{r0} \leq -1 + e_{ir} + T(1 + t_{ir} + t_{jr} - h_{ij}), \forall r \in M, \quad (24)$$

$$d_{ir} \geq 1 - \omega_r^t x_i - \omega_{r0} - T(2 + t_{ir} - t_{jr} - h_{ij}), \forall i, j \in N(y_i = y_j), r \in M, \quad (25)$$

$$d_{ir} \geq 1 + \omega_r^t x_i + \omega_{r0} - T(2 - t_{ir} + t_{jr} - h_{ij}), \forall i, j \in N(y_i = y_j), r \in M, \quad (26)$$

$$\omega_r \in \mathbb{R}^p, \omega_{r0} \in \mathbb{R}, \forall r \in M,$$

$$d_{ir}, e_{ir} \geq 0, t_{ir} \in \{0, 1\}, \forall i \in N, r \in M,$$

$$h_{ij} \in \{0, 1\}, \forall i, j \in N,$$

$$z_{is} \in \{0, 1\}, \forall i \in N, s \in K,$$

$$\xi_i \in \{0, 1\}, \forall i \in N.$$

Note that once the binary variables of our model are fixed, the problem becomes polynomial time solvable and it reduces to find the coordinates of the coefficients and intercepts of the hyperplanes and the different misclassifying errors. In particular, it is clear that the MINLP formulation for the problem is equivalent to:

$$\begin{aligned} & \min_{h, z, t, \xi} \Phi(h, z, t, \xi) \\ & \text{s.t. (17) - (22),} \\ & \quad h_{ij} \in \{0, 1\}, \quad \forall i, j \in N, \\ & \quad t_{ir} \in \{0, 1\} \quad \forall i \in N, r \in M, \\ & \quad z_{is} \in \{0, 1\}, \quad \forall i \in N, s \in K, \\ & \quad \xi_i \in \{0, 1\}, \quad \forall i \in N. \end{aligned}$$

where Φ is the evaluation of the margin and hinge-loss errors for any assignment provided by the binary variables. That is,

$$\begin{aligned} \Phi(h, z, t, \xi) &= \min_{\omega, \omega_0, e, d} \frac{1}{2} \|\omega_1\|^2 + C_1 \sum_{i=1}^n \sum_{r=1}^m e_{ir} + C_2 \sum_{i=1}^n \sum_{r=1}^m d_{ir} \quad (\text{Eval}_\Phi) \\ & \text{s.t. (14), (23) - (26),} \\ & \quad \omega_r \in \mathbb{R}^d, \omega_{r0} \in \mathbb{R}, \quad \forall r \in M, \\ & \quad d_{ir}, e_{ir} \geq 0 \quad \forall i \in N, r \in M. \end{aligned}$$

The above problem would be separable provided that the first $m - 1$ constraints (14) were relaxed. For the sake of simplicity in the notation, we consider the

following functions, $\kappa^\ell : \{0, 1\} \rightarrow \mathbb{R}$ for $\ell = 1, 2, 3$, defined as:

$$\kappa^1(t) := T(1 - t), \quad \kappa^2(t) := Tt, \quad \kappa^3(t) := -1 + Tt$$

for $t \in \{0, 1\}$. Note that $\kappa^1(0) = \kappa^2(1) = T$, $\kappa^1(1) = \kappa^2(0) = 0$, and that $\kappa^3(0) = -1$, $\kappa^3(1) = T - 1$.

Based on the separability mentioned above, we introduce another instrumental family of problems for all $r = 2, \dots, m$, namely,

