

Mixed-Integer Optimization for Semi-Supervised Learning with Cardinality Constraints: Support Vector Machines, Classification Trees, and Random Forests

Dissertation approved by Faculty IV of Trier University
for the award of the academic degree
Doctor of Natural Science (Dr. rer. nat.)

by

Maria Eduarda Pinheiro

Trier, 2024

Supervisor: PD Dr. Jan Pablo Burgard
1st Reviewer: Prof. Dr. Martin Schmidt
2nd Reviewer: Prof. Dr. Dolores Romero Morales
Date of the disputation: 31.10.2024

Acknowledgements

I would like to express my sincere gratitude to my supervisor Jan Pablo Burgard for his invaluable guidance and insights. His advice, in both my research and personal life, will always be remembered by me. My gratitude extends as well to my second supervisor Martin Schmidt. Learning new optimization concepts and how to improve writing was a very enjoyable experience under his orientation. Pablo and Martin, thank you for your availability and encouragement. I consider myself two-time lucky for working with both of you. It was a pleasure for me.

Moreover, I would like to thank Dolores Romero Morales for being part of my thesis committee, reading my thesis, and giving valuable comments that enhanced the work. My thanks also goes to Volker Schulz for being the chair of my thesis defense.

I thank the DFG Research Training Group 2126 in Algorithmic Optimization (ALOP) for the opportunity to pursue my PhD studies and attend several conferences, which significantly enriched my research experience and professional development. Furthermore, I thank my coworkers for the positive discussions and work environment.

I am grateful to my husband João Vitor for always being by my side and encouraging my dreams. João, thank you so much for your kind and love every step of the way, I am so happy for sharing my life with you.

Thanks also goes to my cousin Luiz Rafael for always believing in me and my potential as a mathematician, even when I didn't see it myself. Rafa, thanks for all your support.

Finalmente, agradeço minha mãe Marina por me proporcionar uma boa formação e por me apoiar em cada etapa da minha vida. Muito obrigada por todo zelo, amor e carinho.

Curriculum Vitae

Personal Information

Name	Maria Eduarda Pinheiro
Nationality	Brazilian
Date of Birth	July 18, 1997
Place of Birth	Blumenau, Santa Catarina, Brazil

Education

04/2022 – 11/2024	Research Assistant in the RTG ALOP Speaker: Prof. Dr. Volker Schulz Universität Trier Trier, Rheinland-Pfalz, Germany
02/2022 03/2020 – 02/2022	Master Degree in Mathematics Universidade Federal do Paraná Curitiba, Paraná, Brazil
11/2020 08/2016 – 12/2019	Licentiate Degree in Mathematics Universidade Federal de Santa Catarina Blumenau, Santa Catarina, Brazil
12/2014	High School Colégio Sagrada Família Blumenau, Santa Catarina, Brazil

Abstract

In machine learning, classification is the task of predicting a label for each point within a data set. When the class of each point in the labeled subset is already known, this information is used to recognize patterns and make predictions about the points in the remainder of the set, referred to as the unlabeled set. This scenario falls in the field of supervised learning.

However, the number of labeled points can be restricted, because, e.g., it is expensive to obtain this information. Besides, this subset may be biased, such as in the case of self-selection in a survey. Consequently, the classification performance for unlabeled points may be limited. To improve the reliability of the results, semi-supervised learning tackles the setting of labeled and unlabeled data. Moreover, in many cases, additional information about the size of each class can be available from undisclosed sources.

This cumulative thesis presents different studies to combine this external cardinality constraint information within three important algorithms for binary classification in the supervised context: support vector machines (SVM), classification trees, and random forests. From a mathematical point of view, we focus on mixed-integer programming (MIP) models for semi-supervised approaches that consider a cardinality constraint for each class for each algorithm.

Furthermore, since the proposed MIP models are computationally challenging, we also present techniques that simplify the process of solving these problems. In the SVM setting, we introduce a re-clustering method and further computational techniques to reduce the computational cost. In the context of classification trees, we provide correct values for certain bounds that play a crucial role for the solver performance. For the random forest model, we develop preprocessing techniques and an intuitive branching rule to reduce the solution time. For all three methods, our numerical results show that our approaches have better statistical performances for biased samples than the standard approach.

Author’s Contribution

This thesis is based on one published paper and two more submitted papers that are currently under review. In Part I of this thesis we present an extended summary of the main motivation and contributions of each work. Part II consists of the reprints of all papers. The contribution of the author of this thesis to each paper is clearly described in the following.

[MEP1] J.P. Burgard, M.E. Pinheiro, and M. Schmidt (2024) “[Mixed-Integer Quadratic Optimization and Iterative Clustering Techniques for Semi-Supervised Support Vector Machines](#)”. In: *TOP-Transactions in Operations Research (2024)*, DOI: [10.1007/s11750-024-00668-w](https://doi.org/10.1007/s11750-024-00668-w).

The main motivation for the paper came from Jan Pablo Burgard. The approaches that are presented in the paper resulted from a joint discussion between all authors. The author of this thesis was responsible for proving all theoretical results and for the implementation of the algorithms under the supervision of the other authors. In addition, she contributed to the writing of the paper.

[MEP2] J.P. Burgard, M.E. Pinheiro, and M. Schmidt (2024) “[Mixed-Integer Linear Optimization for Semi-Supervised Optimal Classification Trees](#)”. Preprint 2024. Under review. URL: <https://arxiv.org/abs/2401.09848>. Upload Date: 18 Jan 2024.

The author of this thesis contributed many ideas and all proofs of theoretical results in this paper. The model presented in the paper was developed through a collaborative discussion between all authors. She was responsible for the main writing of the paper and for the implementation of the algorithms under the supervision of the other authors.

[MEP3] J.P. Burgard, M.E. Pinheiro, and M. Schmidt (2024) “[Mixed-Integer Linear Optimization for Cardinality-Constrained Random Forests](#)”. Preprint 2024. Under review. URL: <https://arxiv.org/abs/2405.09832>. Upload Date: 16 May 2024.

The author of this thesis developed the model, techniques, and proofs of all theoretical results presented in this paper, based on discussions with the other authors. She was responsible for the implementation and the writing of the paper was primarily done by her.

Contents

I	Extended Summary	1
1	Introduction	2
1.1	Machine Learning and Optimization	2
1.2	Contributions and Organization	4
2	Semi-Supervised Support Vector Machines	6
2.1	Support Vector Machines	6
2.2	Cardinality-Constrained Semi-Supervised SVM	9
2.3	Techniques to Solve the CS ³ VM Problem	11
2.4	Numerical Results	14
3	Optimal Classification Trees	18
3.1	Fundamental Concepts	19
3.2	Reformulation and Semi-Supervised Setting	22
3.3	Computational Study	24
4	Random Forests	27
4.1	Cardinality-Constrained Random Forests	27
4.2	Techniques to Solve the C ² RF Problem	30
4.3	Numerical Experiments	32
5	Conclusions and Outlook	36
	Bibliography	38
II	Reprints of the Scientific Papers	43
	Mixed-Integer Quadratic Optimization and Iterative Clustering Techniques for Semi-Supervised Support Vector Machines	44
	Mixed-Integer Linear Optimization for Semi-Supervised Optimal Classification Trees	83
	Mixed-Integer Linear Optimization for Cardinality-Constrained Random Forests	106

Part I

Extended Summary

Chapter 1

Introduction

1.1 Machine Learning and Optimization

Machine learning has become increasingly important for decision-making in many real-world problems. It combines concepts of computing and mathematics in models and algorithms to detect patterns in a given data set and to make predictions such as classifications. For a recent survey of basic concepts and classic methods in machine learning see Burkov (2019) and Zhou (2021). One way to set up a machine learning approach is to build an optimization model to find the best parameters that, e.g., minimize a loss function. With the appropriate model, optimization algorithms are then used to solve the optimization problem; some examples are given in Sun et al. (2020).

In recent years, due to the advancement of algorithms for mixed-integer programming (MIP), many strategies for solving machine learning models by globally solving an optimization problem using MIP techniques have been proposed. In the context of decision trees, Bertsimas and Dunn (2017) were the first to propose MIP approaches. They present two mixed-integer linear programming (MILP) models based on univariate and multivariate trees. Verwer and Zhang (2019) propose a binary linear formulation in which the problem size is largely independent of the size of the training data.

For support vector machines, Blanco et al. (2022) presented a mixed-integer-linear model that considers the labels' noise. Regarding clustering, Burgard et al. (2023) proposed several tailored mixed-integer programming techniques to improve the performance of solvers when applied to the minimum sum-of-squares clustering (MSSC). In the tree ensemble setting, Carrizosa et al. (2023) present a mixed-integer linear optimization formulation that take into account explainability and fairness.

One of the problems that machine learning can be used for is classification. In the context of supervised learning, given a labeled data set, a machine

learning model is trained to classify new points for which the classes are previously unknown. This data set can describe diverse information such as demographics, health of a population, or survey responses. The labels assign the data to different groups, where points within each group share the same characteristics of interest. In this thesis, we consider three of the most famous methods for supervised classification. The first one are support vector machines (SVMs) (Boser et al. 1992; Cortes and Vapnik 1995), in which the goal is to find a separating hyperplane that splits the feature space. The second one are decision trees (Breiman et al. 1984; Quinlan 1986). In this approach, the feature space is recursively partitioned and a prediction is attributed to each resulting part. The last method considered are random forests (Breiman 2001), which combine the predictions of several decision trees to get a more concise result.

Oftentimes, acquiring labels can be expensive, and, therefore, the number of labeled data points is often limited when compared to the number of unlabeled points. This happens, e.g., if one has to do a classic survey to obtain the labels. In these cases, training the machine learning algorithm on partly labeled and partly unlabeled data can be favorable. This leads to a semi-supervised learning setting (Zhu and Goldberg 2009). In the context of neural networks, semi-supervised algorithms have already been proposed, such as the ones presented by Lee (2013), Oliver et al. (2018), and Nguyen et al. (2023). Examples of semi-supervised approaches for logistic regression can be found in Amini and Gallinari (2002) and Bzdok et al. (2015). Furthermore, in many cases, the aggregated information about the number of points in each class within a population is known from an external source. For example, an online retailer may observe the total number of good customer reviews but does not have access to individual ratings due to anonymity practices. Another example is a supermarket, for which the information about how many people pay in cash is known, but this information cannot be retrospectively assigned to each individual. Moreover, in healthcare scenarios, it is possible to know how many patients have a disease but, due to data privacy reasons, one does not know which specific person is affected or not. For logistic regression, Burgard et al. (2021) developed a cardinality-constrained multinomial logit model, accounting for the extra information. This external information can be highly beneficial when the labeled data is biased, i.e., it does not represent the population very well. Biased samples frequently happen in non-probability surveys, which are surveys where the inclusion process is not monitored. For instance, in online surveys with autofill enabled, people can accept pre-filled answers, leading to biased answers.

1.2 Contributions and Organization

In this cumulative thesis, we propose considering the external information on the overall number of points in each class for the three approaches discussed in the previous section. The first contribution is in the SVM setting. We present a mixed-integer quadratic optimization (MIQP) model that covers labeled and unlabeled data points as well as the cardinality constraint on the predicted labels for the unlabeled data. This approach outperforms the standard SVM model in terms of accuracy for biased samples. We also introduce a re-clustering method and derive further computational techniques to reduce the computational cost of the MIQP formulation, leading to similar accuracy and precision but at a much lower computational cost.

The second contribution is the development of a big- M -based mixed-integer linear programming (MILP) model to compute semi-supervised optimal classification trees. To achieve this, we employ SOS techniques and determine the correct values for certain required bounds. The proposed model considers the setting of labeled and unlabeled data points and the additional aggregated information about the unlabeled data for binary classification. When applied to biased samples, the proposed method has better statistical performance than an existing approach from the literature.

The third contribution is about random forests. We introduce an MILP model to solve a random forest problem that deals with the cardinality constraint for the unlabeled data and show that our approach is more accurate than the random forests by majority vote, in particular for biased samples. Moreover, we present some preprocessing techniques and an intuitive branching rule to reduce the computation time. For all the approaches proposed in this thesis, numerical studies for real-world data sets have been conducted to show the performance and advantages of the proposed methods. These numerical experiments primarily focus on scenarios with few and biased labeled points.

This extended summary is organized as follows. In Chapter 2, we introduce the SVM formulation, describe our model to deal with the cardinality constraint in this setting, as well as techniques to solve this problem. In Chapter 3, we focus on decision trees. Specifically, we present some fundamental concepts and our approach for semi-supervised classification trees. This approach considers the number of points in each class for binary classification. Afterward, in Chapter 4, we discuss the random forest setting. Both, the standard approach and our model that takes into account the cardinality constraint are described. There, we also present some problem-tailored preprocessing techniques and a branching rule to solve the model. We conclude the first part of this thesis in Chapter 5.

Finally, in Part [II](#) we present the reprints of the three papers.

Chapter 2

Semi-Supervised Support Vector Machines

Classifying data is a common task in machine learning and support vector machines (SVM) are among the most popular approaches for supervised classification (Boser et al. 1992; Cortes and Vapnik 1995). Given a set of labeled data points with two classes, the SVM methodology aims to split the space of this data into two half-spaces. After that, the SVM classifies each point without a label depending on which half-space it lies in. Nevertheless, in some cases, it is possible to know how many of these points should belong to each class. In [MEP1] we consider incorporating this information into the SVM problem and present approaches to reduce its computational cost.

This chapter provides an overview of [MEP1] and is organized as follows. In Section 2.1 we formally present the SVM problem. Then, in Section 2.2, we introduce semi-supervised SVMs and describe our optimization problem including cardinality constraints in this setting. Afterward, the clustering-based model reduction technique and some algorithmic improvements to solve our problem are presented in Section 2.3. Finally, in Section 2.4, we close this chapter with a summary of the main results and conclusions of [MEP1].

2.1 Support Vector Machines

Let $\Omega = \{(x^1, y_1), \dots, (x^n, y_n)\}$ be a data set with $x^i \in \mathbb{R}^p$ and the label $y_i \in \{-1, 1\}$ for $i \in [1, n] := \{1, \dots, n\}$ ¹. We set $X_l = [x^1, \dots, x^n] \in \mathbb{R}^{p \times n}$ and, for each $i \in [1, n]$, if $y_i = 1$, we say that x^i is in the positive class and if $y_i = -1$, x^i belongs to the negative class.

¹In [MEP1], the dimension p is denoted by d . In this extended summary we unify the notation.

When the points associated with the positive class can be separated from the ones in the negative class by a hyperplane, we have a linearly separable data set. In this case, the goal of the SVM is to find the hyperplane (ω, b) , with $\omega \in \mathbb{R}^p$ and $b \in \mathbb{R}$, that best separates the points in X_l into the positive and negative class.

To illustrate this, consider the 2-dimensional exemplary data set presented in Figure 2.1a, where the red points represent the positive class and the blue points represent the negative class. As can be seen in Figure 2.1b, several hyperplanes separate the two classes. However, the optimal hyperplane is the one that maximizes the distance between itself and the closest point of each class. This distance is called the maximum margin. In our example, the optimal hyperplane is the pink one shown in Figure 2.1c and the maximum margin is presented in Figure 2.1d. In more detail, SVM aims to determine (ω, b) such that

$$\begin{aligned} \omega^\top x^i + b &\geq 1, & \text{if } y_i = 1, \\ \omega^\top x^i + b &\leq -1, & \text{if } y_i = -1, \end{aligned} \tag{2.1}$$

holds for all $i \in [1, n]$ and that maximize the distance between

$$\mathcal{H}^-(\omega, b) := \{x \in \mathbb{R}^p : \omega^\top x + b = -1\}$$

and

$$\mathcal{H}^+(\omega, b) := \{x \in \mathbb{R}^p : \omega^\top x + b = 1\}.$$

As can be seen in Mangasarian (1999), the distance between $\mathcal{H}^-(\omega, b)$ and $\mathcal{H}^+(\omega, b)$ is given by $2/\|\omega\|$, where $\|\cdot\|$ denotes the Euclidean norm. Since maximizing this distance is equivalent to minimizing $\frac{\|\omega\|^2}{2}$, the optimal hyperplane of the SVM problem is given by $\omega^* \in \mathbb{R}^p$ and $b^* \in \mathbb{R}$ that solve the optimization problem

$$\begin{aligned} \min_{\omega, b} \quad & \frac{1}{2} \|\omega\|^2 \\ \text{s.t.} \quad & y_i(\omega^\top x^i + b) \geq 1, \quad i \in [1, n], \\ & \omega \in \mathbb{R}^p, \quad b \in \mathbb{R}. \end{aligned}$$

Nevertheless, in practice, the data is often not linearly separable. In these cases, there exists no separating hyperplane and the problem above is infeasible. To work around this issue, one can use SVM with so-called soft margins, by adding slack variables to allow some misclassification. Therefore, the problem can be stated as

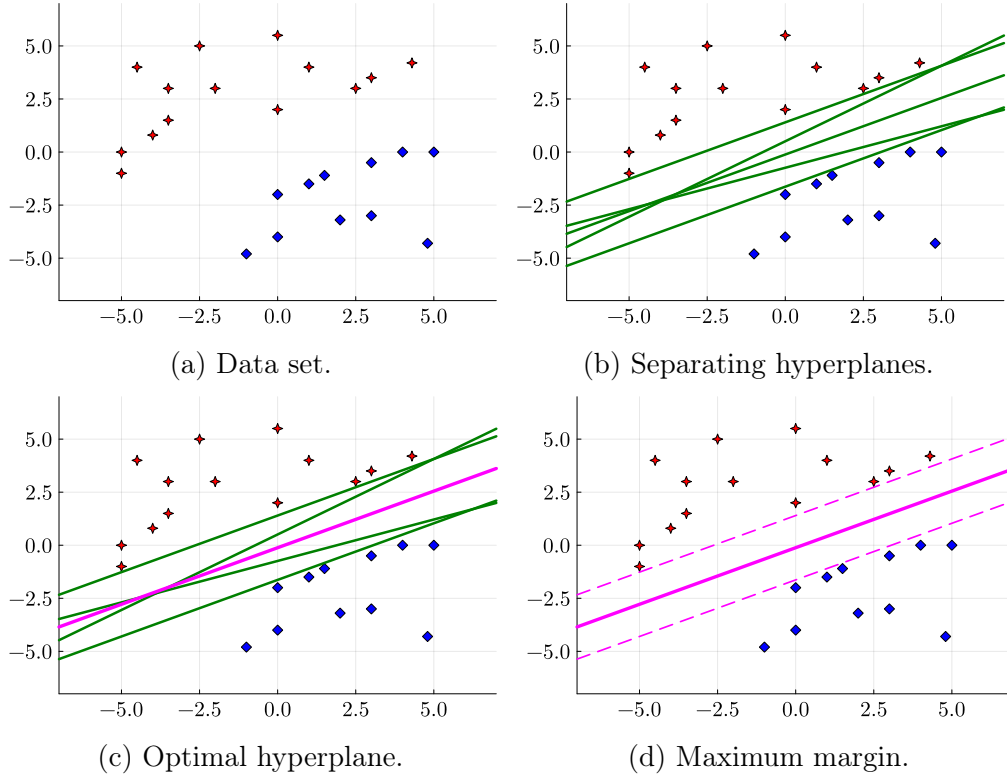


Figure 2.1: A 2-dimensional SVM example.