$$\begin{aligned} \Phi_r(h, z, t, \xi, \omega_1) &= \min_{\omega_r, \omega_{r0}, e, d} C_1 \sum_{i=1}^n e_{ir} + C_2 \sum_{i=1}^n d_{ir} \\ \text{s.t.} \quad &\frac{1}{2} \|\omega_r\|^2 - \frac{1}{2} \|\omega_1\|^2 \leq 0, \\ &-\omega_{ir}^t x_i - \omega_{r0} \leq \kappa^1(t_{ir}), \forall i, r, \\ &\omega_{ir}^t x_i + \omega_{r0} \leq \kappa^2(t_{ir}), \forall i, r, \\ &-\omega_r^t x_i - \omega_{r0} - e_{ir} \leq \kappa^3(u_{ijr}^+), \forall i, j, r, \\ &\omega_r^t x_i + \omega_{r0} - e_{ir} \leq \kappa^3(u_{ijr}^-), \forall i, j, r, \\ &-\omega_r^t x_i - \omega_{r0} - d_{ir} \leq \kappa^3(q_{ijr}^+), \forall i, j (y_i = y_j), \\ &\omega_r^t x_i + \omega_{r0} - d_{ir} \leq \kappa^3(q_{ijr}^-), \forall i, j (y_i = y_j), \\ &-d_{ir} \leq 0, \forall i, \\ &-e_{ir} \leq 0, \forall i, \\ &\omega_r \in \mathbb{R}^d, \omega_{r0} \in \mathbb{R}, \end{aligned} \tag{SP}_r$$

where for simplifying the notation we have introduced the auxiliary variables $u_{ijr}^+ := 3 - t_{ir} - t_{jr} - h_{ij}$, $u_{ijr}^- := 1 + t_{ir} + t_{jr} - h_{ij}$, $q_{ijr}^+ = 2 + t_{ir} - h_{ij} - t_{jr}$ and $q_{ijr}^- = 2 + t_{jr} - h_{ij} - t_{ir}$, for $i, j \in N$ and $r \in M$.

Observe that Φ_r , apart from the first constraint, only considers variables associated to the r th hyperplane.

Moreover, we need another problem that accounts for the first part of Φ .

$$\begin{aligned} \Phi_1(h, z, t, \xi) &= \min_{\omega_1, \omega_{10}, e, d} \frac{1}{2} \|\omega_1\|_1^2 + C_1 \sum_{i=1}^n e_{i1} + C_2 \sum_{i=1}^n d_{i1} \\ \text{s.t.} \quad &-\omega_1^t x_i - \omega_{10} \leq \kappa^1(t_{i1}), \forall i, \\ &\omega_1^t x_i + \omega_{10} \leq \kappa^2(t_{i1}), \forall i, \\ &-\omega_1^t x_i - \omega_{10} - e_{i1} \leq \kappa^3(u_{ij1}^+), \forall i, j, \\ &\omega_1^t x_i + \omega_{10} - e_{i1} \leq \kappa^3(u_{ij1}^-), \forall i, j, r, \\ &-\omega_1^t x_i - \omega_{10} - d_{i1} \leq \kappa^3(q_{ij1}^+), \forall i, j (y_i = y_j), \\ &\omega_1^t x_i + \omega_{10} - d_{i1} \leq \kappa^3(q_{ij1}^-), \forall i, j (y_i = y_j), \\ &-d_{i1} \leq 0, \forall i, \\ &-e_{i1} \leq 0, \forall i, \\ &\omega_1 \in \mathbb{R}^d, \omega_{10} \in \mathbb{R}, \end{aligned} \tag{SP}_1$$

Thus, using the above notation, (MCSVM) is equivalent to the following problem:

$$\begin{aligned}
& \min_{h,z,t,\xi,\omega_1} \Phi_1(h, z, t, \xi) + \sum_{r=2}^m \Phi_r(h, z, t, \xi, \omega_1) \\
& \text{s.t. (17) - (22)} \\
& \quad h_{ij} \in \{0, 1\}, \quad \forall i, j \in N, \\
& \quad t_{ir} \in \{0, 1\} \quad \forall i \in N, r \in M, \\
& \quad z_{is} \in \{0, 1\}, \quad \forall i \in N, s \in K, \\
& \quad \xi_i \in \{0, 1\}, \quad \forall i \in N.
\end{aligned}$$

Observe that the above problem only accounts for ω_1 and the binary variables. Once they are fixed can be plugged into the (SP_r) , $r = 1, \dots, m$ subproblems and then it allows one to find the optimal values of the continuous variables. The elements $\kappa^1(t_{ir}), \kappa^2(t_{ir}), \kappa^3(u_{ijr}^+), \kappa^3(u_{ijr}^-), \kappa^3(q_{ijr}^+)$, and $\kappa^3(q_{ijr}^-)$ are fixed constants once the binary variables are fixed.

In order to solve the problem for a fixed set of binary variables we can proceed recursively, solving independently (SP_r) for all $r = 2, \dots, m$, and then combining their solutions with (SP_1) .

Therefore, to get that goal we apply Lagrangean duality to obtain an exact dual reformulation. Indeed, relaxing the constraints of (SP_r) with dual multipliers $\mu_r \geq 0$ (assuming that $\mu_r^0 \neq 0$) and denoting by $\alpha_{ir} = \alpha_{ir} = \mu_{ir}^1 - \mu_{ir}^2 + \sum_{j:y_i=y_j} (\mu_{ijr}^3 - \mu_{ijr}^4 + \mu_{ijr}^5 - \mu_{ijr}^6)$, for all $i = 1, \dots, n$ and $r = 1, \dots, m$;

after some derivations that can be followed in Lemma .1, one can check that evaluating Φ for given values of the binary variables, can be obtained solving the continuous optimization problem (JSP). (All details can be found in Lemma .1.)

Next, we analyze how the evaluation of the function Φ , given in problem (Eval_Φ) , depends on the original data. In order to do that we find the optimal solutions of (JSP). Dualizing the constraints that correspond to Φ_1 with multipliers $\mu_1 \geq 0$ and those where the variables Γ_r appear, with dual multipliers γ_r , the Lagrangean function of the problem (JSP) results in:

$$\begin{aligned}
 \mathcal{L}(\omega_1, \omega_{01}, d, e; \mu) &= \frac{1}{2} \|\omega_1\|^2 (1 - \sum_{r=2}^m \gamma_r \mu_r^0) + C_1 \sum_{i=1}^n e_{i1} + C_2 \sum_{i=1}^n d_{i1} + \sum_{r=2}^m \Gamma_r (1 - \gamma_r) \\
 &+ \sum_{r=2}^m \left(\frac{\gamma_r}{2\mu_r^0} \sum_{i,j=1}^n \alpha_{ir} \alpha_{jr} x_i^t x_j + \sum_{i,j:u_{ijr}^+=0} \mu_{ijr}^3 \right. \\
 &\left. + \sum_{i,j:u_{ijr}^-=0} \mu_{ijr}^4 + \sum_{i,j:q_{ijr}^+=0} \mu_{ijr}^5 + \sum_{i,j:q_{ijr}^-=0} \mu_{ijr}^6 \right) \\
 &+ \sum_i \mu_{i1}^1 \left(-\omega_1^t x_i - \omega_{10} - \kappa^1(t_{i1}) \right) \\
 &+ \sum_i \mu_{i1}^2 \left(\omega_1^t x_i + \omega_{10} - \kappa^2(t_{i1}) \right) \\
 &+ \sum_i \mu_{ij1}^3 \left(-\omega_1^t x_i - \omega_{10} - e_{i1} - \kappa^3(\bar{u}_{ij1}^+) \right) \\
 &+ \sum_i \mu_{ij1}^4 \left(\omega_1^t x_i + \omega_{10} - e_{i1} - \kappa^3(\bar{u}_{ij1}^-) \right) \\
 &+ \sum_{i,j} \mu_{ij1}^5 \left(-d_{ir} - \omega_1^t x_i - \omega_{10} - \kappa^3(\bar{q}_{ij1}^+) \right) \\
 &+ \sum_{i,j} \mu_{ij1}^6 \left(-d_{ir} + \omega_1^t x_i + \omega_{10} - \kappa^3(\bar{q}_{ij1}^-) \right) \\
 &+ \sum_i \mu_{i1}^7 (-d_{i1}) + \sum_i \mu_{i1}^8 (-e_{i1})
 \end{aligned}$$

and its KKT optimality conditions reduce to:

- $(1 - \sum_{r=2}^m \gamma_r \mu_r^0) \omega_1 = \sum_{i=1}^n \left(\mu_{i1}^1 - \mu_{i1}^2 + \sum_{j:y_i=y_j} (\mu_{ij1}^3 - \mu_{ij1}^4 + \mu_{ij1}^5 - \mu_{ij1}^6) \right) x_i.$
- $\sum_{i=1}^n \left(\mu_{i1}^1 - \mu_{i1}^2 + \sum_{j:y_i=y_j} (\mu_{ij1}^3 - \mu_{ij1}^4 + \mu_{ij1}^5 - \mu_{ij1}^6) \right) = 0.$
- $\sum_{j:y_i=y_j} (\mu_{ij1}^3 + \mu_{ij1}^4) + \mu_{i1}^8 = C_1,$ for all $i.$
- $\sum_{j:y_i=y_j} (\mu_{ijr}^5 + \mu_{ijr}^6) + \mu_{ir}^7 = C_2,$ for all $i.$
- $1 - \gamma_r = 0,$ for all $r = 2, \dots, m.$
- $\mu \geq 0, \gamma \geq 0.$