$$\min_{\omega, b, \xi} \frac{\|\omega\|^2}{2} + C_1 \sum_{i=1}^n \xi_i \quad (\text{P1a})$$

$$\text{s.t. } y_i(\omega^\top x^i + b) \geq 1 - \xi_i, \quad i \in [1, n], \quad (\text{P1b})$$

$$\xi_i \geq 0, \quad i \in [1, n], \quad (\text{P1c})$$

$$\omega \in \mathbb{R}^p, \quad b \in \mathbb{R}. \quad (\text{P1d})$$

Note that the objective function in (P1a) is a compromise between maximizing the distance between the two classes and minimizing the classification error of the labeled data. The penalty parameter $C_1 > 0$ aims to control the importance of ξ . That is, a larger value of C_1 places a higher penalty on misclassifications. Let (ω^*, b^*, ξ^*) be the solution of Problem (P1). Then, for each $i \in [1, n]$ we have

$$\xi_i^* = \begin{cases} [1 - (\omega^*)^\top x^i - b^*]^+, & \text{if } y_i = 1, \\ [1 + (\omega^*)^\top x^i + b^*]^+, & \text{if } y_i = -1, \end{cases} \quad (2.2)$$

with $[v]^+ := \max\{0, v\}$. That is, if for some $i \in [1, n]$, x^i satisfies Expression (2.1), we get $\xi_i^* = 0$. However, if x^i is wrongly classified we obtain $\xi_i^* \geq 1$. When $0 < \xi_i^* < 1$ holds, x^i lies on the correct half-space but does not satisfies $|(\omega^*)^\top x^i + b^*| > 1$. After solving Problem (P1), we can classify an unlabeled point $x \in \mathbb{R}^p$, as positive if $(\omega^*)^\top x + b^* > 0$ holds and negative if it satisfies $(\omega^*)^\top x + b^* < 0$.

2.2 Cardinality-Constrained Semi-Supervised SVM

Observe that in the standard SVM problem (P1), only the labeled data set is considered. Meanwhile, the size of this set may be insufficient and, therefore, the optimal hyperplane provided by the SVM is not well-suited for classifying unlabeled data. This happens mainly when obtaining labels for all units of interest is costly. To overcome this situation, Bennett and Demiriz (1998) formulate a semi-supervised SVM (S³VM) as a mixed-integer linear problem (MILP) that considers labeled and unlabeled data. In the following decades, some researchers have proposed approaches for solving S³VM such as the transductive approach by Joachims (2002) and Xu Yu (2012) or manifold regularization by Belkin et al. (2006) and Melacci and Belkin (2009). Chapelle and Zien (2005) propose a balancing constraint, that is also considered by Chapelle et al. (2006) and Kontonatsios et al. (2017). Nevertheless, this constraint enforces that the unlabeled data has similar proportions per class as the labeled data, but this might not be true. Moreover, in many cases, the information about the number of positive and negative points in a population is known from an external source. Considering $X_u = [x^{n+1}, \dots, x^N]$ as the unlabeled data and let λ be the total number of positive labels in X_u , in [MEP1] we propose to add this aggregated additional information to the SVM formulation by imposing a cardinality constraint on the predicted labels for the unlabeled data². For that, introducing $\eta \in \mathbb{R}^2$ and binary variables $z_i \in \{0, 1\}$, $i \in [n+1, N]$, we present the big- M formulation

$$\min_{\omega, b, \xi, \eta, z} \frac{\|\omega\|^2}{2} + C_1 \sum_{i=1}^n \xi_i + C_2(\eta_1 + \eta_2) \quad (\text{P2a})$$

$$\text{s.t. } y_i(\omega^\top x^i + b) \geq 1 - \xi_i, \quad i \in [1, n], \quad (\text{P2b})$$

$$\omega^\top x^i + b \leq z_i M, \quad i \in [n+1, N], \quad (\text{P2c})$$

²Since we consider this cardinality information in [MEP1], [MEP2], and [MEP3] and their notation differ, in this extended summary we unify the notation, using λ .

$$\omega^\top x^i + b \geq -(1 - z_i)M, \quad i \in [n + 1, N], \quad (\text{P2d})$$

$$\lambda - \eta_1 \leq \sum_{i=n+1}^N z_i \leq \lambda + \eta_2, \quad (\text{P2e})$$

$$\xi_i \geq 0, \quad i \in [1, n], \quad (\text{P2f})$$

$$\eta_1, \eta_2 \geq 0, \quad (\text{P2g})$$

$$z_i \in \{0, 1\}, \quad i \in [n + 1, N], \quad (\text{P2h})$$

where M needs to be chosen sufficiently large. As z_i is binary, Constraints (P2c) and (P2d) lead to

$$\omega^\top x^i + b > 0 \implies z_i = 1, \quad i \in [n + 1, N],$$

$$\omega^\top x^i + b < 0 \implies z_i = 0, \quad i \in [n + 1, N].$$

If x^i lies on the hyperplane, i.e., $\omega^\top x^i + b = 0$, Constraints (P2c) and (P2d) hold for $z_i = 1$ and $z_i = 0$. In this case, x^i can be counted either on the positive or on the negative side. The objective function in (P2a) is very similar to the one in (P1a), the only difference is that it now also minimizes the classification error in the unlabeled data. The penalty parameter $C_2 > 0$ aims to control the importance of the slack variable η . Constraint (P2e) ensures that the number of unlabeled points on the positive side is close to λ as possible. Reformulation (P2) is a mixed-integer quadratic problem (MIQP) in which all constraints are linear and the objective function is quadratic. We refer to this problem as CS³VM.

In the big- M formulation, the choice of M is crucial. If M is too small, the problem can become infeasible or solutions could be cut off. If M is chosen too large, the respective continuous relaxations usually lead to bad lower bounds and solvers may encounter numerical troubles. Considering $m := N - n$, the choice of M is discussed in the following theorem.

Theorem 1 (Theorem 2 in [MEP1]) *A valid big- M for Problem (P2) is given by*

$$M = 2\sqrt{2(2C_1\bar{n} + C_2(m - \lambda))} \max_{i \in [1, N]} \|x^i\| + 1 \quad (2.3)$$

with $\bar{n} := |\{i \in [1, n]: y_i = -1\}|$.

We refer to [MEP1] for the proof of this theorem and a corresponding auxiliary lemma. Now, with Model (P2) at hand, we briefly discuss our approaches for solving it in the next section.

2.3 Techniques to Solve the CS³VM Problem

As already mentioned, CS³VM is an MIQP. Therefore, finding a solution efficiently is challenging. In this extended summary, we present an overview of the techniques proposed in [MEP1] to reduce the computational burden of solving CS³VM.

2.3.1 Clustering

In Model (P2) each binary variable is related to an unlabeled point. The larger the number of unlabeled data, the larger the number of binary variables and, hence, the larger the computational burden to solve Problem (P2). To reduce this computational burden, in [MEP1] we propose to cluster the unlabeled data. Given a number k of clusters and a matrix $S = [s^1, \dots, s^d] \in \mathbb{R}^{p \times d}$ of given points, the goal of the minimum sum-of-squares clustering (MSSC) is to find mean vectors $c^j \in \mathbb{R}^p$, $j \in [1, k]$, that solve the problem

$$c^* = \arg \min_c \ell(S, c), \quad c = (c^j)_{j=1, \dots, k},$$

where the loss function ℓ is the sum of the squared Euclidean distances, i.e.,

$$\ell(S, c) = \sum_{j=1}^k \sum_{s^i \in \mathcal{C}_j} \|s^i - c^j\|^2$$

with $\mathcal{C}_j \subset \mathbb{R}^p$ being the set of data points that are assigned to cluster j .

We solve this problem heuristically using the k -means algorithm (MacQueen 1967; Lloyd 1982) for $S = X_u$, i.e., we cluster the unlabeled data. Then, instead of using all unlabeled data, we only use the clusters' centroids c^1, \dots, c^k , and the numbers e_1, \dots, e_k of data points in each cluster to obtain the following problem

$$\min_{\omega, b, \xi, \eta, z} \frac{\|\omega\|^2}{2} + C_1 \sum_{i=1}^n \xi_i + C_2(\eta_1 + \eta_2) \quad (\text{P3a})$$

$$\text{s.t.} \quad y_i(\omega^\top x^i + b) \geq 1 - \xi_i, \quad i \in [1, n], \quad (\text{P3b})$$

$$\omega^\top c^j + b \leq z_j M, \quad j \in [1, k], \quad (\text{P3c})$$

$$\omega^\top c^j + b \geq -(1 - z_j)M, \quad j \in [1, k], \quad (\text{P3d})$$

$$\lambda - \eta_1 \leq \sum_{j=1}^k e_j z_j \leq \lambda + \eta_2, \quad (\text{P3e})$$

$$\xi_i \geq 0, \quad i \in [1, n], \quad (\text{P3f})$$

$$\eta_1, \eta_2 \geq 0, \tag{P3g}$$

$$z_j \in \{0, 1\}, \quad j \in [1, k]. \tag{P3h}$$

In Proposition 1 of [MEP1] we prove that a valid big- M for Problem (P3) is still given by (2.3). However, the hyperplane given by (ω^*, b^*) that results from the solution of Problem (P3) can cut through some cluster. In this case, not all data points of the cluster lie on the same side of the hyperplane. If this happens, the solution of Problem (P3) might not satisfy the cardinality constraint (P2e) of Problem (P2).

To fix this, we propose an iterative method that is given in Algorithm 1. In Theorem 3 and Theorem 4 of [MEP1] we prove, respectively, that Algorithm 1 terminates after finitely many iterations, and, although the point obtained by Algorithm 1 is not necessarily a solution to Problem (P2), it is a feasible point for Problem (P2).

Algorithm 1: Re-Clustering Method (RCM) (Alg. 1 in [MEP1])

Input : $X \in \mathbb{R}^{d \times N}$, $y \in \{-1, 1\}^n$, $k^1 \in \mathbb{N}$, $C_1 > 0$, $C_2 > 0$, and $\lambda \in \mathbb{N}$.

- 1 Set $t \leftarrow 1$, compute M^t as in (2.3), compute a clustering of X_u in k^1 many clusters using the k -means algorithm, and obtain the centroids c^1, \dots, c^{k^1} as well as the numbers e_1, \dots, e_{k^1} of data points in each cluster.
- 2 Solve Problem (P3) to compute the hyperplane (ω^t, b^t) as well as ξ^t, η^t, z^t .
- 3 **if** the hyperplane (ω^t, b^t) cuts a cluster **then**
- 4 Set $k^{t+1} \leftarrow k^t$.
- 5 **for** each cluster that is cut by the hyperplane (ω^t, b^t) **do**
- 6 Split the cluster into two new clusters so that neither of the two new clusters is cut by the hyperplane (ω^t, b^t) .
- 7 Update the centroids of the newly created clusters.
- 8 Set $k^{t+1} \leftarrow k^{t+1} + 1$.
- 9 **end**
- 10 Update $t \leftarrow t + 1$ and go to Step 2.
- 11 **else**
- 12 Return the hyperplane (ω^t, b^t) as well as ξ^t, η^t, z^t .
- 13 **end**

2.3.2 Further Algorithmic Enhancements

Observe that in Algorithm 1, the number of clusters increases in each iteration. Consequently, the computational cost of solving Problem (P3) typically

increases with each iteration. From a geometric point of view, as in the original SVM problem, points closer to the hyperplane have a greater influence on the resulting hyperplane than the other points. Hence, eliminating points that are far from the hyperplane decreases the size of the problem without significantly affecting the solution. In [MEP1], the idea proposed is based on the following thought. The farther a cluster is from the hyperplane in an iteration, the less likely it is that the cluster will be split or change sides completely in a future iteration. Hence, the clusters farthest from the current hyperplane mainly add information about their side and capacity. However, in a later iteration, the cluster may become relevant again.

To discard detailed information on certain clusters but also to reactivate the discarded clusters when necessary, in [MEP1] we propose the following. If the number of clusters exceeds a threshold value, we first fix the cluster with the centroid farthest from the hyperplane as a kind of residual cluster on a side if this side has points far from the hyperplane. Then, we discard all clusters in which all points are farther from the hyperplane than some given value and assign them to the residual cluster on their side of the hyperplane. This way the cardinality constraint remains valid. Moreover, all formerly discarded clusters are checked for re-consideration. This procedure is added to Algorithm 1 and the details can be seen in Section 4.2 of [MEP1].

Moreover, as discussed in Section 2.2, M in the presented formulation needs to be sufficiently large. However, the bigger the M , the more likely we face numerical issues. As shown in Theorem 1, feasible points with lower objective function values allow us to choose a smaller value for M . Based on that, we update M in each iteration of Algorithm 1 intending to decrease M . For more details, we refer to Theorem 5 of [MEP1].

Motivated by the above discussions, we add new steps in Algorithm 1, which can be seen in Algorithm 2 of [MEP1]. We refer to this Algorithm as the Improved Re-Clustering Method (IRCM). In Theorem 6 and Theorem 7 of [MEP1], we show, respectively, that as Algorithm 1, IRCM always terminates after finitely many iterations, and the point obtained by it is feasible for Problem (P2). Hence, we can use this point for warm-starting the solution process of Problem (P2).

In [MEP1] we also propose an approach to fix some unlabeled points in one side of the hyperplane. This approach and the warm-start technique give rise to a new algorithm called Improved & Warm-Started Re-Clustering Method (WIRCM) and its description and all details can be seen in Section 5 of [MEP1]. It is important to point out that, different from IRCM, WIRCM always finds the optimal solution of CS³VM.

2.4 Numerical Results

In Section 6 of [MEP1], we perform a detailed computational study and present the results that illustrate the benefits of knowing the total number of data points in each class of unlabeled data in the SVM setting. Our study focuses primarily on the case of non-representative biased samples. We also present the advantages of using our approaches to speed up the solution process. To do so, we consider different test sets from the literature and for each one, we create 5 biased samples with 10 percent of the data being labeled. Then, we compare the standard SVM, as described in Problem (P1), where only labeled data is considered, and our proposed methods CS³VM, IRCM, and WIRCM. In this extended summary, we briefly discuss the main results and conclusions of our simulations. For all the details of the numerical results, we refer to [MEP1].

Our first observation from the computational analysis is the empirical cumulative distribution functions (ECDFs) for the run time. Specifically, for S being a set of solvers (or methods) and for P being a set of problems, we denote by $t_{\bar{p},s} \geq 0$ the run time of approach $s \in S$ applied to the problem $\bar{p} \in P$ in seconds. If $t_{\bar{p},s} > 3600$, we consider problem \bar{p} as not being solved by approach s . With these notations, the performance profile of approach s is the graph of the function $\gamma_s : [0, \infty) \rightarrow [0, 1]$ given by

$$\gamma_s(\sigma) = \frac{1}{|P|} |\{\bar{p} \in P : t_{\bar{p},s} \leq \sigma\}|. \quad (2.4)$$

The ECDFs for the run time can be seen in Figure 2.2. As expected, SVM is the fastest algorithm. This is the case because SVM does not include any binary variables related to the unlabeled points, which is in contrast to the alternative approaches. IRCM comes second, which shows that the idea of clustering unlabeled data points significantly decreases the run time. Nevertheless, the termination of SVM and IRCM does not imply that a globally optimal point is found, whereas this is guaranteed by CS³VM and the WIRCM. The quality of the points found by SVM and IRCM will be discussed in the next paragraph. From Figure 2.2 we can also conclude that Problem (P2) is rather challenging. Even IRCM, which terminates for the most instances within the time limit of 1 h (indicated by the gray and dashed vertical line) only solves 57% of the instances.

In [MEP1] we show that IRCM and SVM provide a feasible point for CS³VM. Hence, we compare how close the objective function values obtained from the approaches are to the optimal solution. To this end, we use ECDFs, for which we replace $t_{\bar{p},s}$ by $f_{\bar{p},s}$ in Equation (2.4) with

$$f_{\bar{p},s} := \frac{b_{\bar{p},s} - f_{\bar{p}}^*}{f_{\bar{p}}^*},$$

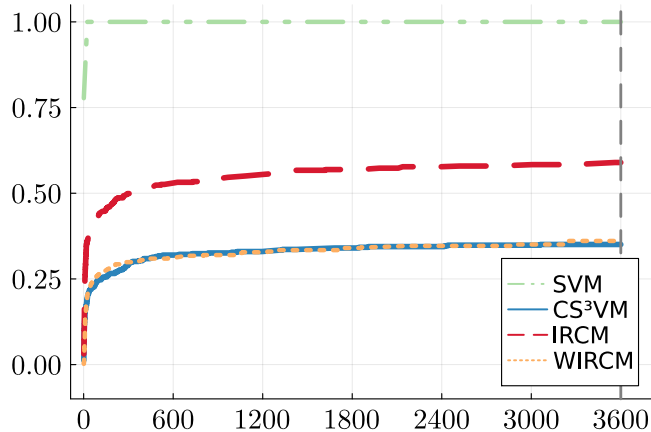


Figure 2.2: ECDFs for run time (in seconds).

where $f_{\bar{p}}^*$ is the optimal objective function value of problem \bar{p} and $b_{\bar{p},s}$ is the objective function value obtained by approach s . Figure 2.3 shows the ECDFs for the upper bound quality. The objective function value obtained by SVM is very far from the optimal value, while the IRCM finds an objective function value rather close to the optimal value (with $f_{p,s} \leq 0.2$, see the gray dashed vertical line) in 90 % of these instances. Moreover, the WIRCM outperforms CS³VM in this comparison, which means that using the IRCM as a warm start improves the result.