Using the same notation as before, $\alpha_{i1} = \mu_{i1}^1 - \mu_{i1}^2 + \sum_{j:y_i=y_j} (\mu_{ij1}^3 - \mu_{ij1}^4 + \mu_{ij1}^5 - \mu_{ij1}^6),$ for all $i = 1, \dots, n,$ and simplifying the expressions using the KKT conditions and the complementary slackness conditions we get that the strong dual of (JSP) is:

$$\begin{aligned}
& \max_{\omega, \omega_0, d, e; \mu} \frac{1}{2(1 - \sum_{r=2}^m \mu_r^0)} \sum_{i,j=1}^n \alpha_{i1} \alpha_{j1} x_i^t x_j + \sum_{i,j:u_{ij1}^+=0} \mu_{ijr}^3 + \sum_{i,j:u_{ij1}^-=0} \mu_{ijr}^4 \\
& + \sum_{i,j:q_{ij1}^+=0} \mu_{ijr}^5 + \sum_{i,j:q_{ij1}^-=0} \mu_{ijr}^6 + \sum_{r=2}^m \left(\frac{1}{2\mu_r^0} \sum_{i,j=1}^n \alpha_{ir} \alpha_{jr} x_i^t x_j \right. \\
& \left. + \sum_{i,j:u_{ijr}^+=0} \mu_{ijr}^3 + \sum_{i,j:u_{ijr}^-=0} \mu_{ijr}^4 + \sum_{i,j:q_{ijr}^+=0} \mu_{ijr}^5 + \sum_{i,j:q_{ijr}^-=0} \mu_{ijr}^6 \right) \\
& \text{s.t. } \sum_{i=1}^n \alpha_{ir} = 0, \forall r \tag{DJSP} \\
& \sum_{i=1}^n \alpha_{i1} x_i = (1 - \sum_{r=2}^m \mu_r^0) \omega_1, \\
& \sum_{i=1}^n \alpha_{ir} x_i = \mu_r^0 \omega_r, \forall r \geq 2 \\
& \sum_{j:u_{ijr}^+=0} \mu_{ijr}^3 + \sum_{j:u_{ijr}^-=0} \mu_{ijr}^4 \leq C_1, \forall i, \\
& \sum_{j:q_{ijr}^+=0} \mu_{ijr}^5 + \sum_{j:q_{ijr}^-=0} \mu_{ijr}^6 \leq C_2, \forall i, r, \\
& \mu_{ir}^1 = 0, \forall i, r \text{ with } \bar{t}_{ir} = 0, \\
& \mu_{ir}^2 = 0, \forall i, r \text{ with } \bar{t}_{ir} = 1, \\
& \alpha_{ir} = \mu_{ir}^1 - \mu_{ir}^2 + \sum_{i,j:u_{ij1}^+=0} \mu_{ijr}^3 - \sum_{i,j:u_{ij1}^-=0} \mu_{ijr}^4 + \sum_{i,j:q^-+ij1=0} \mu_{ijr}^5 \tag{27} \\
& - \sum_{i,j:q_{ij1}^-=0} \mu_{ijr}^6, \forall i, \\
& \mu_{\mathbf{r}} \geq 0, \omega_{\mathbf{r}} \in \mathbb{R}^d, \omega_{r0} \in \mathbb{R}, \forall r = 1, \dots, m
\end{aligned}$$

Note that the objective function of (DJSP) only depends of the x -input data through the inner products $x_i^t x_j$ for $i, j = 1, \dots, n$, and also the expressions of ω_1 from the dual variables is given as:

$$(1 - \sum_{r=2}^m \mu_r^0) \omega_1 = \sum_{i=1}^n \alpha_{i1} x_i.$$

The dependence of ω_1 with other ω_r in the primal formulation is only through the nonincreasing sorted values of $\|\omega_1\|, \dots, \|\omega_m\|$ in which we now that the largest value is $\|\omega_1\|$. Thus, solving the dual problem (DJSP) allows us to determine all the optimal hyperplanes ω_r , for all $r = 1, \dots, m$. In case a transformation φ is performed to the input data, the dependence of the data in the problem will

be through the inner products $\varphi(x_i)^t \varphi(x_j)$ for all $i, j = 1, \dots, n$, and the kernel Theory can be applied. \square