So far, we can conclude the following. If one is interested in getting a rather good feasible point as quickly as possible, one should use the IRCM. If one is able to spend some more run time, one should use the WIRCM. Hence, both novel methods derived in [MEP1] have their advantage over just solving the CS³VM with a standard MIQP solver.

Knowing the true label of all points, we then distinguish all points in four categories: true positive (TP) or true negative (TN) if the point is classified correctly in the positive or negative class, respectively, as well as false positive (FP) if the point is misclassified in the positive class and as false negative (FN) if the point is misclassified in the negative class. Based on that, in [MEP1] we compute some classification metrics. In this extended summary, we present the accuracy, (AC), which measures the proportion of correctly classified points and is given by

$$\text{AC} := \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \in [0, 1].$$

The main comparison in terms of accuracy is w.r.t. the “true hyperplane”, i.e., the solution of Problem (P1) on the complete data with all labels available. The main question is how close the accuracy is to the one of the true hyper-

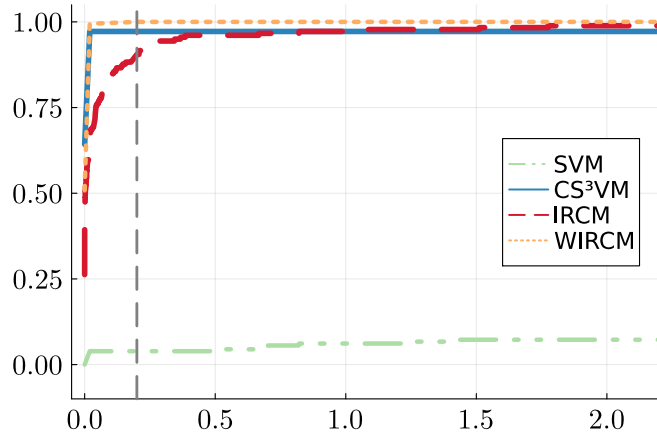
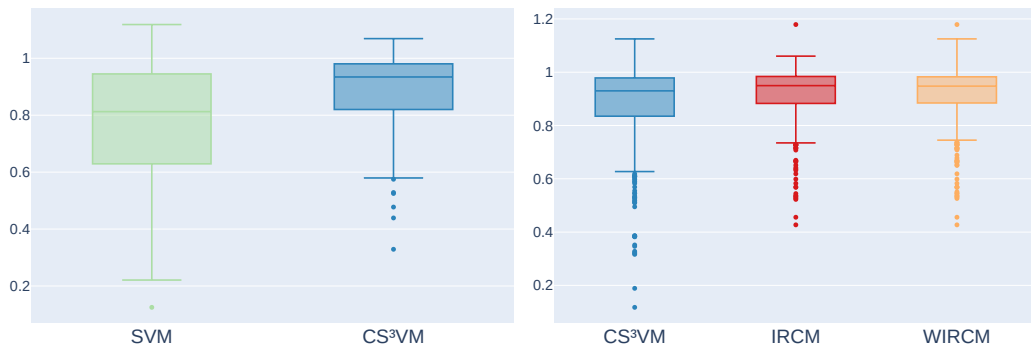


Figure 2.3: ECDFs for the quality of the obtained upper bounds.

plane. Hence, in Figure 2.4 we present the ratios of the accuracy between the approaches mentioned above and the true hyperplane. The box in the boxplot depicts the range of the medium 50 % of the values; 25 % of the values are below and 25 % are above the box.

In Figure 2.4a, we only consider instances for which CS^3VM terminates within the time limit of 1 h. As can be seen, the relative accuracy w.r.t. the true hyperplane of CS^3VM , is closer to 1 than the relative accuracy of SVM. This means that using the unlabeled points as well as the cardinality constraint allows to re-produce the classification of the true hyperplane with higher accuracy than the standard SVM does. In Figure 2.4b we consider only those three approaches that actually consider the unlabeled data for all instances. It shows that, even though IRCM does not have an optimality guarantee, it has a better relative accuracy than the hyperplane obtained from CS^3VM within the time limit. Consequently, as the hyperplane obtained from IRCM is used as a warm-start in WIRCM, the WIRCM also has better accuracy than the one obtained from CS^3VM .

For more details of accuracy performance and other comparisons such as precision and false-positive rate, we refer to Section 6 of [MEP1]. There, we also present numerical results for simple random samples, where, different from biased samples, each unit in the data set has the same probability to be included in the sample of labeled data. We see that, in this setting, the results from the proposed methods are similar to the classic SVM approach.



(a) Instances for which CS³VM terminated.

(b) All instances.

Figure 2.4: Relative accuracy w.r.t. the true hyperplane. Comparison for all data points.

Chapter 3

Optimal Classification Trees

Depending on the data set, splitting the data space into just two half-spaces, as SVMs do, may not yield an effective prediction. In these cases, another classification approach, such as decision trees can be employed. Consider $X_l = [x^1, \dots, x^n] \in \mathbb{R}^{p \times n}$ being labeled data and $c \in \{\mathcal{A}, \mathcal{B}\}^n$ as the vector of class labels for the labeled data. The core idea of decision trees is to recursively partition the feature space, according to branching rules, and assign a prediction to each resulting part of the partition. After that, given an unlabeled data set $X_u = [x^{n+1}, \dots, x^N] \in \mathbb{R}^{p \times m}$, each $x^i \in X_u$ is classified either as \mathcal{A} or \mathcal{B} , depending on the partition it lies in.

In general, the partitioning is done by hyperplanes. If these hyperplanes involve a single feature we have a univariate tree and some approaches can be seen in Yildiz and Dikmen (2007) and Kotsiantis (2014). In a multivariate tree, each hyperplane is created using more than one feature, see, e.g., Orsenigo and Vercellis (2003) and Altincay (2007). It is well-known that univariate trees are often simpler than multivariate trees (Breiman et al. 1984). However, multivariate trees give better results than univariate trees (Brodley and Utgoff 1995). Moreover, the significant progress made in mixed-integer programming in recent years has led to many approaches for computing optimal classification trees (OCT) by globally solving an optimization problem. The recent surveys by Carrizosa et al. (2021) and Gambella et al. (2021) discuss some strategies. The first techniques were proposed by Bertsimas and Dunn (2017). They present two mixed-integer linear programming (MILP) models based on univariate and multivariate trees.

In [MEP2] we consider a multivariate tree in the semi-supervised setting. Specifically, we propose an MILP model for classification trees that considers the cardinality constraint of each class in a binary classification scenario. This chapter presents an overview of [MEP2] and is organized as follows. In Section 3.1 we introduce some preliminary concepts, based on Bennett and Blue (1996) and Bertsimas and Dunn (2017). In Section 3.2 we present our

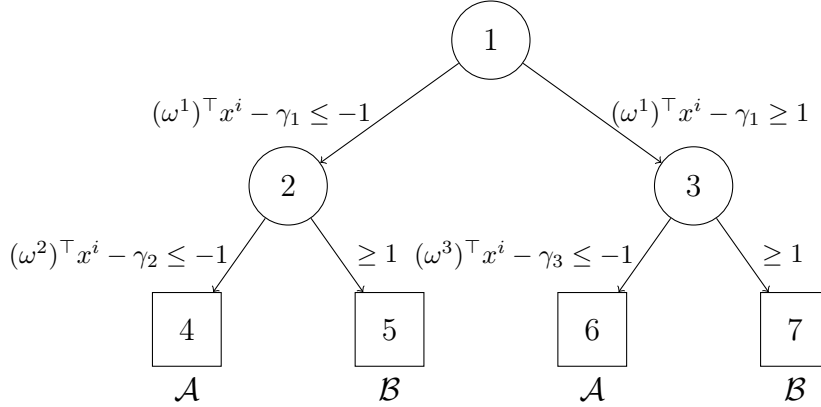


Figure 3.1: A classification tree with depth $D = 2$.

optimization model for computing a semi-supervised optimal classification tree. Numerical results and conclusion of [MEP2] are presented in Section 3.3. All the details and results can be found in [MEP2].

3.1 Fundamental Concepts

Given a depth $D \in \mathbb{N}$, a classification tree has $2^{D+1} - 1$ nodes, divided into two types; the branch nodes $\mathcal{T}_B = [1, 2^D - 1]$ and the leaf nodes $\mathcal{T}_L = [2^D, 2^{D+1} - 1]$. Every branch node $b \in \mathcal{T}_B$ provides a hyperplane characterized by (ω^b, γ_b) , with $\omega^b \in \mathbb{R}^p$ and $\gamma_b \in \mathbb{R}$, that splits the feature space into half-spaces. If a point x^i , $i \in [1, N]$, satisfies $(\omega^b)^\top x^i - \gamma_b \leq -1$, then x^i follows the left branch of the node b . If $(\omega^b)^\top x^i - \gamma_b \geq 1$ holds, the point x^i follows the right branch of the node b . In each leaf node, $t \in \mathcal{T}_L$, all points x^i , $i \in [1, N]$, are classified as \mathcal{A} or \mathcal{B} . In a simple example with $D = 2$, the classification tree has $\mathcal{T}_B = [1, 3]$ and $\mathcal{T}_L = [4, 7]$; see Figure 3.1. Concerning the classification at the leaf nodes, define $\mathcal{T}_L^{\mathcal{A}} = \{t \in \mathcal{T}_L : t \text{ is even}\}$ as the set of leaf nodes that are classified as \mathcal{A} and $\mathcal{T}_L^{\mathcal{B}} = \{t \in \mathcal{T}_L : t \text{ is odd}\}$ as the set of leaf nodes that are classified as \mathcal{B} . In the case with $D = 2$, see Figure 3.1 again, $\mathcal{T}_L^{\mathcal{A}} = \{4, 6\}$ and $\mathcal{T}_L^{\mathcal{B}} = \{5, 7\}$ holds.

For each leaf node $t \in \mathcal{T}_L$, let $\mathcal{N}_R(t)$ be the index set of the branch nodes in which the right or “greater than” branch is traversed to reach leaf t . Moreover, let $\mathcal{N}_L(t)$ be the index set of the branch nodes in which the left or “less than” branch is traversed to reach leaf t . For our running example of the classification tree in Figure 3.1 we have

$$\begin{aligned} \mathcal{N}_L(4) &= \{1, 2\}, & \mathcal{N}_R(4) &= \emptyset, & \mathcal{N}_L(5) &= \{1\}, & \mathcal{N}_R(5) &= \{2\}, \\ \mathcal{N}_L(6) &= \{3\}, & \mathcal{N}_R(6) &= \{1\}, & \mathcal{N}_L(7) &= \emptyset, & \mathcal{N}_R(7) &= \{1, 3\}. \end{aligned}$$

With the fixed depth D , a labeled point x^i , $i \in [1, n]$, with $c_i \in \{\mathcal{A}, \mathcal{B}\}$, is considered as correctly classified if all inequalities from the root to the leaf node are satisfied for some leaf node $t \in \mathcal{T}_L^{c_i}$. That is, if

$$\bigvee_{t \in \mathcal{T}_L^{c_i}} \left\{ \left\{ \bigwedge_{b \in \mathcal{N}_R(t)} \left[(\omega^b)^\top x^i - \gamma_b \geq 1 \right] \right\} \wedge \left\{ \bigwedge_{b \in \mathcal{N}_L(t)} \left[(\omega^b)^\top x^i - \gamma_b \leq -1 \right] \right\} \right\} \quad (3.1)$$

holds.

In the example with $D = 2$ present in Figure 3.1, a point x^i with $c_i = \mathcal{A}$ is correctly classified if

$$\begin{aligned} & \left\{ \left[(\omega^1)^\top x^i - \gamma_1 \leq -1 \right] \wedge \left[(\omega^2)^\top x^i - \gamma_2 \leq -1 \right] \right\} \\ & \vee \left\{ \left[(\omega^1)^\top x^i - \gamma_1 \geq 1 \right] \wedge \left[(\omega^3)^\top x^i - \gamma_3 \leq -1 \right] \right\} \end{aligned}$$

is satisfied. To visualize these concepts in more detail, consider the 2-dimensional example presented in Figure 3.2a. Here, the blue points belong to class \mathcal{A} and the red points are in class \mathcal{B} . Considering $D = 2$, the classification tree partitions the data space into four parts using three hyperplanes, as can be seen in Figure 3.2b. Observe that all labeled points are correctly classified.

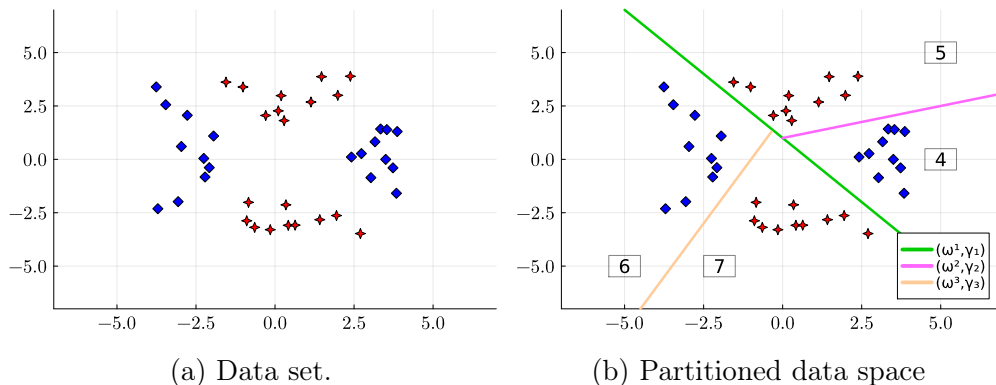


Figure 3.2: A 2-dimensional example of a classification tree.

In the context of supervised classification using decision trees, Bennett and Blue (1996) propose to find the hyperplanes (ω^b, γ_b) , $b \in [1, 2^D - 1]$, such that Expression (3.1) is satisfied for as many labeled points x^i as possible. To do so, they define some suitable error measures that then have to be minimized. In [MEP2], we define the branch and leaf error according to the decision error and leaf error proposed by Bennett and Blue (1996). The first error is related to branch nodes. For each labeled point, at each branch node, we consider the inequality that must be satisfied and then measure by how much it is violated.

Definition 1 (Branch Error) Given a labeled point x^i , $i \in [1, n]$, in any branch node $b \in \mathcal{T}_B$, the branch errors $(y_b^R)_i$ and $(y_b^L)_i$ are defined by

$$(y_b^R)_i := \left[-(\omega^b)^\top x^i + \gamma_b + 1 \right]^+ \quad \text{and} \quad (y_b^L)_i := \left[(\omega^b)^\top x^i - \gamma_b + 1 \right]^+$$

with $[v]^+ := \max\{0, v\}$.

The interpretation of the definition above is quite similar to the role of ξ in the SVM setting that was discussed in Section 2.1, more specifically in Expression (2.2). It can be understood as follows. If the point x^i follows the right branch in some node $b \in \mathcal{T}_B$, i.e., if $(\omega^b)^\top x^i - \gamma_b \geq 1$ holds, we obtain $(y_b^R)_i = 0$ and $(y_b^L)_i > 0$. However, if the point x^i satisfies $(\omega^b)^\top x^i - \gamma_b \leq -1$, and therefore follows the left branch in some node $b \in \mathcal{T}_B$, it holds $(y_b^L)_i = 0$ and $(y_b^R)_i > 0$.

The second definition represents the error in each leaf node t . For each labeled point, we sum over the branch errors along the path from the root to the leaf node t .

Definition 2 (Leaf Error) The leaf error of a labeled point x^i , $i \in [1, n]$, at a leaf node $t \in \mathcal{T}_L$ is defined by

$$LE(x^i, t) := \sum_{b \in \mathcal{N}_R(t)} (y_b^R)_i + \sum_{b \in \mathcal{N}_L(t)} (y_b^L)_i. \quad (3.2)$$

Observe that $LE(x^i, t)$ is a linear expression. As explained in Bennett and Blue (1996), each labeled point x^i is correctly classified if the minimum value of all leaf errors is zero, i.e., if

$$\min_{t \in \mathcal{C}_i} \{LE(x^i, t)\} = 0$$

is satisfied. Based on that, Bennett and Blue (1996) propose the following approach for computing a supervised classification tree: Find the parameters $\omega \in \mathbb{R}^{p \times 2^D - 1}$, $\gamma \in \mathbb{R}^{2^D - 1}$ as well as $y^R, y^L \in \mathbb{R}^{2^D - 1 \times n}$ that solve the optimization problem

$$\min_{\omega, \gamma, y^R, y^L} \sum_{i=1}^n \min_{t \in \mathcal{T}_L^i} \{LE(x^i, t)\} \quad (\text{P3a})$$

$$\text{s.t.} \quad (y_b^R)_i \geq -(\omega^b)^\top x^i + \gamma_b + 1, \quad b \in \mathcal{T}_B, \quad i \in [1, n], \quad (\text{P3b})$$

$$(y_b^L)_i \geq (\omega^b)^\top x^i - \gamma_b + 1, \quad b \in \mathcal{T}_B, \quad i \in [1, n], \quad (\text{P3c})$$

$$(y_b^R)_i, (y_b^L)_i \geq 0, \quad b \in \mathcal{T}_B, \quad i \in [1, n]. \quad (\text{P3d})$$

Note that the objective function in (P3a) minimizes the classification error on the labeled data. Constraints (P3b) and (P3c) enforce on which branch (left or right) the labeled point x^i should traverse in branch node b . The function $\min\{\text{LE}(x^i, t): t \in \mathcal{T}_L^{c_i}\}$ in the objective function (P3a) is discontinuous, which means that Problem (P3) cannot be solved easily by standard algorithms. Bennett and Blue (1996) present an extreme point tabu search to find a feasible point for Problem (P3) that correctly classifies the same number of labeled points as the solution to Problem (P3). However, they do not necessarily find the solution to Problem (P3). To overcome this issue, in [MEP2] we present a mixed-integer linear programming (MILP) formulation using binary variables to reformulate the objective function (P3a), which is sketched in the next section.

3.2 Reformulation and Semi-Supervised Setting

In [MEP2] we use classic SOS1-techniques (Beale and Tomlin 1969) and McCormick inequalities (McCormick 1976) to re-phrase the min-min problem present on the objective function (P3a). For more details, we refer to Lemma 1 and Proposition 1 in [MEP2]. To achieve this, some bounds need to be derived and for that, we define the domain of each variable ω^b in Problem (P3) as follows

$$-s \leq \omega_j^b \leq s, \quad b \in \mathcal{T}_B, \quad j \in [1, p], \quad (3.3)$$

for some $s > 0$. Note that the bounds imposed on each entrance of ω^b (which determine the hyperplane's inclination) also serves as a regularization, limiting the overall size of the hyperplane. Besides that, Problem (P3) only considers the labeled points. However, as already mentioned, the number of these points can be limited. Thus, it would be beneficial to train the classification tree with labeled and unlabeled data, leading to a semi-supervised decision tree field. In this context, Kocev et al. (2017) consider minimizing a local impurity function and Tanha et al. (2017) present self-training as base learners. Recently, Santhiappan and Ravindran (2021) consider a maximum mean discrepancy to estimate the class ratio at every splitting rule in a univariate decision tree. Hence, although these approaches present various ideas, none of them considers globally solving a single optimization problem as we do in [MEP2].

Moreover, similar as in Section 2.2 for the SVM problem, consider λ to be the total number of unlabeled points that belong to class \mathcal{A} . In [MEP2] we propose to add this aggregated additional information as well as the unlabeled information into Problem (P3). To sum up, by introducing the parameter $\xi \geq 0$ and the matrices $\beta \in \mathbb{R}^{n \times 2^{D-1}}$, $\alpha \in \{0, 1\}^{n \times 2^{D-1}}$, $z \in \{0, 1\}^{m \times 2^D - 1}$, and

$\delta \in \{0, 1\}^{m \times 2^{D-1}}$ of binary variables, in [MEP2] we propose the optimization problem

$$\begin{aligned}
\min_{\omega, \gamma, y^R, y^L, \xi, \alpha, \beta, z, \delta} \quad & \sum_{i=1}^n \sum_{t \in \mathcal{T}_L^{c_i}} \beta_t^i + C\xi & (\text{P4a}) \\
\text{s.t.} \quad & (\text{P3b})\text{--}(\text{P3d}), (3.3), & (\text{P4b}) \\
& \sum_{t \in \mathcal{T}_L^{c_i}} \alpha_t^i = 1, \quad i \in [1, n], & (\text{P4c}) \\
& \alpha_t^i \in \{0, 1\}, \quad i \in [1, n], \quad t \in \mathcal{T}_L^{c_i}, & (\text{P4d}) \\
& \beta_t^i \leq \text{LE}(x^i, t), \quad i \in [1, n], \quad t \in \mathcal{T}_L^{c_i}, & (\text{P4e}) \\
& \beta_t^i \geq \text{LE}(x^i, t) - B(s)(1 - \alpha_t^i), \quad i \in [1, n], \quad t \in \mathcal{T}_L^{c_i}, & (\text{P4f}) \\
& 0 \leq \beta_t^i \leq B(s)\alpha_t^i, \quad i \in [1, n], \quad t \in \mathcal{T}_L, & (\text{P4g}) \\
& (\omega^b)^\top x^i - \gamma_b \leq z_i^b M - 1, \quad b \in \mathcal{T}_B, \quad i \in \mathcal{U}, & (\text{P4h}) \\
& (\omega^b)^\top x^i - \gamma_b \geq -(1 - z_i^b)M + 1, \quad b \in \mathcal{T}_B, \quad i \in \mathcal{U}, & (\text{P4i}) \\
& z_i^b \in \{0, 1\}, \quad b \in \mathcal{T}_B, \quad i \in \mathcal{U}, & (\text{P4j}) \\
& \delta_i^t \leq z_i^b, \quad b \in \mathcal{N}_R(t), \quad t \in \mathcal{T}_L^A, \quad i \in \mathcal{U}, & (\text{P4k}) \\
& \delta_i^t \leq -z_i^b + 1, \quad b \in \mathcal{N}_L(t), \quad t \in \mathcal{T}_L^A, \quad i \in \mathcal{U}, & (\text{P4l}) \\
& \delta_i^t \geq \sum_{b \in \mathcal{N}_R(t)} z_i^b + \sum_{b \in \mathcal{N}_L(t)} (-z_i^b + 1) - (D - 1), \quad t \in \mathcal{T}_L^A, \quad i \in \mathcal{U}, & (\text{P4m}) \\
& \delta_i^t \in \{0, 1\}, \quad t \in \mathcal{T}_L^A, \quad i \in \mathcal{U}, & (\text{P4n}) \\
& \lambda - \xi \leq \sum_{i=1+n}^N \sum_{t \in \mathcal{T}_L^A} \delta_i^t \leq \lambda + \xi, & (\text{P4o}) \\
& \xi \geq 0, & (\text{P4p})
\end{aligned}$$

where $\mathcal{U} := [n + 1, \dots, N]$ and $B(s) := D(\eta s \sqrt{p} + 1)$ is an upper bound for $\text{LE}(x^i, t)$. Note that the objective function in (P4a) models a compromise between minimizing the classification error on the labeled and unlabeled data. The penalty parameter $C > 0$ aims to control the importance of the slack variable ξ . Constraints (P4c) and (P4d) enforce that the minimum value of $\text{LE}(x^i, t)$ is selected for each $x^i \in [1, n]$, and the constraints in (P4e)–(P4g) ensure

$$\beta_t^i = \alpha_t^i \text{LE}(x^i, t), \quad i \in [1, n], \quad t \in \mathcal{T}_L.$$

Regarding the unlabeled data, as z_i^b is binary, Constraints (P4h) and (P4i)

lead to

$$\begin{aligned} (\omega^b)^\top x^i - \gamma_b \geq 1 &\implies z_i^b = 1, & b \in \mathcal{T}_B, & i \in \mathcal{U}, \\ (\omega^b)^\top x^i - \gamma_b \leq -1 &\implies z_i^b = 0, & b \in \mathcal{T}_B, & i \in \mathcal{U}. \end{aligned}$$

Moreover, as δ_i^t is binary as well, for all $i \in \mathcal{U}$ and $t \in \mathcal{T}_L^A$, Constraints (P4k)–(P4m) lead to

$$\delta_i^t = \begin{cases} 1, & \text{if } z_i^b = 1 \text{ for } b \in \mathcal{N}_R(t) \text{ and } z_i^b = 0 \text{ for } b \in \mathcal{N}_L(t), \\ 0, & \text{otherwise,} \end{cases}$$

i.e.,

$$\delta_i^t = \begin{cases} 1, & \text{if } x^i \text{ ends up in the leaf node } t, \\ 0, & \text{otherwise.} \end{cases}$$

Constraint (P4o) ensures that the number of unlabeled data classified as \mathcal{A} is as close to λ as possible. Reformulation (P4) is an MILP. We refer to this problem as S²OCT. As usual in mixed-integer optimization, the big- M -value is crucial as is the choice of s in Expression (3.3). Based on s , in [MEP2] we provide an exact value for M , as discussed in the following theorem.

Theorem 2 (Theorem 1 in [MEP2]) *Consider $\eta := \max_{i,k \in [1,N]} \|x^i - x^k\|$. Any feasible point for Problem (P4) satisfies (P4h) and (P4i) for*

$$M = \eta s \sqrt{p} + 1.$$

We refer to [MEP2] for the proof of this theorem.

3.3 Computational Study

In Section 4 of [MEP2], we perform a comprehensive computational evaluation and show the advantages of knowing the total class distribution within unlabeled data for classification trees. We focus our study on non-representative biased samples. To achieve this, we utilize various test sets from the literature. For each dataset, we generate 5 biased samples, where only 10 percent of the data is labeled. We then compare the model of a multivariate tree proposed by Bertsimas and Dunn (2017), OCT-H, and our model S²OCT as given in Problem (P4). Moreover, we set a time limit of 2 h per sample for each method. Here, we report the main discussion and conclusions drawn from the computational study. We refer to [MEP2] for all the details of our numerical results.

Table 3.1: Different quantile values for the number of continuous and binary variables.

	Continuous		Binary	
	OCT-H	S ² OCT	OCT-H	S ² OCT
min	31	43	42	151
25 %	61	195	123	846
50 %	97	366	226	1843
75 %	319	3028	1451	16 469
max	14 039	121 910	54 373	695 835

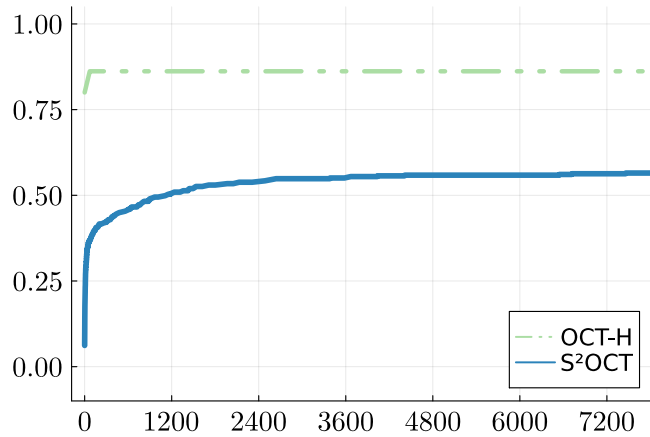


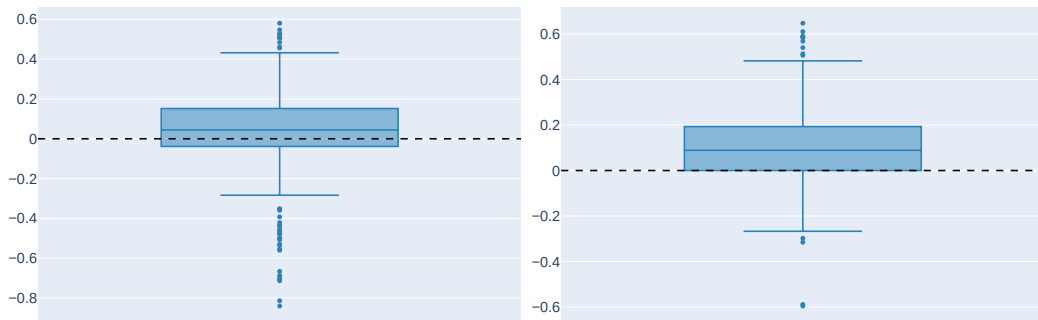
Figure 3.3: ECDFs for run times (in seconds).

Before presenting the results, observe that OCT-H only considers the labeled points, while S²OCT considers the labeled and unlabeled sets as well as the cardinality constraint of each class. Due to this, OCT-H is a considerably simpler model than S²OCT. To show the different complexities, in Table 3.1 we provide a quantile analysis of the number of continuous and binary variables in each approach for all instances utilized in the numerical tests.

Similar to the discussion in Section 2.4, our first analysis from the numerical results are the ECDFs for run time. Table 3.1 shows that S²OCT has more variables than OCT-H. Therefore, it is expected that OCT-H solves more instances than S²OCT within the limit. Figure 3.3 shows that OCT - H solved 86 % of the instances within the time limit, while S²OCT does so for 58 %. As expected, OCT-H has significantly shorter run times, and introducing the cardinality constraint leads to larger computational costs.

In our study, one evaluation measure is again accuracy, i.e., the proportion

of points correctly classified. The main question is: does S²OCT or OCT-H have a higher accuracy for a specific instance? For that, in Figure 3.4 we present the box plots with the difference in the accuracy between S²OCT and OCT-H. Observe that a value greater than zero indicates that S²OCT had a better result than OCT-H and lower than zero indicates that OCT-H had a better result than S²OCT. As already mentioned in Section 2.4, the box in the boxplot depicts the range of the medium 50% of the values; 25% of the values are below and 25% are above the box. Figure 3.4a presents the difference in accuracy for all instances. As can be seen, this difference is greater than zero in the majority of the instances. Hence, we can conclude that our proposed method takes advantage of the additional information on the total number of cases for the classes and has a better accuracy than OCT-H. Moreover, the negative outliers indicate worse accuracy for S²OCT than OCT-H in some cases. This happens because in some instances our method does not terminate within the time limit while OCT-H does. When comparing only instances that both approaches terminate within the time limit, Figure 3.4b shows that S²OCT has better accuracy than OCT-H for 75% of the instances in the results.



(a) All instances.

(b) Instances that both methods solve.

Figure 3.4: Difference in accuracy. Comparison for all data points.

For other comparisons, such as those based on Matthews correlation coefficient and precision, we refer to [MEP2], respectively. As in [MEP1], there we also present numerical results for simple random samples. In these cases, our proposed semi-supervised method performs as good as OCT-H.

Chapter 4

Random Forests

Many machine learning algorithms are susceptible to overfitting, meaning they prioritize performing well on the training data but fail to generalize the classification of the unlabeled points. To overcome this issue, combining predictions from multiple models can be a powerful approach. This is called ensemble learning; for more details, we refer to Brown (2010) and Re and Valentini (2012). In this chapter, we focus on random forests (Breiman 2001), an ensemble learning approach that combines predictions from several decision trees. Each tree is trained on a random subset of features and data points. In the context of classification, once trained, each tree independently classifies an unlabeled point. Then, the final classification of this point is the majority vote, i.e., the dominant class chosen by the individual trees. Nevertheless, external sources may inform how many unlabeled points belongs to each class. In these cases, the final classification made by a majority vote may not comply with this constraint. In [MEP3] we propose a model to aggregate the information of how many points belong to each class in the random forest setting for binary classification. Moreover, we present some preprocessing techniques and a branching rule to reduce the computational cost of solving the proposed model.

The remainder of this chapter is organized as follows. In Section 4.1 we describe our proposed model for cardinality-constrained random forests. Then, in Section 4.2 we provide a short description of the preprocessing techniques and the branching rule that we propose in [MEP3]. Section 4.3 presents the numerical experiments and the conclusion of [MEP3].

4.1 Cardinality-Constrained Random Forests

Consider $X_l \in \mathbb{R}^{p \times n}$ as being the labeled data matrix, $y \in \{-1, 1\}^n$ as the vector of class labels for the labeled points and $X_u = [x^1, \dots, x^m] \in \mathbb{R}^{p \times m}$ as

being the unlabeled data matrix. If $y_i = 1$, we say that x^i is in the positive class and if $y_i = -1$, x^i belongs to the negative class. Let t be the number of given decision trees, a subset $A^j \in \mathbb{R}^{p \times d}$ of the labeled data with size d is selected, for $j \in [1, t]$. In a random forest, for each $j \in [1, t]$, based on each column of A^j and its label, the j th tree generates a vector $r^j \in \{-1, 1\}^m$ that classifies each unlabeled point in X_u . Hence, for each unlabeled point x^i we observe a vector of classifications $r_i = [r_i^1, \dots, r_i^t] \in \{-1, 1\}^t$. Thus, $R = [r_1, \dots, r_m] \in \{-1, 1\}^{t \times m}$ and r_i^j is the classification of $x^i \in X_u$ given by the tree j . To illustrate this, let $[x^1, \dots, x^6]$ be a set of unlabeled data. Consider 5 trees, i.e., $t = 5$, that provide the following matrix of classifications:

$$R = \begin{bmatrix} 1 & 1 & -1 & -1 & 1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 \\ 1 & 1 & 1 & -1 & 1 & -1 \\ -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & -1 & 1 \end{bmatrix}. \quad (4.1)$$

Each row of the matrix above indicates the classification of each tree. For instance, the first row shows that the first tree classifies x^1, x^2, x^5 , and x^6 as positive and x^3 and x^4 as negative. On the other hand, each column of matrix R shows the classification of each point. For example, the first column indicates that x^1 is classified as negative by the fourth tree and as positive by the others trees.

In a random forest, the classification of a point $x^i \in X_u$ is done by the majority vote. In our running example, x^1, \dots, x^4 are classified as positive whereas x^5 and x^6 are classified as negative. However, the cardinality constraint of each class could be provided by an undisclosed source as already discussed in the previous chapters. In what follows we assume to know the total number λ of unlabeled points that belong to the positive class. For the example with the matrix R described in (4.1), let us assume that $\lambda = 3$. This means that the random forest by majority vote classifies the unlabeled data without respecting the cardinality constraint. In [MEP3] we propose to use a linear combination of the tree classifications to deal with this constraint. For this, our goal is to find optimal decision variables $\alpha^* \in \mathbb{R}^t$, $\eta^* \in \mathbb{R}$, and $z^* \in \{0, 1\}^m$ that solve the optimization problem

$$\min_{\alpha, \eta, z} \eta \quad (\text{P5a})$$

$$\text{s.t. } \alpha^\top r_i \leq -1 + z_i M, \quad i \in [1, m], \quad (\text{P5b})$$

$$\alpha^\top r_i \geq 1 - (1 - z_i)M, \quad i \in [1, m], \quad (\text{P5c})$$

$$\lambda - \eta \leq \sum_{i=1}^m z_i \leq \lambda + \eta, \quad (\text{P5d})$$

$$\ell \leq \alpha_j \leq u, \quad j \in [1, t], \quad (\text{P5e})$$

$$0 \leq \eta \leq \bar{\eta}, \quad (\text{P5f})$$

$$z_i \in \{0, 1\}, \quad i \in [1, m], \quad (\text{P5g})$$

where M needs to be chosen sufficiently large, $u > \ell > 0$ holds, and $\bar{\eta} := \max\{\lambda, m - \lambda\}$.

In particular, in [MEP3] we show that a valid M is linear in the number of trees in the forest. Observe that the objective function in (P5a) minimizes the classification error on the unlabeled data. As z_i is binary, Constraints (P5b) and (P5c) lead to

$$\begin{aligned} \alpha^\top r_i \geq 1 &\implies z_i = 1, \quad i \in [1, m], \\ \alpha^\top r_i \leq -1 &\implies z_i = 0, \quad i \in [1, m]. \end{aligned}$$

Constraint (P5d) ensures that the number of unlabeled data points classified as positive is as close to λ as possible. Constraint (P5e) bounds the weight of each tree's decision for the final classification. This means that for $j \in [1, t]$, as α_j gets closer to u , the j th tree gets greater influence on the final classification, and as α_j gets closer to ℓ , the j th tree has less influence on the final classification. If α_j has the same value for all $j \in [1, t]$, all trees contribute equally to the final classification and we are in the standard random forest setup with majority vote. Although the upper bound $\bar{\eta}$ is not necessary for the correctness of Model (P5), it does not cut off any solution. Thus, we include this bound in our implementation because tight bounds can improve the solution process. Problem (P5) is an MILP. We refer to this problem as C²RF (Cardinality-Constrained Random Forest).