Lemma .1. *The evaluation of Φ for given values of the binary variables \bar{t} , $\bar{\xi}$, \bar{h} and \bar{z} can be obtained solving the following continuous optimization problem:*

$$\begin{aligned}
 \hat{\Phi}(h, z, t, \xi) = \min & \left\{ \frac{1}{2} \|\omega_1\|^2 + C_1 \sum_{i=1}^n e_{i1} + C_2 \sum_{i=1}^n d_{i1} + \sum_{r=2}^m \Gamma_r \right\} \\
 \text{s.t.} & \left(-\frac{\mu_r^0}{2} \|\omega_1\|^2 + \frac{1}{2\mu_r^0} \sum_{i,j=1}^n \alpha_{ir} \alpha_{jr} x_i^t x_j \right. \\
 & + \sum_{i,j:u_{ijr}^+=0} \mu_{ijr}^3 + \sum_{i,j:u_{ijr}^-=0} \mu_{ijr}^4 \\
 & \left. + \sum_{i,j:q_{ijr}^+=0} \mu_{ijr}^5 + \sum_{i,j:q_{ijr}^-=0} \mu_{ijr}^6 + C_1 \right) \leq \Gamma_r, \forall r \geq 2, \\
 & -\omega_1^t x_i - \omega_{10} \leq \kappa^1(t_{i1}), \forall i, \\
 & \omega_1^t x_i + \omega_{10} \leq \kappa^2(t_{i1}), \forall i, \\
 & -\omega_1^t x_i - \omega_{10} - e_{i1} \leq \kappa^3(u_{ij1}^+), \forall i, j, \\
 & \omega_1^t x_i + \omega_{10} - e_{i1} \leq \kappa^3(u_{ij1}^-), \forall i, j, r, \\
 & -\omega_1^t x_i - \omega_{10} - d_{i1} \leq \kappa^3(q_{ij1}^+), \forall i, j (y_i = y_j), \\
 & \omega_1^t x_i + \omega_{10} - d_{i1} \leq \kappa^3(q_{ij1}^-), \forall i, j (y_i = y_j), \\
 & -d_{i1} \leq 0, \forall i, \\
 & -e_{i1} \leq 0, \forall i, \\
 & \omega_1 \in \mathbb{R}^d, \omega_{10} \in \mathbb{R}, \\
 & \sum_{j:u_{ijr}^+=0} \mu_{ijr}^3 + \sum_{j:u_{ijr}^-=0} \mu_{ijr}^4 + \mu_{ir}^8 \geq C_1, \forall i, \\
 & \sum_{j:q_{ijr}^+=0} \mu_{ijr}^5 + \sum_{j:q_{ijr}^-=0} \mu_{ijr}^6 \leq C_2, \forall i, r, \\
 & \mu_{ir}^1 = 0, \forall i, r \text{ with } \bar{t}_{ir} = 0, \\
 & \mu_{ir}^2 = 0, \forall i, r \text{ with } \bar{t}_{ir} = 1, \\
 & \alpha_{ir} = \mu_{ir}^1 - \mu_{ir}^2 + \sum_{i,j:u_{ij1}^+=0} \mu_{ijr}^3 - \sum_{i,j:u_{ij1}^-=0} \mu_{ijr}^4 + \sum_{i,j:q_{ij1}^+=0} \mu_{ijr}^5 \\
 & - \sum_{i,j:q_{ij1}^-=0} \mu_{ijr}^6, \forall i, \\
 & \mu \geq 0.
 \end{aligned} \tag{JSP}$$

where $\mu_r \geq 0$ are the dual multipliers of the constraints in Problem (SP_r).