Since we now stated our model, let us go back to the example with the matrix of classification given by R in (4.1) and $\lambda = 3$. Considering $\ell = 1$ and $u = 10$, a solution of Problem (P5) is given by

$$\alpha^* = [3 \ 1 \ 1 \ 5 \ 1]^\top, \quad \eta^* = 0, \quad z^* = [1 \ 0 \ 1 \ 1 \ 0 \ 0]^\top,$$

That is, x^1, x^3 , and x^4 are classified as positive and the other points are classified as negative. This classification complies with the cardinality constraint $\lambda = 3$.

4.2 Techniques to Solve the C²RF Problem

As mentioned in Section 4.1, C²RF is an MILP, in which the computation time grows with the number of variables—especially if they are binary variables. Hence, reducing the number of variables, especially the binary ones, can be beneficial. Providing a branching rule can also be helpful for the performance of MILP solvers. In this section, we provide a short description of the pre-processing techniques and the branching rule proposed in [MEP3] to reduce the model size and improve the efficiency of solving the C²RF problem.

4.2.1 Preprocessing

The first preprocessing technique present in [MEP3] is given by the next proposition and is based on the following insight. If all trees have the same classification for some unlabeled points, these points must have the same final classification, and, therefore, the respective binary variables always have the same values.

Proposition 1 (Proposition 3 in [MEP3]) *Let $k \in [1, m]$ and consider $\mathcal{K} := \{i \in [1, m] : r_i = r_k\}$. Then, (α, η, z) is a feasible point of Problem (P5) if and only if exist $\bar{z} \in \{0, 1\}^{m+1-|\mathcal{K}|}$ such that (α, η, \bar{z}) is a feasible point of the problem*

$$\min_{\alpha, \eta, z} \eta \tag{P6a}$$

$$\text{s.t. } \alpha^\top r_i \leq -1 + z_i M, \quad i \in \{k\} \cup [1, m] \setminus \mathcal{K}, \tag{P6b}$$

$$\alpha^\top r_i \geq 1 - (1 - z_i)M, \quad i \in \{k\} \cup [1, m] \setminus \mathcal{K}, \tag{P6c}$$

$$\lambda - \eta \leq \sum_{i \in [1, m] \setminus \mathcal{K}} z_i + |\mathcal{K}|z_k \leq \lambda + \eta, \tag{P6d}$$

$$\text{(P5e), (P5f)}, \tag{P6e}$$

$$z_i \in \{0, 1\}, \quad i \in \{k\} \cup [1, m] \setminus \mathcal{K}. \tag{P6f}$$

We refer to [MEP3] for the proof of Proposition 1. Moreover, if one or more trees classify all points exactly as another tree, these trees must have equal importance in the final classification, and some continuous variables of Problem (P5) can be eliminated. This is established by the following proposition.

Proposition 2 (Proposition 4 in [MEP3]) *Given $g \in [1, t]$, consider $\mathcal{G} := \{j \in [1, t] : r^g = r^j\}$. Then, (α^*, η^*, z^*) is a solution to Problem (P5) if*

and only if exist $\bar{\alpha} \in \mathbb{R}^{t+1-|\mathcal{G}|}$ such that $(\bar{\alpha}, \eta^*, z^*)$ is a solution to the problem

$$\min_{\alpha, \eta, z} \eta \quad (\text{P7a})$$

$$\text{s. t.} \quad \sum_{j \in [1, t] \setminus \mathcal{G}} \alpha_j r_i^j + |\mathcal{G}| \alpha_g r_i^g \leq -1 + z_i M, \quad i \in [1, m], \quad (\text{P7b})$$

$$\sum_{j \in [1, t] \setminus \mathcal{G}} \alpha_j r_i^j + |\mathcal{G}| \alpha_g r_i^g \geq 1 - (1 - z_i) M, \quad i \in [1, m], \quad (\text{P7c})$$

$$(\text{P5d}), \quad (\text{P7d})$$

$$\ell \leq \alpha_j \leq u, \quad j \in \{g\} \cup [1, t] \setminus \mathcal{G}, \quad (\text{P7e})$$

$$(\text{P5f}), (\text{P5g}). \quad (\text{P7f})$$

We refer to [MEP3] for the proof of Proposition 2. In [MEP3] we also present the next proposition that specifies in which cases some binary variables z_i of Problem (P5) can be fixed.

Proposition 3 (Proposition 5 in [MEP3]) *For each $i \in [1, m]$, consider $\mathcal{A}_i = \{j \in [1, t]: r_i^j = -1\}$ and $\mathcal{B}_i = \{j \in [1, t]: r_i^j = 1\}$. If for some $i \in [1, m]$,*

$$\varphi_i := -u|\mathcal{A}_i| + \ell|\mathcal{B}_i| \geq 1 \quad (4.2)$$

holds, then any feasible point (α, η, z) of Problem (P5) satisfies $z_i = 1$. On the other hand, if

$$\phi_i := -\ell|\mathcal{A}_i| + u|\mathcal{B}_i| \leq -1 \quad (4.3)$$

is satisfied for some $i \in [1, m]$, then any feasible point (α, η, z) of Problem (P5) satisfies $z_i = 0$.

We refer to [MEP3] for the proof of Proposition 3. Consider now

$$\mathcal{P} := \{i \in [1, m]: \varphi_i \geq 1\} \quad \text{and} \quad \mathcal{N} := \{i \in [1, m]: \phi_i \leq -1\}.$$

By the proposition above, $|\mathcal{P}| + |\mathcal{N}|$ binary variables can be fixed. Moreover, $|\mathcal{P}|$ points are then already classified as positive. If $|\mathcal{P}| \geq \lambda$, due to cardinality constraint, all remaining points $x^i \in X_u \setminus (\mathcal{P} \cup \mathcal{N})$ must be classified as negative, and λ can be set to 0. On the other hand, if $|\mathcal{P}| < \lambda$, only $\lambda - |\mathcal{P}|$ points in $X_u \setminus (\mathcal{P} \cup \mathcal{N})$ must be classified as positive.

4.2.2 Branching Priorities

One fact that affects the performance of MILP solvers is the order of selection of a fractional variable for branching during the tree search. Let us consider now binary variables $z_i, z_k \in \{0, 1\}$, $i, k \in [1, m]$, and positive integer value ξ_i and ξ_k so that $\xi_i > \xi_k$ implies that the solver should branch on z_i before z_k . In our context, a point for which the percentage of trees that classify the point as positive (or negative) is larger than for another point seems to be “easier” to classify. Hence, we want to branch on the respective binary variable first. Based on that, we establish a criterion for a branching strategy.

We set $\theta_i = |\text{mean}(r_i)|$ for each $x^i \in X_u \setminus (\mathcal{P} \cup \mathcal{N})$. Observe that the higher the value of θ_i , the more trees classified the point x^i into one specific class. Hence, we consider ξ_i the position of θ_i in the vector of the increasingly sorted values of θ . Thus, the higher the value of θ_i , the higher the value of ξ_i , and, hence, the higher the branch priority of the point x^i .

Motivated by the preprocessing techniques and the branching priorities discussed in this extended summary, in [MEP3] we propose an algorithm to solve Problem (P5) called p-C²RF, given in Algorithm 2.

4.3 Numerical Experiments

In Section 5 of [MEP3], we present and discuss computational results that demonstrate the benefits of considering the total amount of points in each class for the random forests setting. We also discuss the advantages of using our preprocessing techniques and branching rule to speed up the solution process. As in [MEP1] and [MEP2], we focus our analysis on biased samples. For this, we consider various test sets from the literature. Then, for each data set, we generate 5 biased samples, where only 1% of the data is labeled. Afterward, we compare the following approaches.

- (a) RF: random forest by majority vote.
- (b) C²RF as given in Problem (P5).
- (c) p-C²RF as described in Algorithm 2.
- (d) only PP: Algorithm 2 without the branching rule described in Step 18.
- (e) only BR: C²RF as given in Problem (P5) with the branching rule as described above but without our problem-tailored preprocessing.

Each approach has a time limit of 2 h. The first evaluation from the numerical study is the ECDFs for the run time. Figure 4.1 shows that RF is the fastest

Algorithm 2: Pre-processing C²RF (p-C²RF) (Alg. 1 in [MEP3])

Input : $R \in \{-1, 1\}^{t \times m}$, $u > \ell > 0$, $\lambda \in \mathbb{N}$, $\mathcal{K} = \emptyset$, $\beta = 0$, and $\gamma = 0$.

- 1 Compute $M = ut + 1$ and $\bar{R} = [\bar{r}_1, \dots, \bar{r}_h] \in \{-1, 1\}^{t \times h}$ being the set of all different $r_i \in R$.
- 2 **for** $k \in \{1, \dots, h\}$ **do**
- 3 Compute $w_k = |\{i \in [1, m] : r_i = \bar{r}_k\}|$, φ_k as described in (4.2), and ϕ_k as described in (4.3).
- 4 **if** $\varphi_k \geq 1$ **then**
- 5 Set $\mathcal{K} \leftarrow \mathcal{K} \cup \{k\}$ and $\beta \leftarrow \beta + w_k$.
- 6 **else if** $\phi_k \leq -1$ **then**
- 7 Set $\mathcal{K} \leftarrow \mathcal{K} \cup \{k\}$ and $\gamma \leftarrow \gamma + w_k$.
- 8 **end**
- 9 **end**
- 10 Compute $S = [s^1, \dots, s^q]^\top \in \{-1, 1\}^{q \times h}$ being the set of all different $\bar{r}^j \in \bar{R}$.
- 11 **for** $g \in \{1, \dots, q\}$ **do**
- 12 Compute $v_g = |\{j \in [1, t] : r^j = \bar{r}^g\}|$ and set $s^g \leftarrow v_g s^g$.
- 13 **end**
- 14 **for** $i \in \{1, \dots, h\} \setminus \mathcal{K}$ **do**
- 15 Compute $\theta_i = |\text{mean}(s_i)|$.
- 16 **end**
- 17 **for** $i \in \{1, \dots, h\} \setminus \mathcal{K}$ **do**
- 18 Compute ξ_i , i.e., the position of θ_i in the vector of the increasingly sorted values of θ .
- 19 **end**
- 20 Compute $\bar{\lambda} = \min\{0, \lambda - \beta\}$ and $\bar{\eta} = \max\{\bar{\lambda}, m - \beta - \gamma - \bar{\lambda}\}$ and solve

$$\begin{aligned}
 & \min_{\alpha, \eta, z} \quad \eta \\
 & \text{s.t.} \quad \alpha^\top s_i \leq -1 + z_i M, \quad i \in [1, h] \setminus \mathcal{K}, \\
 & \quad \quad \alpha^\top s_i \geq 1 - (1 - z_i)M, \quad i \in [1, h] \setminus \mathcal{K}, \\
 & \quad \quad \bar{\lambda} - \eta \leq \sum_{i \in [1, h] \setminus \mathcal{K}} w_i z_i \leq \bar{\lambda} + \eta, \\
 & \quad \quad \ell \leq \alpha_j \leq u, \quad j \in [1, q], \\
 & \quad \quad 0 \leq \eta \leq \bar{\eta}, \\
 & \quad \quad z_i \in \{0, 1\}, \quad i \in [1, h] \setminus \mathcal{K}.
 \end{aligned}$$

with branching priorities ξ to compute α^*, η^*, z^* .

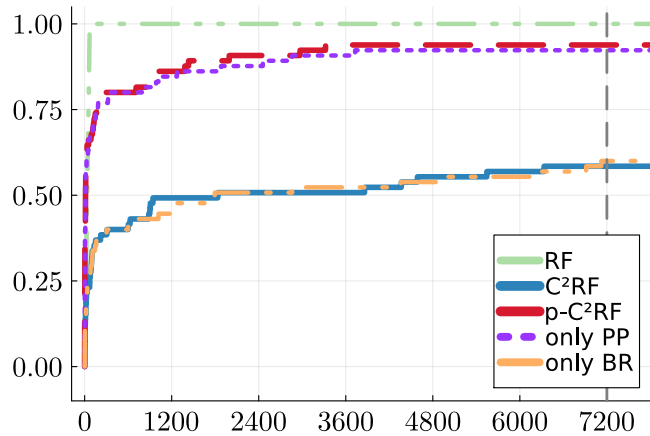


Figure 4.1: ECDFs for run times (in seconds).

algorithm. This is the case because RF does not include any binary variable related to the unlabeled points as C^2RF and $p-C^2RF$ do. It can be seen that $p-C^2RF$ outperforms C^2RF . C^2RF solves only 58% of the instances within the time limit, while $p-C^2RF$ solves 94%. This shows that the preprocessing techniques and the branching priorities significantly decrease the run time. However, by comparing the two lines for “only PP” and “only BR”, we see that most of the speed-up is obtained by the preprocessing techniques while the branching rule only helps to improve the performance for a few instances.

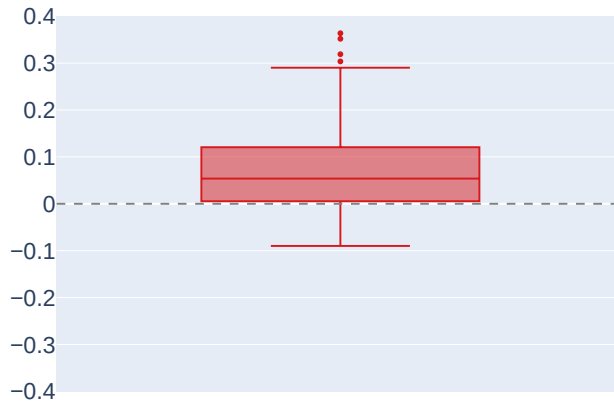


Figure 4.2: Difference in the accuracy. Comparisons for the unlabeled points.

The second analysis is again about accuracy. Observe that, different from Sections 2.4 and 3.3 our approaches for random forests only classify unlabeled points. Moreover, since C^2RF , $p-C^2RF$, only PP, and only BR solve the same

problem to optimality, we only compare the difference in the accuracy of RF and $\mathbf{p}\text{-C}^2\text{RF}$ for each instance. Note that a greater value than zero indicates that $\mathbf{p}\text{-C}^2\text{RF}$ has better accuracy than the standard random forest. Figure 4.2 shows that $\mathbf{p}\text{-C}^2\text{RF}$ has better accuracy in 75% of the instances. We can therefore conclude that our proposed approach benefits from the aggregated information of the cardinality constraint in each class.

For further comparisons such as those based on Matthews correlation coefficient, we refer to [MEP3]. In [MEP3] we also present the results for simple random samples, where we conclude that, in these cases, the proposed method $\mathbf{p}\text{-C}^2\text{RF}$ and the standard random forest are very similar in accuracy.

Chapter 5

Conclusions and Outlook

In many classification tasks, acquiring labels for the entire population of interest can be expensive. Fortunately, aggregate information on the cardinality constraint of each class can be available from external sources. This information can be particularly useful in biased samples, where the data misrepresent the entire population. In this thesis, we propose aggregating this information for three popular methods for binary classification: support vector machines (SVM), classification trees, and random forests. For each method, we present a mixed-integer programming (MIP) model for computing a semi-supervised approach.

Compared to the standard approaches, in the case of simple random samples, our proposed methods perform as good as the standard approaches in statistical performance. However, in many applications, the available data come from non-probability samples. Thus, there is the risk of obtaining biased samples. In this setting, the models we propose achieve significantly higher statistical performance than the standard approach. This confirms that, when the data is biased, considering how many points are in each class improves the prediction performance.

Furthermore, in the SVM setting, our proposed re-clustering method significantly helps to decrease the run time of the MIP approach, while still maintaining the same accuracy. Besides that, the re-clustering method finds an objective function value very close to the optimal value and, when used as a warm-start, it helps to improve the quality of the objective function value.

In the context of optimal classification trees, we present a big- M model, apply SOS techniques, and present theoretical results on the correct value of certain bounds, such as the big- M in the formulation. These bounds are crucial to prevent cutting off optimal solutions and help the solvers avoid numerical troubles in solving the MIP model. As expected, the drawback of introducing the cardinality constraint is that we get larger computational cost. Therefore, the development of techniques to reduce the computational

burden of solving the proposed model is certainly a direction of future work.

Regarding our proposed approach to random forests, we also introduce intuitive problem-tailored preprocessing techniques and a branching rule to solve the MIP model. Our techniques help to find a solution to the MIP problem in a shorter time than just solving the classic formulation using a standard solver. Thus, we can solve larger problems, such as those containing around 70000 data points, even if only few labeled points are available.

Despite these contributions in the context of binary classification, there is still room for improvement and future research. The first one is the generalization of our approaches to multi-class problems. In the context of SVM, this could be done with the one-vs.-rest strategy. In the classification trees setting, other models that consider multiclass could be adapted to include the cardinality constraints. For the random forest approach, each class could have an associated binary variable. However, in all cases, there would inevitably be an increase in computational cost. Hence, new strategies to mitigate this cost increase can also be explored.

Besides that, our random forest model is flexible enough to incorporate preliminary classifications from a variety of classifiers, beyond decision trees, resulting in an ensemble learning method.

Moreover, additional information about the number of data points in each class might be available at regional levels. For instance, statistical agencies often provide detailed information about individual states within a country. Therefore, developing models that consider the cardinality constraint for regionalized predictions could be a valuable direction for future work as well.