Proof. In order to prove the result, in what follows we derive the optimality conditions of the Lagrangean dual of (SP_r) for $r = 2, \dots, m$. Relaxing the constraints of (SP_r) with dual multipliers $\mu_r \geq 0$, the Lagrangean function for given values of the binary variables \bar{t} , $\bar{\xi}$, \bar{h} and \bar{z} (and consequently for values \bar{u}^+ , \bar{u}^- , \bar{q}^+ and \bar{q}^-) is:

$$\begin{aligned} \mathcal{L}_r(\omega_r, \omega_{0r}, d, e; \mu) &= C_1 \sum_{i=1}^n e_{ir} + C_2 \sum_{i=1}^n d_{ir} + \mu_r^0 \left(\frac{1}{2} \|\omega_r\|^2 - \frac{1}{2} \|\omega_1\|^2 \right) \\ &\quad + \sum_i \mu_{ir}^1 \left(-\omega_r^t x_i - \omega_{r0} - \kappa^1(t_{ir}) \right) + \sum_i \mu_{ir}^2 \left(\omega_r^t x_i + \omega_{r0} - \kappa^2(t_{ir}) \right) \\ &\quad + \sum_{i,j} \mu_{ijr}^3 \left(-\omega_r^t x_i - \omega_{r0} - e_{ir} - \kappa^3(\bar{u}_{ijr}^+) \right) \\ &\quad + \sum_{i,j} \mu_{ijr}^4 \left(\omega_r^t x_i + \omega_{r0} - e_{ir} - \kappa^3(\bar{u}_{ijr}^-) \right) \\ &\quad + \sum_{i,j} \mu_{ijr}^5 \left(-d_{ir} - \omega_r^t x_i - \omega_{r0} - \kappa^3(\bar{q}_{ijr}^+) \right) \\ &\quad + \sum_{i,j} \mu_{ijr}^6 \left(-d_{ir} + \omega_r^t x_i + \omega_{r0} - \kappa^4(\bar{q}_{ijr}^-) \right) \\ &\quad + \sum_i \mu_{ir}^7 (-d_{ir}) + \sum_i \mu_{ir}^8 (-e_{ir}). \end{aligned}$$

Therefore, the KKT optimality conditions read as:

- $\mu_r^0 \omega_r = \sum_{i=1}^n \left(\mu_{ir}^1 - \mu_{ir}^2 + \sum_{j:y_i=y_j} (\mu_{ijr}^3 - \mu_{ijr}^4 + \mu_{ijr}^5 - \mu_{ijr}^6) \right) x_i$.
- $\sum_{i=1}^n \left(\mu_{ir}^1 - \mu_{ir}^2 + \sum_{j:y_i=y_j} (\mu_{ijr}^3 - \mu_{ijr}^4 + \mu_{ijr}^5 - \mu_{ijr}^6) \right) = 0$.
- $\sum_{j:y_i=y_j} (\mu_{ijr}^3 + \mu_{ijr}^4) + \mu_{ir}^8 = C_1$, for all i .
- $\sum_{j:y_i=y_j} (\mu_{ijr}^5 + \mu_{ijr}^6) + \mu_{ir}^7 = C_2$, for all i .
- $\mu_r \geq 0$.

First of all, we observe that if $\|\omega_r\| < \|\omega_1\|$ at optimality then $\mu_r^0 = 0$ and actually, we do not have to consider the corresponding constraint nor the addend $\frac{\mu_r^0}{2} (\|\omega_r\|^2 - \|\omega_1\|^2)$ in the Lagrangean function. Hence, denoting by $\alpha_{ir} = \mu_{ir}^1 - \mu_{ir}^2 + \sum_{j:y_i=y_j} (\mu_{ijr}^3 - \mu_{ijr}^4 + \mu_{ijr}^5 - \mu_{ijr}^6)$, for all $i = 1, \dots, n$, and assuming that $\mu_r^0 \neq 0$, the dual of (SP_r) reads as:

$$\begin{aligned} \max_{\omega_r, \omega_{r0}, d, e; \mu} & - \frac{\mu_r^0}{2} \|\omega_1\|^2 + \frac{1}{2\mu_r^0} \sum_{i,j=1}^n \alpha_{ir} \alpha_{jr} x_i^t x_j - \sum_i \mu_{ir}^1 \kappa^1(\bar{t}_{ir}) - \sum_i \mu_{ir}^2 \kappa^2(\bar{t}_{ir}) \\ & - \sum_i \mu_{ir}^3 \kappa^3(\bar{u}_{ijr}^+) - \sum_i \mu_{ir}^4 \kappa^4(\bar{u}_{ijr}^-) \end{aligned}$$