Bibliography

- MacQueen, J. (1967). “Some Methods for Classification and Analysis of Multivariate Observations.” In: *5th Berkeley Symposium on Mathematical Statistics and Probability*, pp. 281–297. URL: <http://projecteuclid.org/euclid.bsmsp/1200512992>.
- Beale, E. and J. Tomlin (1969). “Special facilities in a general mathematical programming system for nonconvex problems using ordered sets of variables.” In: *Operational Research* 69, pp. 447–454. DOI: [10.1007/BF01580653](https://doi.org/10.1007/BF01580653).
- McCormick, G. P. (1976). “Computability of Global Solutions to Factorable Nonconvex Programs: Part I – Convex Underestimating Problems.” In: *Mathematical Programming* 10.1, pp. 147–175. DOI: [10.1007/BF01580665](https://doi.org/10.1007/BF01580665).
- Lloyd, S. (1982). “Least squares quantization in PCM.” In: *IEEE Transactions on Information Theory* 28.2, pp. 129–137. DOI: [10.1109/TIT.1982.1056489](https://doi.org/10.1109/TIT.1982.1056489).
- Breiman, L., J. H. Friedman, R. A. Olshen, and C. J. Stone (1984). *Classification and Regression Trees*. Wadsworth. DOI: [10.1201/9781315139470](https://doi.org/10.1201/9781315139470).
- Quinlan, J. R. (1986). “Induction of Decision Trees.” In: *Machine Learning* 1, pp. 81–106. DOI: doi.org/10.1007/BF00116251.
- Boser, B. E., I. M. Guyon, and V. N. Vapnik (1992). “A Training Algorithm for Optimal Margin Classifiers.” In: *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*. COLT '92. Pittsburgh, Pennsylvania, USA: ACM Press, pp. 144–152. DOI: [10.1145/130385.130401](https://doi.org/10.1145/130385.130401).
- Brodley, C. E. and P. E. Utgoff (1995). “Multivariate Decision Trees.” In: *Machine Learning* 19.1, 45–77. DOI: [10.1023/A:1022607123649](https://doi.org/10.1023/A:1022607123649).
- Cortes, C. and V. Vapnik (1995). “Support Vector Networks.” In: *Machine Learning* 20, pp. 273–297. DOI: [10.1007/BF00994018](https://doi.org/10.1007/BF00994018).
- Bennett, K. P. and J. A. Blue (1996). “Optimal Decision Trees.” In: *Rensselaer Polytechnic Institute Math Report* 214. URL: https://www.researchgate.net/publication/2796065_Optimal_Decision_Trees.
- Bennett, K. P. and A. Demiriz (1998). “Semi-Supervised Support Vector Machines.” In: *Proceedings of the 11th International Conference on Neural Information Processing Systems*. NIPS'98. MIT Press, pp. 368–374. URL:

- https://proceedings.neurips.cc/paper_files/paper/1998/file/b710915795b9e9c02cf10d6d2bdb688c-Paper.pdf.
- Mangasarian, O. (1999). “Arbitrary-norm separating plane.” In: *Operations Research Letters* 24.1, pp. 15–23. DOI: [10.1016/S0167-6377\(98\)00049-2](https://doi.org/10.1016/S0167-6377(98)00049-2).
- Breiman, L. (2001). “Random Forests.” In: *Machine Learning* 45.1, pp. 5–32. DOI: [10.1023/A:1010933404324](https://doi.org/10.1023/A:1010933404324).
- Amini, M.-R. and P. Gallinari (2002). “Semi-Supervised Logistic Regression.” In: *Proceedings of the 15th European Conference on Artificial Intelligence. ECAI’02*. Lyon, France: IOS Press, pp. 390–394. URL: <https://hal.science/hal-01561456>.
- Joachims, T. (2002). “Training Transductive Support Vector Machines.” In: *Learning to Classify Text Using Support Vector Machines*. Springer US, pp. 163–174. DOI: [10.1007/978-1-4615-0907-3_9](https://doi.org/10.1007/978-1-4615-0907-3_9).
- Orsenigo, C. and C. Vercellis (2003). “Multivariate classification trees based on minimum features discrete support vector machines.” In: *IMA Journal of Management Mathematics* 14.3, pp. 221–234. DOI: [10.1093/imaman/14.3.221](https://doi.org/10.1093/imaman/14.3.221).
- Chapelle, O. and A. Zien (2005). “Semi-Supervised Classification by Low Density Separation.” In: *Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics*. Ed. by R. G. Cowell and Z. Ghahramani. Vol. R5. Proceedings of Machine Learning Research. PMLR, pp. 57–64. URL: <http://proceedings.mlr.press/r5/chapelle05b/chapelle05b.pdf>.
- Belkin, M., P. Niyogi, and V. Sindhwani (2006). “Manifold regularization: a geometric framework for learning from labeled and unlabeled examples.” In: *Journal of Machine Learning Research* 7, pp. 2399–2434. URL: <http://jmlr.org/papers/v7/belkin06a.html>.
- Chapelle, O., M. Chi, and A. Zien (2006). “A Continuation Method for Semi-Supervised SVMs.” In: *Proceedings of the 23rd International Conference on Machine Learning. ICML ’06*. New York, NY, USA: Association for Computing Machinery, pp. 185–192. DOI: [10.1145/1143844.1143868](https://doi.org/10.1145/1143844.1143868).
- Altincay, H. (2007). “Decision trees using model ensemble-based nodes.” In: *Pattern Recognition* 40, pp. 3540–3551. DOI: [10.1016/j.patcog.2007.03.023](https://doi.org/10.1016/j.patcog.2007.03.023).
- Yildiz, O. and O. Dikmen (2007). “Parallel Univariate Decision Trees.” In: *Pattern Recognition Letters* 28, pp. 825–832. DOI: [10.1016/j.patrec.2006.11.009](https://doi.org/10.1016/j.patrec.2006.11.009).
- Melacci, S. and M. Belkin (2009). “Laplacian Support Vector Machines Trained in the Primal.” In: *Journal of Machine Learning Research* 12. DOI: [10.48550/ARXIV.0909.5422](https://doi.org/10.48550/ARXIV.0909.5422).

- Zhu, X. and A. B. Goldberg (2009). *Introduction to Semi-Supervised Learning*. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool Publishers. DOI: [10.2200/S00196ED1V01Y200906AIM006](https://doi.org/10.2200/S00196ED1V01Y200906AIM006).
- Brown, G. (2010). “Ensemble Learning.” In: *Encyclopedia of Machine Learning*. Ed. by C. Sammut and G. I. Webb. Boston, MA: Springer US, pp. 312–320. DOI: [10.1007/978-0-387-30164-8_252](https://doi.org/10.1007/978-0-387-30164-8_252).
- Re, M. and G. Valentini (2012). “Ensemble methods: A review.” In: *Advances in Machine Learning and Data Mining for Astronomy*, pp. 563–594. URL: <https://api.semanticscholar.org/CorpusID:14575090>.
- Xu Yu Jing Yang, J.-p. Z. (2012). “A Transductive Support Vector Machine Algorithm Based on Spectral Clustering.” In: *AASRI Procedia 1*. Elsevier Ltd, pp. 384–388. DOI: [10.1016/j.aasri.2012.06.059](https://doi.org/10.1016/j.aasri.2012.06.059).
- Lee, D.-H. (2013). “Pseudo-Label : The Simple and Efficient Semi-Supervised Learning Method for Deep Neural Networks.” In: *ICML 2013 Workshop : Challenges in Representation Learning (WREPL)*. URL: https://www.researchgate.net/publication/280581078_Pseudo-Label_The_Simple_and_Efficient_Semi-Supervised_Learning_Method_for_Deep_Neural_Networks.
- Kotsiantis, S. (2014). “A hybrid decision tree classifier.” In: *Journal of Intelligent & Fuzzy Systems: Applications in Engineering and Technology* 26, pp. 327–336. DOI: [10.3233/IFS-120741](https://doi.org/10.3233/IFS-120741).
- Bzdok, D., M. Eickenberg, O. Grisel, B. Thirion, and G. Varoquaux (2015). “Semi-Supervised Factored Logistic Regression for High-Dimensional Neuroimaging Data.” In: *Advances in Neural Information Processing Systems*. Ed. by C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett. Vol. 28. Curran Associates, Inc. URL: https://proceedings.neurips.cc/paper_files/paper/2015/file/06a15eb1c3836723b53e4abca8d9b879-Paper.pdf.
- Bertsimas, D. and J. Dunn (2017). “Optimal classification trees.” In: *Machine Learning* 106.7, pp. 1039–1082. DOI: [10.1007/s10994-017-5633-9](https://doi.org/10.1007/s10994-017-5633-9).
- Kocev, M. C. D., J. Levatić, and S. Džeroski (2017). “Semi-supervised classification trees.” In: *Journal of Intelligent Information Systems* 49, pp. 461–486. DOI: [10.1007/s10844-017-0457-4](https://doi.org/10.1007/s10844-017-0457-4).
- Kontonatsios, G., A. J. Brockmeier, P. Przybyła, J. McNaught, T. Mu, J. Y. Goulermas, and S. Ananiadou (2017). “A semi-supervised approach using label propagation to support citation screening.” In: *Journal of Biomedical Informatics* 72, pp. 67–76. DOI: [10.1016/j.jbi.2017.06.018](https://doi.org/10.1016/j.jbi.2017.06.018).
- Tanha, J., M. van Someren, and H. Afsarmanesh (2017). “Semi-supervised self-training for decision tree classifiers.” In: *International Journal of Machine Learning and Cybernetics* 8, pp. 355–370. DOI: [10.1007/s13042-015-0328-7](https://doi.org/10.1007/s13042-015-0328-7).

- Oliver, A., A. Odena, C. A. Raffel, E. D. Cubuk, and I. Goodfellow (2018). “Realistic Evaluation of Deep Semi-Supervised Learning Algorithms.” In: *Advances in Neural Information Processing Systems*. Vol. 31. Curran Associates, Inc. DOI: [10.48550/arXiv.1804.09170](https://doi.org/10.48550/arXiv.1804.09170).
- Burkov, A. (2019). *The Hundred-Page Machine Learning Book*. Andriy Burkov. URL: <https://books.google.de/books?id=0jbxwQEACAAJ>.
- Verwer, S. and Y. Zhang (2019). “Learning Optimal Classification Trees Using a Binary Linear Program Formulation.” In: vol. 33. AAAI Press, pp. 1625–1632. DOI: [10.1609/aaai.v33i01.33011624](https://doi.org/10.1609/aaai.v33i01.33011624).
- Sun, S., Z. Cao, H. Zhu, and J. Zhao (2020). “A survey of optimization methods from a machine learning perspective.” In: *IEEE Transactions on Cybernetics* 50.8, pp. 3668–3681. DOI: [10.1109/TCYB.2019.2950779](https://doi.org/10.1109/TCYB.2019.2950779).
- Burgard, J. P., J. Krause, and S. Schmaus (2021). “Estimation of regional transition probabilities for spatial dynamic microsimulations from survey data lacking in regional detail.” In: *Computational Statistics & Data Analysis* 154, p. 107048. DOI: <https://doi.org/10.1016/j.csda.2020.107048>.
- Carrizosa, E., C. Molero-Río, and D. R. Morales (2021). “Mathematical optimization in classification and regression trees.” In: *TOP: An Official Journal of the Spanish Society of Statistics and Operations Research* 29.1, pp. 5–33. DOI: [10.1007/s11750-021-00594-1](https://doi.org/10.1007/s11750-021-00594-1).
- Gambella, C., B. Ghaddar, and J. Naoum-Sawaya (2021). “Optimization problems for machine learning: A survey.” In: *European Journal of Operational Research* 290.3, pp. 807–828. DOI: [10.1016/j.ejor.2020.08.045](https://doi.org/10.1016/j.ejor.2020.08.045).
- Santhiappan, S. and B. Ravindran (2021). “A Semi-Supervised Approach to Growing Classification Trees.” In: *Proceedings of the 3rd ACM India Joint International Conference on Data Science & Management of Data (8th ACM IKDD CODS & 26th COMAD)*. CODS-COMAD ’21. Bangalore, India: Association for Computing Machinery, pp. 29–37. DOI: [10.1145/3430984.3431009](https://doi.org/10.1145/3430984.3431009).
- Zhou, Z.-H. (2021). *Machine learning*. Springer nature. DOI: [10.1007/978-981-15-1967-3](https://doi.org/10.1007/978-981-15-1967-3).
- Blanco, V., A. Japón, and J. Puerto (2022). “A mathematical programming approach to SVM-based classification with label noise.” In: *Computers & Industrial Engineering* 172, p. 108611. DOI: [10.1016/j.cie.2022.108611](https://doi.org/10.1016/j.cie.2022.108611).
- Burgard, J. P., C. Moreira Costa, C. Hojny, T. Kleinert, and M. Schmidt (2023). “Mixed-integer programming techniques for the minimum sum-of-squares clustering problem.” In: *Journal of Global Optimization* 87.1, 133–189. DOI: [10.1007/s10898-022-01267-4](https://doi.org/10.1007/s10898-022-01267-4).
- Carrizosa, E., K. Kurishchenko, and D. Romero Morales (Sept. 2023). *On enhancing the explainability and fairness of tree ensembles*. URL: <https://>

[//www.researchgate.net/publication/374013681_On_enhancing_the_explainability_and_fairness_of_tree_ensembles](https://www.researchgate.net/publication/374013681_On_enhancing_the_explainability_and_fairness_of_tree_ensembles).

Nguyen, T. N. N., B. Veeravalli, and X. Fong (2023). “A Semi-Supervised Learning Method for Spiking Neural Networks Based on Pseudo-Labeling.” In: *2023 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–7. DOI: [10.1109/IJCNN54540.2023.10191317](https://doi.org/10.1109/IJCNN54540.2023.10191317).

Part II

Reprints of the Scientific Papers

Paper 1

Mixed-Integer Quadratic Optimization and Iterative Clustering Techniques for Semi-Supervised Support Vector Machines

Jan Pablo Burgard, Maria Eduarda Pinheiro, Martin Schmidt

Transactions in Operations Research (2024),

DOI: [10.1007/s11750-024-00668-w](https://doi.org/10.1007/s11750-024-00668-w)



Mixed-integer quadratic optimization and iterative clustering techniques for semi-supervised support vector machines

Jan Pablo Burgard¹ · Maria Eduarda Pinheiro² · Martin Schmidt² 

Received: 22 March 2023 / Accepted: 7 February 2024
© The Author(s) 2024

Abstract

Among the most famous algorithms for solving classification problems are support vector machines (SVMs), which find a separating hyperplane for a set of labeled data points. In some applications, however, labels are only available for a subset of points. Furthermore, this subset can be non-representative, e.g., due to self-selection in a survey. Semi-supervised SVMs tackle the setting of labeled and unlabeled data and can often improve the reliability of the results. Moreover, additional information about the size of the classes can be available from undisclosed sources. We propose a mixed-integer quadratic optimization (MIQP) model that covers the setting of labeled and unlabeled data points as well as the overall number of points in each class. Since the MIQP's solution time rapidly grows as the number of variables increases, we introduce an iterative clustering approach to reduce the model's size. Moreover, we present an update rule for the required big- M values, prove the correctness of the iterative clustering method as well as derive tailored dimension-reduction and warm-starting techniques. Our numerical results show that our approach leads to a similar accuracy and precision than the MIQP formulation but at much lower computational cost. Thus, we can solve larger problems. With respect to the original SVM formulation, we observe that our approach has even better accuracy and precision for biased samples.

Keywords Semi-supervised learning · Support vector machines · Clustering · Mixed-integer quadratic optimization

Martin Schmidt
martin.schmidt@uni-trier.de

Jan Pablo Burgard
burgardj@uni-trier.de

Maria Eduarda Pinheiro
pinheiro@uni-trier.de

¹ Department of Economic and Social Statistics, Trier University, Universitätsring 15, 54296 Trier, Germany

² Department of Mathematics, Trier University, Universitätsring 15, 54296 Trier, Germany

Mathematics Subject Classification 90C11 · 90C90 · 90-08 · 68T99

1 Introduction

Support vector machines (SVMs) are a standard approach for supervised binary classification (Boser et al. 1992; Cortes and Vapnik 1995). The core idea is to find a separating hyperplane that optimally splits the feature space in a positive and a negative side according to the positive and negative labels of the data.

Obtaining labels for all units of interest can be costly. This is especially the case if one has to do a classic survey to obtain the labels. In this case, it would be favorable to train the SVM on only partly labeled data. This yields a semi-supervised learning setting. Bennett and Demiriz (1998) formulate and solve the semi-supervised SVM (S^3VM) as a mixed-integer linear problem (MILP). Many strategies for solving S^3VM have been proposed in the following decades such as the transductive approach (TSVM) by Joachims (2002) and Yu et al. (2012) or manifold regularization (LapSVM) by Belkin et al. (2006) and Melacci and Belkin (2009). Some researchers also consider a balancing constraint as done in mean S^3VM by Kontonatsios et al. (2017) and in c^3SVM by Chapelle et al. (2006). Moreover, the balancing constraint proposed by Chapelle and Zien (2005) enforces that the proportion of unlabeled and labeled data on both sides is similar to the proportion given by the labeled data.

In many cases, however, the aggregated information about the number of positive and negative cases in a population is known from an external source. For example, in population surveys, there are population figures from official statistics agencies. This setting is studied, e.g., by Burgard et al. (2021), who develop a cardinality-constrained multinomial logit model and apply it in the context of micro-simulations. As another example, in some businesses, the total amount of positive labels could be known but not which customer has a positive or a negative label. An intuitive example is a supermarket for which the amount of cash payments is known. However, this information is not ex-post attributable to the individual customers. We propose to add this aggregated additional information to the optimization model by imposing a cardinality constraint on the predicted labels for the unlabeled data. As will be shown in our numerical experiments, this improves the accuracy of the classification of the unlabeled data. Furthermore, the inclusion of such a cardinality constraint is very useful in the case in which the labeled data is not a representative sample from the population. When obtaining the labels from process data or from online surveys, the inclusion process of the labeled data is generally not known. This is subsumed under the non-probability sample. In this case, inverse inclusion probability weighting, as typically done in survey sampling, is not applicable. By not controlling the inclusion process, strong over- or under-coverage of relevant information in the data set is possible and should be taken into account in the analysis. Not accounting for possible biases in the data generally leads to biased results.