$$\begin{aligned}
 & - \sum_{i,j} \mu_{ijr}^5 \kappa^5(\bar{q}_{ijr}^+) - \sum_{i,j} \mu_{ijr}^6 \kappa^6(\bar{q}_{ijr}^-) \\
 \text{s.t. } & \sum_{i=1}^n \alpha_{ir} x_i = \mu_r^0 \omega_r, \\
 & \sum_{i=1}^n \alpha_{ir} = 0, \\
 & \sum_{j:y_i=y_j} (\mu_{ijr}^3 + \mu_{ijr}^4) \leq C_1, \forall i, \\
 & \sum_{j:y_i=y_j} (\mu_{ijr}^5 + \mu_{ijr}^6) \leq C_2, \forall i, \\
 & \alpha_{ir} = \mu_{ir}^1 - \mu_{ir}^2 + \sum_{i,j:u_{ij1}^+=0} \mu_{ijr}^3 - \sum_{i,j:u_{ij1}^-=0} \mu_{ijr}^4 + \sum_{i,j:q_{ij1}^+=0} \mu_{ijr}^5 \\
 & - \sum_{i,j:q_{ij1}^-=0} \mu_{ijr}^6, \forall i, \\
 & \mu_r \geq 0.
 \end{aligned}$$

Let us simplify further the expressions above. We observe that:

$$\begin{aligned}
 & \sum_i \mu_{ir}^1 \kappa^1(\bar{t}_{ir}) + \sum_i \mu_{ir}^2 \kappa^2(\bar{t}_{ir}) = 0, \\
 & - \sum_{i,j} \mu_{ijr}^3 \kappa^3(\bar{u}_{ijr}^+) - \sum_{i,j} \mu_{ijr}^4 \kappa^3(\bar{u}_{ijr}^-) = \sum_{\substack{i,j: \\ \bar{u}_{ijr}^+=0}} \mu_{ijr}^3 + \sum_{\substack{i,j: \\ \bar{u}_{ijr}^-=0}} \mu_{ijr}^4,
 \end{aligned}$$

and

$$- \sum_{i,j} \mu_{ijr}^5 \kappa^3(\bar{q}_{ijr}^+) - \sum_{i,j} \mu_{ijr}^6 \kappa^4(\bar{q}_{ijr}^-) = \sum_{\substack{i,j: \\ \bar{q}_{ijr}^+=0}} \mu_{ijr}^5 + \sum_{\substack{i,j: \\ \bar{q}_{ijr}^-=0}} \mu_{ijr}^6.$$

Using those equations and substituting the problem becomes:

$$\begin{aligned}
 \max_{\omega_r, \omega_0, d, e; \mu} & - \frac{\mu_r^0}{2} \|\omega_1\|^2 + \frac{1}{2\mu_r^0} \sum_{i,j=1}^n \alpha_{ir} \alpha_{jr} x_i^t x_j + \sum_{i,j:u_{ijr}^+=0} \mu_{ijr}^3 + \sum_{i,j:u_{ijr}^-=0} \mu_{ijr}^4 \\
 & + \sum_{i,j:q_{ijr}^+=0} \mu_{ijr}^5 + \sum_{i,j:q_{ijr}^-=0} \mu_{ijr}^6 \\
 \text{s.t. } & \sum_{i=1}^n \alpha_{ir} x_i = \mu_r^0 \omega_r, \\
 & \sum_{i=1}^n \alpha_{ir} = 0, \forall r, \\
 & \sum_{j:u_{ijr}^+=0} \mu_{ijr}^3 + \sum_{j:u_{ijr}^-=0} \mu_{ijr}^4 \leq C_1, \forall i,
 \end{aligned}$$

$$\begin{aligned}
& \sum_{j:q_{ijr}^+=0} \mu_{ijr}^5 + \sum_{j:q_{ijr}^-=0} \mu_{ijr}^6 \leq C_2, \forall i, r, & (\text{DSP}_r) \\
& \mu_{ir}^1 = 0, \forall i, r \text{ with } \bar{t}_{ir} = 0, \\
& \mu_{ir}^2 = 0, \forall i, r \text{ with } \bar{t}_{ir} = 1, \\
& \alpha_{ir} = \mu_{ir}^1 - \mu_{ir}^2 + \sum_{i,j:u_{ij1}^+=0} \mu_{ijr}^3 - \sum_{i,j:u_{ij1}^-=0} \mu_{ijr}^4 + \sum_{i,j:q_{ij1}^+=0} \mu_{ijr}^5 & (29) \\
& - \sum_{i,j:q_{ij1}^-=0} \mu_{ijr}^6, \forall i, \\
& \mu \geq 0.
\end{aligned}$$