We propose a big- M -based MIQP to solve the semi-supervised SVM problem with a cardinality constraint for the unlabeled data. Here, we restrict ourselves to the linear kernel. Other kernels such as Gaussian and polynomial ones can, in principle, be used as well. However, this would lead to additional nonlinear constraints in a our mixed-

integer model and would thus significantly increase the computational challenge of solving the problem. Although we strongly suspect that the problem is NP-hard, we have no proof for it since we focus here on solution techniques and not on a formal complexity analysis of the problem. The cardinality constraint helps to account for biased samples since the number of positive predictions on the population is bounded by the constraint. The computation time for this MIQP grows rapidly with the number of variables—especially for an increasing number of integer variables. We develop an algorithm that uses a clustering-based model reduction to reduce the computation time. Similar reduction approaches can be found for the classic SVM using, e.g., fuzzy clustering (Almasi and Rouhani 2016; Cervantes et al. 2006), clustering-based convex hulls (Birzhandi and Youn 2019), and k -means clustering (de Almeida et al. 2000; Yao et al. 2013). We prove the correctness of our iterative clustering method and further show that it computes feasible points for the original problem. Hence, it also delivers proper upper bounds. Within our iterative approach, we additionally derive a scheme for updating the required big- M values and present tailored dimension-reduction as well as warm-starting techniques.

The paper is organized as follows. In Sect. 2, we describe our optimization problem and the big- M -based MIQP formulation. Afterward, the clustering-based model reduction technique is presented in Sect. 3. There, we also present our algorithm that combines the model reduction and the MIQP formulation. In Sect. 4, we discuss some algorithmic improvements such as the handling of data points that are far away from the hyperplane and the choice of M in the big- M formulation. In Sect. 5, we present how to use the solution of our algorithm to obtain the solution of the initial MIQP formulation by fixing some points on the correct side of the hyperplane. Finally, in Sect. 6, numerical results are reported and discussed and we conclude in Sect. 7.

2 An MIQP formulation for a cardinality-constrained semi-supervised SVM

Let $X \in \mathbb{R}^{d \times N}$ be the data matrix with $X_l = [x^1, \dots, x^n]$ being the labeled data and $X_u = [x^{n+1}, \dots, x^N]$ being the unlabeled data. Hence, we have $x^i \in \mathbb{R}^d$ for all $i \in [1, N] := \{1, \dots, N\}$. We set $m := N - n$ and $y \in \{-1, 1\}^n$ is the vector of class labels for the labeled data. When the data is linearly separable, the SVM provides a hyperplane (ω, b) that separates the positively and negatively labeled data. In the case that the data is not linearly separable, the standard approach is to use the ℓ_2 -SVM by Cortes and Vapnik (1995) given by

$$\min_{\omega, b, \xi} \frac{\|\omega\|^2}{2} + C_1 \sum_{i=1}^n \xi_i \quad (\text{P1a})$$

$$\text{s.t. } y_i (\omega^\top x^i - b) \geq 1 - \xi_i, \quad i \in [1, n], \quad (\text{P1b})$$

$$\xi_i \geq 0, \quad i \in [1, n]. \quad (\text{P1c})$$

Here and in what follows, $\|\cdot\|$ denotes the Euclidean norm. However, other norms such as the 1- or the max-norm could be used as well. For being able to include

unlabeled data in the optimization process, Bennett and Demiriz (1998) propose the semi-supervised SVM (S³VM). In many applications, the aggregated information on the labels is available, e.g., from census data. In the following, we know the total number τ of positive labels for the unlabeled data from an external source. We adapt the idea of the S³VM such that we can use τ as an additional information in the optimization model. Our goal is to find optimal parameters $\omega^* \in \mathbb{R}^d$, $b^* \in \mathbb{R}$, $\xi^* \in \mathbb{R}^n$, and $\eta^* \in \mathbb{R}^2$ that solve the optimization problem

$$\min_{\omega, b, \xi, \eta} \frac{\|\omega\|^2}{2} + C_1 \sum_{i=1}^n \xi_i + C_2(\eta_1 + \eta_2) \tag{P2a}$$

$$\text{s.t. } y_i(\omega^\top x^i - b) \geq 1 - \xi_i, \quad i \in [1, n], \tag{P2b}$$

$$\tau - \eta_1 \leq \sum_{i=n+1}^N h_{\omega, b}(x^i) \leq \tau + \eta_2, \tag{P2c}$$

$$\xi_i \geq 0, \quad i \in [1, n], \tag{P2d}$$

$$\eta_1, \eta_2 \geq 0, \tag{P2e}$$

with

$$h_{\omega, b}(x) = \begin{cases} 1, & \text{if } \omega^\top x + b \geq 0, \\ 0, & \text{otherwise.} \end{cases}$$

Note that the objective function in (P2a) is a compromise between maximizing the distance between the two classes as well as minimizing the classification error for the label and the unlabeled data. The penalty parameters $C_1 > 0$ and $C_2 > 0$ aim to control the importance of the slack variables ξ and η , respectively. Constraint (P2b) enforces on which side of the hyperplane the labeled data x^i should lie. Constraint (P2c) ensures that we have τ unlabeled data on the positive side. If $\eta_1^* > 0$ holds for a solution $(\omega^*, b^*, \xi^*, \eta^*)$, then less than τ unlabeled points are classified as positive. On the other hand, if $\eta_2^* > 0$ holds, more than τ unlabeled points are classified as positive. If $\eta_1^* = \eta_2^* = 0$ holds, exactly τ unlabeled points are classified in the positive class. Note that, having assigned a very high value to C_1 or C_2 , the objective function value is dominated by these slack variables.

The function $h_{\omega, b}(\cdot)$ in Constraint (P2c) is not continuous, which means that Problem (P2) cannot be easily solved by standard solvers. A typical way to overcome this problem is to add binary variables to turn on or off the enforcement of a constraint. By introducing binary variables $z_i \in \{0, 1\}$, $i \in [n + 1, N]$, we can reformulate the optimization Problem (P2) using the following big- M formulation:

$$\min_{\omega, b, \xi, \eta, z} \frac{\|\omega\|^2}{2} + C_1 \sum_{i=1}^n \xi_i + C_2(\eta_1 + \eta_2) \tag{P3a}$$

$$\text{s.t. } y_i(\omega^\top x^i + b) \geq 1 - \xi_i, \quad i \in [1, n], \tag{P3b}$$

$$\omega^\top x^i + b \leq z_i M, \quad i \in [n + 1, N], \tag{P3c}$$

$$\omega^\top x^i + b \geq -(1 - z_i)M, \quad i \in [n + 1, N], \tag{P3d}$$

$$\tau - \eta_1 \leq \sum_{i=n+1}^N z_i \leq \tau + \eta_2, \tag{P3e}$$

$$\xi_i \geq 0, \quad i \in [1, n], \tag{P3f}$$

$$\eta_1, \eta_2 \geq 0, \tag{P3g}$$

$$z_i \in \{0, 1\}, \quad i \in [n + 1, N], \tag{P3h}$$

where M needs to be chosen sufficiently large. As z_i is binary, Constraints (P3c) and (P3d) lead to

$$\begin{aligned} \omega^\top x^i + b > 0 &\implies z_i = 1, \quad i \in [n + 1, N], \\ \omega^\top x^i + b < 0 &\implies z_i = 0, \quad i \in [n + 1, N]. \end{aligned}$$

If x^i lies on the hyperplane, i.e., $\omega^\top x^i + b = 0$, Constraints (P3c) and (P3d) hold for $z_i = 1$ and $z_i = 0$. In this case, it can be counted either on the positive or on the negative side. For this reason, Problem (P3) is not formally equivalent to Problem (P2). Reformulation (P3) is a mixed-integer quadratic problem (MIQP) in which all constraints are linear but the objective function is quadratic. We refer to this problem as CS³VM.

Since we now stated our first model, let us shed some light on the results depending on whether the standard SVM or CS³VM is used. Figure 1 shows a 2-dimensional example data set and the corresponding hyperplanes for SVM and CS³VM. In this case, $\tau = 11$, i.e., 11 unlabeled points belong to the positive class. Note that SVM only classifies 6 unlabeled points as positive, while CS³VM classifies 11 as such. The point that lies on the CS³VM hyperplane is classified as positive because the binary variable regarding this point is 1. This example shows that using τ as additional information can improve the classification of unlabeled points.

In the big- M formulation, the choice of M is crucial. If M is too small, the problem can become infeasible or optimal solutions could be cut off. If M is chosen too large, the respective continuous relaxations usually lead to bad lower bounds and solvers may encounter numerical troubles. The choice of M is discussed in the following lemma and theorem. In Lemma 1 we show how M is related to the objective function and the given data. This is then used in Theorem 2 to derive a provably correct big- M .

Lemma 1 *Given a feasible point for Problem (P3) with an objective function value f , an optimal solution $(\omega^*, b^*, \xi^*, \eta^*, z^*)$ of (P3) satisfies*

$$\|\omega^*\| \leq \sqrt{2f} \quad \text{and} \quad |b^*| \leq \|\omega^*\| \max_{i \in [1, N]} \|x^i\| + 1$$

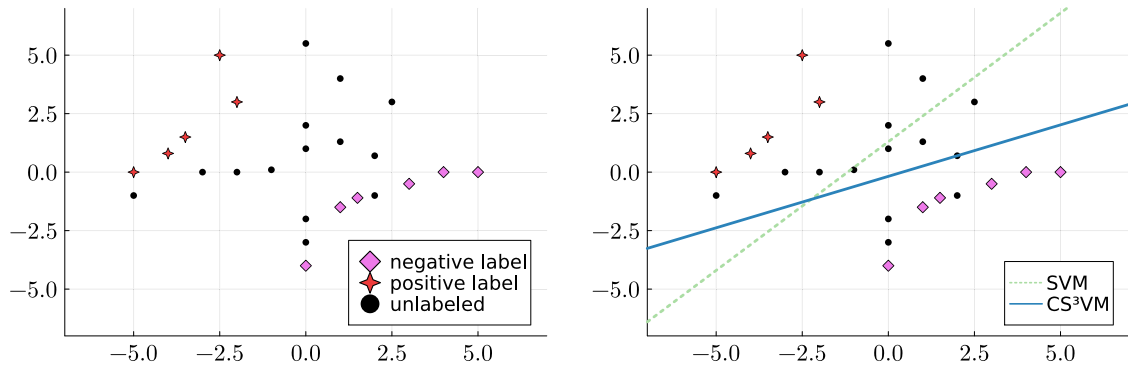


Fig. 1 A 2-dimensional example (left) and the hyperplanes resulting from the SVM and the CS³VM (right)

and, consequently, every optimal solution satisfies (P3c) and (P3d) for

$$M = 2\sqrt{2f} \max_{i \in [1, N]} \|x^i\| + 1.$$

Proof Due to optimality, we get

$$\frac{\|\omega^*\|^2}{2} \leq \frac{\|\omega^*\|^2}{2} + C_1 \sum_{i=1}^n \xi_i^* + C_2(\eta_1^* + \eta_2^*) \leq f \implies \|\omega^*\| \leq \sqrt{2f}.$$

The second inequality is shown by contradiction. To this end, we w.l.o.g. assume that $\tilde{b} = \|\omega^*\| \max_{i \in [1, N]} \|x^i\| + 1 + \delta$ is part of an optimal solution for some $\delta > 0$. Using the inequality of Cauchy–Schwarz then yields

$$\begin{aligned} (\omega^*)^\top x^i + \tilde{b} &= (\omega^*)^\top x^i + \|\omega^*\| \max_{j \in [1, N]} \|x^j\| + 1 + \delta \\ &\geq -\|\omega^*\| \|x^i\| + \|\omega^*\| \max_{j \in [1, N]} \|x^j\| + 1 + \delta \\ &> 1 \end{aligned}$$

for all $i \in [1, N]$. Hence, for all $i \in [1, n]$ with $y_i = 1$, we get $\tilde{\xi}_i = 0$ from Constraint (P3b) and the objective function. Moreover, for $i \in [1, n]$ with $y_i = -1$, the same reasoning implies

$$-(\omega^*)^\top x^i - \tilde{b} = 1 - \tilde{\xi}_i \implies \tilde{\xi}_i = 2 + (\omega^*)^\top x^i + \|\omega^*\| \max_{j \in [1, N]} \|x^j\| + \delta.$$

Besides that, for the unlabeled data $i \in [n + 1, N]$, since $(\omega^*)^\top x^i + \tilde{b} > 1$, we get $\tilde{z}_i = 1$, which leads to

$$\sum_{i=n+1}^N \tilde{z}_i = m \implies \tilde{\eta}_1 = 0, \tilde{\eta}_2 = m - \tau.$$

This means that the objective function value for the point $(\omega^*, \tilde{b}, \tilde{\xi}, \tilde{\eta}, \tilde{z})$ is given by

$$\tilde{f} := \frac{\|\omega^*\|^2}{2} + C_1 \sum_{i: y_i = -1} \left(2 + (\omega^*)^\top x^i + \|\omega^*\| \max_{j \in [1, N]} \|x^j\| + \delta \right) + C_2(m - \tau).$$

However, if we set $\bar{b} := \|\omega^*\| \max_{i \in [1, N]} \|x^i\| + 1$, we get

$$(\omega^*)^\top x^i + \bar{b} \geq 1, \quad i \in [1, N],$$

i.e., $z_i = 1$ for all $i \in [n + 1, N]$, $\bar{\eta}_1 = 0$, $\bar{\eta}_2 = m - \tau$, and $\bar{\xi}_i = 0$ for i with $y_i = 1$. Moreover, for $i \in [1, n]$ with $y_i = -1$, from Constraint (P3b) we obtain

$$-(\omega^*)^\top x^i - \tilde{b} = 1 - \bar{\xi}_i \implies \bar{\xi}_i = 2 + (\omega^*)^\top x^i + \|\omega^*\| \max_{i \in [1, N]} \|x^i\|.$$

All this implies that the objective function value \bar{f} for the point $(\omega^*, \bar{b}, \bar{\xi}, \bar{\eta}, \bar{z})$ satisfies

$$\bar{f} := \frac{\|\omega^*\|^2}{2} + C_1 \sum_{i: y_i = -1} (2 + (\omega^*)^\top x^i + \|\omega^*\| \max_{j \in [1, N]} \|x^j\|) + C_2(m - \tau) < \tilde{f},$$

which contradicts the assumption that \tilde{f} is optimal. Hence,

$$|b^*| \leq \|\omega^*\| \max_{i \in [1, N]} \|x^i\| + 1$$

holds, which proves the second inequality. Note further that

$$(\omega^*)^\top x^i + b^* \leq \|\omega^*\| \|x^i\| + |b^*| \leq 2\sqrt{2f} \max_{j \in [1, N]} \|x^j\| + 1 = M$$

and

$$(\omega^*)^\top x^i + b^* \geq -\|\omega^*\| \|x^i\| - |b^*| \geq -2\sqrt{2f} \max_{j \in [1, N]} \|x^j\| - 1 = -M$$

holds for all $i \in [n + 1, N]$. □

We now use the result from the last technical lemma to obtain a provably correct big- M .

Theorem 2 *A valid big- M for Problem (P3) is given by*

$$M = 2\sqrt{2(2C_1\bar{n} + C_2(m - \tau))} \max_{i \in [1, N]} \|x^i\| + 1 \tag{1}$$

with $\bar{n} := |\{i \in [1, n]: y_i = -1\}|$.

Proof Consider the feasible point of (P3) given by $\omega = 0 \in \mathbb{R}^d$ and $b = 1$. Since $\omega^\top x^i + b = 1$ holds for all $i \in [1, N]$, Constraint (P3b) implies

$$\xi_i = \begin{cases} 2, & \text{if } y_i = -1, \\ 0, & \text{otherwise.} \end{cases}$$

Moreover, using Constraints (P3c)–(P3e) leads to

$$z_i = 1, \quad i \in [n + 1, N], \quad \eta_1 = 0, \quad \eta_2 = m - \tau,$$

which implies that the objective function for the point $(\omega, b, \xi, \eta, z)$ is given by

$$f = 0 + 2C_1\bar{n} + C_2(m - \tau).$$

Finally, from Lemma 1, we get

$$M = 2\sqrt{2(2C_1\bar{n} + C_2(m - \tau))} \max_{i \in [1, N]} \|x^i\| + 1. \quad \square$$

3 A re-clustering method for solving CS³VM

In Model (P3) of the last section, each binary variable is related to an unlabeled point. The larger the number of unlabeled data, the larger the number of binary variables and, hence, the larger the computational burden to solve Problem (P3). To reduce this computational burden, we propose to cluster the unlabeled data. This way, only one binary variable per cluster is needed. For every cluster, we use its centroid as its representative point. To obtain clusterings, we use minimum sum-of-squares clustering (MSSC). The MSSC problem is NP-hard; see, e.g., Aloise et al. (2009), Mahajan et al. (2012), and Dasgupta (2007). However, we do not need a globally optimal solution for the MSSC problem as will be shown below. Given a number k of clusters and a matrix $S = [s^1, \dots, s^p] \in \mathbb{R}^{d \times p}$ of given points, the goal of the MSSC is to find mean vectors $c^j \in \mathbb{R}^d$, $j \in [1, k]$, that solve the problem

$$c^* = \arg \min_c \ell(S, c), \quad c = (c^j)_{j=1, \dots, k},$$

where the loss function ℓ is the sum of the squared Euclidean distances, i.e.,

$$\ell(S, c) = \sum_{j=1}^k \sum_{s^i \in \mathcal{C}_j} \|s^i - c^j\|^2$$

with $\mathcal{C}_j \subset \mathbb{R}^d$ being the set of data points that are assigned to cluster j .

We solve this problem heuristically using the k -means algorithm (MacQueen 1967; Lloyd 1982) for $S = X_u$, i.e., we cluster the unlabeled data. Then, instead of using

all unlabeled data as in the last section, we only use the clusters' centroids c^1, \dots, c^k and the numbers e_1, \dots, e_k of data points in each cluster to obtain the problem

$$\min_{\omega, b, \xi, \eta, z} \frac{\|\omega\|^2}{2} + C_1 \sum_{i=1}^n \xi_i + C_2(\eta_1 + \eta_2) \tag{P4a}$$

$$\text{s.t. } y_i(\omega^\top x^i + b) \geq 1 - \xi_i, \quad i \in [1, n], \tag{P4b}$$

$$\omega^\top c^j + b \leq z_j M, \quad j \in [1, k], \tag{P4c}$$

$$\omega^\top c^j + b \geq -(1 - z_j)M, \quad j \in [1, k], \tag{P4d}$$

$$\tau - \eta_1 \leq \sum_{j=1}^k e_j z_j \leq \tau + \eta_2, \tag{P4e}$$

$$\xi_i \geq 0, \quad i \in [1, n], \tag{P4f}$$

$$\eta_1, \eta_2 \geq 0, \tag{P4g}$$

$$z_j \in \{0, 1\}, \quad j \in [1, k]. \tag{P4h}$$

A valid big- M is still given by (1) as shown in the next proposition.