Using the strong duality in all the subproblems (SP_r) for $r = 2, \dots, m$, we can obtain the following expansion of the join subproblem (SP_r) that allows one the evaluation of Φ defined in problem (Eval_Φ) .

$$\begin{aligned}
\hat{\Phi}(h, z, t, \xi) = \min & \left\{ \frac{1}{2} \|\omega_1\|^2 + C_1 \sum_{i=1}^n e_{i1} + C_2 \sum_{i=1}^n d_{i1} \right. \\
& + \max_{\omega_r, \omega_0, d, e; \mu} \sum_{r=2}^m \left(-\frac{\mu_r^0}{2} \|\omega_1\|^2 + \frac{1}{2\mu_r^0} \sum_{i,j=1}^n \alpha_{ir} \alpha_{jr} x_i^t x_j + \sum_{i,j:u_{ijr}^+=0} \mu_{ijr}^3 \right. & (30) \\
& \left. \left. + \sum_{i,j:u_{ijr}^-=0} \mu_{ijr}^4 + \sum_{i,j:q_{ijr}^+=0} \mu_{ijr}^5 + \sum_{i,j:q_{ijr}^-=0} \mu_{ijr}^6 \right) \right. \\
\text{s.t.} & -\omega_1^t x_i - \omega_{10} \leq \kappa^1(t_{i1}), \forall i, \\
& \omega_1^t x_i + \omega_{10} \leq \kappa^2(t_{i1}), \forall i, \\
& -\omega_1^t x_i - \omega_{10} - e_{i1} \leq \kappa^3(u_{ij1}^+), \forall i, j, \\
& \omega_1^t x_i + \omega_{10} - e_{i1} \leq \kappa^3(u_{ij1}^-), \forall i, j, r, \\
& -\omega_1^t x_i - \omega_{10} - d_{i1} \leq \kappa^3(q_{ij1}^+), \forall i, j (y_i = y_j), \\
& \omega_1^t x_i + \omega_{10} - d_{i1} \leq \kappa^3(q_{ij1}^-), \forall i, j (y_i = y_j), \\
& -d_{i1} \leq 0, \forall i, \\
& -e_{i1} \leq 0, \forall i, \\
& \omega_1 \in \mathbb{R}^d, \omega_{10} \in \mathbb{R}, \\
& \sum_{j:u_{ijr}^+=0} \mu_{ijr}^3 + \sum_{j:u_{ijr}^-=0} \mu_{ijr}^4 \leq C_1, \forall i, \\
& \sum_{j:q_{ijr}^+=0} \mu_{ijr}^5 + \sum_{j:q_{ijr}^-=0} \mu_{ijr}^6 \leq C_2, \forall i, r, \\
& \mu_{ir}^1 = 0, \forall i, r \text{ with } \bar{t}_{ir} = 0,
\end{aligned}$$

$$\begin{aligned}
& \mu_{ir}^2 = 0, \forall i, r \text{ with } \bar{t}_{ir} = 1, \\
& \alpha_{ir} = \mu_{ir}^1 - \mu_{ir}^2 + \sum_{i,j:u_{ij1}^+=0} \mu_{ijr}^3 - \sum_{i,j:u_{ij1}^-=0} \mu_{ijr}^4 + \sum_{i,j:q_{ij1}^+=0} \mu_{ijr}^5 \quad (31) \\
& - \sum_{i,j:q_{ij1}^-=0} \mu_{ijr}^6, \forall i, \\
& \mu \geq 0.
\end{aligned}$$

The usual transformation of the maximum in the objective function gives rise to the equivalent reformulation of the above problem as (JSP). \square

IEMATH-GR, UNIVERSIDAD DE GRANADA, SPAIN.
E-mail address: vblanco@ugr.es

IMUS, UNIVERSIDAD DE SEVILLA, SPAIN.
E-mail address: ajapon1@us.es; puerto@us.es