Proposition 1 *If $e_j \geq 1$ for all $j \in [1, k]$, a valid big- M for Problem (P4) is given by (1).*

Proof The proof follows the same lines as the proofs of Lemma 1 and Theorem 2 with the additional observation that for all $j \in [1, k]$, it holds

$$\|c^j\| = \frac{1}{e_j} \left\| \sum_{i: x^i \in C_j} x^i \right\| \leq \frac{e_j \max_{i \in [n+1, N]} \|x^i\|}{e_j} = \max_{i \in [n+1, N]} \|x^i\|. \quad \square$$

It can happen that the hyperplane given by (ω^*, b^*) that results from the solution of Problem (P4) cuts through some cluster. This means that not all data points of the cluster actually lie on the same side of the hyperplane. If this happens, the solution of Problem (P4) does not satisfy the cardinality constraint (P3e) of Problem (P3). To fix this, we propose an iterative method that is formally listed in Algorithm 1. Note that the use of the k -means algorithm is helpful here as it automatically provides the convex hulls of the clusters. Hence, it is easy to check if the hyperplane cuts through some cluster or not.

If Algorithm 1 terminates it holds that all points in a cluster are on the same side of the final hyperplane. This implies the cardinality constraint (P3e) is satisfied. Note that the k -means algorithm is only called once to initialize the clustering. For all other iterations, we manually split clusters if they are cut by the hyperplane of the respective iteration and compute the new centroids directly.

The next theorem establishes that Algorithm 1 always terminates after finitely many iterations.

Algorithm 1: Re-Clustering Method (RCM)

Input: $X \in \mathbb{R}^{d \times N}$, $y \in \{-1, 1\}^n$, $k^1 \in \mathbb{N}$, $C_1 > 0$, $C_2 > 0$, and $\tau \in \mathbb{N}$.

- 1 Set $t \leftarrow 1$, compute M^t as in (1), compute a clustering of X_u in k^1 many clusters using the k -means algorithm, and obtain the centroids c^1, \dots, c^{k^1} as well as the numbers e_1, \dots, e_{k^1} of data points in each cluster.
- 2 Solve Problem (P4) to compute the hyperplane (ω^t, b^t) as well as ξ^t, η^t, z^t .
- 3 **if** the hyperplane (ω^t, b^t) cuts a cluster **then**
- 4 Set $k^{t+1} \leftarrow k^t$.
- 5 **for** each cluster that is cut by the hyperplane (ω^t, b^t) **do**
- 6 Split the cluster into two new clusters so that neither of the two new clusters is cut by the hyperplane (ω^t, b^t) .
- 7 Update the centroids of the newly created clusters.
- 8 Set $k^{t+1} \leftarrow k^{t+1} + 1$.
- 9 **end**
- 10 Update $t \leftarrow t + 1$ and go to Step 2.
- 11 **else**
- 12 Return the hyperplane (ω^t, b^t) as well as ξ^t, η^t, z^t .
- 13 **end**

Theorem 3 Suppose that $e_j \geq 1$ for all $j \in [1, k^1]$ after Step 1 of Algorithm 1. Then, Algorithm 1 terminates after at most $m - k^1$ iterations, where m is the number of the unlabeled data points and k^1 is the number of initial clusters.

Proof Observe that since we cluster m unlabeled points, the maximum number of clusters we can obtain is m . Besides that, if in an iteration t , Algorithm 1 does not terminate, at least one cluster is split Step 6. Because we start with k^1 clusters and since in each iteration, we increase the number of clusters at least by one, the maximum number of iterations is $m - k^1$. □

Note that the point obtained by Algorithm 1 is not necessarily a minimizer of Problem (P3). However, the objective function value of the point obtained by Algorithm 1 is an upper bound for the objective function value of Problem (P3).

Theorem 4 Let $(\bar{\omega}, \bar{b}, \bar{\xi}, \bar{\eta}, \bar{z})$ be the point returned by Algorithm 1. Then, $(\bar{\omega}, \bar{b}, \bar{\xi}, \bar{\eta}, \bar{z})$ is feasible for Problem (P3) with

$$M = 2\sqrt{2\bar{f}} \max_{i \in [1, N]} \|x^i\| + 1$$

and, consequently,

$$\bar{f} := \frac{\|\bar{\omega}\|^2}{2} + C_1 \sum_{i=1}^n \bar{\xi}_i + C_2(\bar{\eta}_1 + \bar{\eta}_2)$$

is an upper bound of Problem (P3).

Proof For all clusters \mathcal{C}_j , $j \in \{1, \dots, k^t\}$, where t is the final iteration of Algorithm 1, we set $\tilde{z}_i = \bar{z}_j$ for all i with $x^i \in \mathcal{C}_j$. We now show that $(\bar{\omega}, \bar{b}, \bar{\xi}, \bar{\eta}, \tilde{z})$ is a feasible

point for Problem (P3). Indeed, Constraints (P3b), (P3f), (P3g), and (P3h) are clearly fulfilled. Furthermore, since

$$\sum_{i \in \mathcal{C}_j} \tilde{z}_i = e_j \bar{z}_j$$

for all $j \in [1, k^t]$, using (P4e) we get

$$\sum_{i=n+1}^N \tilde{z}_i = \sum_{j=1}^{k^t} e_j \bar{z}_j \implies \tau - \bar{\eta}_1 \leq \sum_{i=n+1}^N \tilde{z}_i \leq \tau + \bar{\eta}_2$$

and Constraint (P3e) is satisfied. Besides that,

$$\frac{\|\bar{\omega}\|^2}{2} \leq \bar{f} \implies \|\bar{\omega}\| \leq \sqrt{2\bar{f}} \tag{2}$$

holds and as in Lemma 1, we get

$$|\bar{b}| \leq \|\bar{\omega}\| \max_{i \in [1, N]} \|x^i\| + 1. \tag{3}$$

Moreover, by construction, for all $i \in \{n + 1, \dots, N\}$ with $\tilde{z}_i = 1$, x^i belongs to a cluster \mathcal{C}_j such that $\bar{\omega}^\top c^j + \bar{b} \geq 0$. Using the fact that all points in \mathcal{C}_j are on the same side of the hyperplane, this side must be the positive one. This fact together with (2) and (3) implies

$$\begin{aligned} -(1 - \tilde{z}_i)M = 0 &\leq \bar{\omega}^\top x^i + \bar{b} \leq \|\bar{\omega}\| \max_{i \in [1, N]} \|x^i\| + |\bar{b}| \\ &\leq 2\sqrt{2\bar{f}} \max_{i \in [1, N]} \|x^i\| + 1 = M = \tilde{z}_i M. \end{aligned}$$

Similarly, for all $i \in \{n + 1, \dots, N\}$ with $\tilde{z}_i = 0$, we get

$$-M = -(1 - \tilde{z}_i)M \leq \bar{\omega}^\top x^i + \bar{b} \leq 0 = \tilde{z}_i M$$

and (P3c) as well as (P3d) are fulfilled. Because $(\bar{\omega}, \bar{b}, \bar{\xi}, \bar{\eta}, \bar{z})$ is a feasible point for Problem (P3), \bar{f} is an upper bound to the Problem (P3). \square

Note, finally, that since the point obtained from Algorithm 1 is feasible for Problem (P3), we can use it for warm starting.

4 Further algorithmic enhancements

In order to reduce computational costs, we propose two additional enhancements. The first one (see Sect. 4.1) makes use of the fact that the SVM is mostly influenced by

data points that are close to the separating hyperplane. The second one (see Sect. 4.2) introduces a rule for updating M in each iteration of Algorithm 1.

4.1 Handling points far from the hyperplane

In Algorithm 1, the number of clusters increases in each iteration. Hence, the time to solve Problem (P4) increases from iteration to iteration in general. Like in the original SVM, the points closest to the hyperplane influence the resulting hyperplane more than the other points. Obviously, eliminating points that do not strongly influence the hyperplane decreases the size of the problem. Some approaches to eliminate these points have also been proposed for the original SVM. For a survey, see, e.g., Birzhandi et al. (2002). However, most of these approaches are heuristics and do not necessarily yield a feasible point of the problem.

The idea for our setting is the following. Clusters that are far away from the hyperplane could be omitted as this will not change the solution. The farther a cluster is from the hyperplane in an iteration, the less likely it is that the cluster will be split or change sides completely in a future iteration. Hence, the clusters farthest from the current hyperplane mainly add information about their side and capacity. However, in a later iteration, the cluster may become relevant again. Thus, we need to find a way to discard detailed information on certain clusters but also a way to reactivate the discarded clusters if necessary.

We propose the following procedure to reduce the amount of clusters that have to be considered in the current iteration of the algorithm. If the number of clusters exceeds a fixed value k^+ , we first fix the cluster with the centroid farthest from the hyperplane as a kind of residual cluster on a side if this side has points far from the hyperplane. Second, we discard all clusters in which all points are farther from the hyperplane than some Δ^t and assign them to the residual cluster on their side of the hyperplane. This way the cardinality constraint remains valid. Moreover, all formerly discarded clusters are checked for re-consideration. If a discarded cluster has a point with a distance to the hyperplane less than Δ^t or if any point in the cluster changed the side, the cluster is reactivated.

Let $\bar{S} = (s_{\alpha(1)}, \dots, s_{\alpha(d)})^\top$ be the vector of increasingly sorted values of $S = \{s_1, \dots, s_d\}$ and let $a \in (0, 1)$. The a -quantile of S , as proposed by Hyndman and Fan (1996), is given by

$$P_S(a) := s_{\alpha(q)} + \frac{s_{\alpha(q)} - s_{\alpha(r)}}{q - r} ((d - 1)a - q + 1)$$

with

$$q := \max_{i \in [1, d]} \left\{ i : \frac{i - 1}{d - 1} \leq a \right\}, \quad r := \min_{i \in [1, d]} \left\{ i : \frac{i - 1}{d - 1} \geq a \right\}.$$

Given a parameter $\hat{\Delta}^t \in (0, 1)$, we choose Δ^t in each iteration t according to

$$\Delta^t = P_{D^t}(\hat{\Delta}^t) \quad \text{with} \quad D_j^t = \left| (\omega^t)^\top c_j + b^t \right| \quad \text{for all } j \in [1, k^t]. \tag{4}$$

Note that if in an iteration t , a point in some discarded cluster changed the side, the vector z as part of the current solution does not fit to this change. This happens when, e.g., $(\omega^{t-1})^\top x^i + b^{t-1} > 0$ and $(\omega^t)^\top x^i + b^t < 0$ but $z_j^t > 0$ with \mathcal{C}_j being the cluster with centroid farthest from the hyperplane on the positive side. To avoid that this happens too often, $\hat{\Delta}^{t+1}$ is increased by a fixed value $\tilde{\Delta} \in (0, 1)$ when there is some point in some discarded cluster that has changed sides.

Motivated by the above discussions, we add new steps in the Algorithm 1 that can be seen in Algorithm 2. In Step 5, if the number of clusters exceeds k^+ , clusters far from the hyperplane are discarded. In Steps 9 and 10, clusters discarded with a point that changed sides or that is closer to the hyperplane than Δ^t are reactivated. In Step 12, $\hat{\Delta}^t$ is updated.

Algorithm 2: Improved Re-Clustering Method (IRCM)

Input: $X \in \mathbb{R}^{d \times N}$, $y \in \{-1, 1\}^n$, $k^1 \in \mathbb{N}$, $C_1 > 0$, $C_2 > 0$, $\tau \in \mathbb{N}$, $\hat{\Delta}^1 \in (0, 1)$, $\tilde{\Delta} \in (0, 1)$, $\mathcal{G}^1 = \emptyset$, $k^+ \in \mathbb{N}$.

- 1 Set $t = 1$, compute M^t as in (1), cluster X_u in k^1 clusters using k -means, leading to centroids c^1, \dots, c^{k^1} and the numbers e_1, \dots, e_{k^1} of data points in each cluster.
- 2 Solve Problem (P4) to compute the hyperplane (ω^t, b^t) as well as ξ^t, η^t, z^t .
- 3 Compute Δ^t as in (4).
- 4 **if** $k^t > k^+$ **then**
- 5 | update $\mathcal{G}^{t+1} \leftarrow \mathcal{G}^t \cup \{\mathcal{C}_j : |(\omega^t)^\top x^\ell + b^t| > \Delta^t \forall x^\ell \in \mathcal{C}_j\}$.
- 6 **else**
- 7 | set $\mathcal{G}^{t+1} \leftarrow \mathcal{G}^t$.
- 8 **end**
- 9 Set $\mathcal{J}^t := \{\mathcal{C}_j \in \mathcal{G}^t : \exists x^\ell \in \mathcal{C}_j : \text{sign}((\omega^t)^\top x^\ell + b^t) \neq \text{sign}((\omega^{t+1})^\top x^\ell + b^{t+1})\}$.
- 10 Update $\mathcal{G}^{t+1} \leftarrow \mathcal{G}^{t+1} \setminus (\{\mathcal{C}_j \in \mathcal{G}^t : \exists x^\ell \in \mathcal{C}_j \text{ with } |(\omega^t)^\top x^\ell + b^t| \leq \Delta^t\} \cup \mathcal{J}^t)$.
- 11 **if** $\mathcal{J}^t \neq \emptyset$ **then**
- 12 | update $\hat{\Delta}^{t+1} \leftarrow \min\{\hat{\Delta}^t + \tilde{\Delta}, 1\}$.
- 13 **else**
- 14 | set $\hat{\Delta}^{t+1} \leftarrow \hat{\Delta}^t$
- 15 **end**
- 16 Compute M^{t+1} as in (8).
- 17 **if** $\mathcal{J}^t \neq \emptyset$ or the hyperplane (ω^t, b^t) cuts a cluster **then**
- 18 | Set $k^{t+1} \leftarrow k^t$.
- 19 | **for** each cluster that is cut by the hyperplane (ω^t, b^t) **do**
- 20 | | Split the cluster into two new clusters so that neither of the two new clusters is cut by the hyperplane (ω^t, b^t) .
- 21 | | Update the centroids of the newly created clusters.
- 22 | | Set $k^{t+1} \leftarrow k^{t+1} + 1$.
- 23 | **end**
- 24 | Update $t \leftarrow t + 1$ and back to Step 2.
- 25 **else**
- 26 | Return the hyperplane (ω^t, b^t) as well as ξ^t, η^t, z^t .
- 27 **end**

4.2 Updating the Big- M

As discussed in Sect. 2, M needs to be sufficiently large. However, the bigger the M , the more likely we face numerical issues. As shown in Sect. 2, the smaller the objective function provided by a feasible point, the smaller the value of M can be chosen. Based on that, we update M in each iteration with the aim of decreasing it. We do this by adding Step 16 in Algorithm 2 and the next theorem justifies this.

Theorem 5 Consider X, y, C_1, C_2, τ , as well as c^1, \dots, c^{k^t} and e_1, \dots, e_{k^t} in an iteration t of Algorithm 1. Then, the optimal solution $(\bar{\omega}^t, \bar{b}^t, \bar{\xi}^t, \bar{\eta}^t, \bar{z}^t)$ of Problem (P4) provides an upper bound

$$\tilde{f}_t := \frac{\|\bar{\omega}^t\|^2}{2} + C_1 \sum_{i=1}^n \bar{\xi}_i + C_2(\tilde{\eta}_1 + \tilde{\eta}_2), \tag{5}$$

with

$$\tilde{z}_j = \begin{cases} 1, & \text{if } (\bar{\omega}^t)^\top \tilde{c}_j + \bar{b}^t \geq 0, \\ 0, & \text{otherwise,} \end{cases} \tag{6}$$

and

$$\tilde{\eta}_1 = \max \left\{ 0, \tau - \sum_{j=1}^s e_j \tilde{z}_j \right\}, \quad \tilde{\eta}_2 = \max \left\{ 0, \sum_{j=1}^s e_j \tilde{z}_j - \tau \right\}, \tag{7}$$

for Problem (P4) with $c^1, \dots, c^{k^{t+1}}$ and $e_1, \dots, e_{k^{t+1}}$ as updated in iteration t with

$$M = 2\sqrt{2\tilde{f}_t} \max_{i \in [1, N]} \|x^i\| + 1. \tag{8}$$

Proof Consider \tilde{z} as given in (6) and $\tilde{\eta}_1, \tilde{\eta}_2$ as given in (7). We now show that $(\bar{\omega}^t, \bar{b}^t, \bar{\xi}^t, \tilde{z}, \tilde{\eta})$ is a feasible point for Problem (P4). Indeed, Constraints (P4b) and (P4e)–(P4h) are clearly satisfied. Moreover, $(\bar{\omega}^t, \bar{b}^t, \bar{\xi}^t, \bar{\eta}^t, \bar{z}^t)$ provides the objective function value given by (5) and

$$\|\bar{\omega}^t\| \leq \sqrt{2\tilde{f}_t}, \quad |\bar{b}^t| \leq \|\bar{\omega}^t\| \max_{i \in [1, N]} \|x^i\| + 1,$$

see the proof of Lemma 1. This together with $\|c^j\| \leq \max_{i \in [n+1, N]} \|x^i\|$ implies

$$(\bar{\omega}^t)^\top c^j + \bar{b}^t \leq \|\bar{\omega}^t\| \max_{i \in [n+1, N]} \|x^i\| + |\bar{b}^t| \leq 2\sqrt{2\tilde{f}_t} \max_{i \in [1, N]} \|x^i\| + 1 = M$$

and

$$(\bar{\omega}^t)^\top c^j + \bar{b}^t \geq -M.$$

Hence, Constraints (P4c) and (P4d) are satisfied. Since $(\bar{\omega}^t, \bar{b}^t, \bar{\xi}^t, \bar{z}, \bar{\eta})$ is a feasible point for Problem (P4), \tilde{f}_t is an upper bound for Problem (P4). \square

Using Theorem 5, we can update M in each iteration of Algorithm 2 as in (8). The following theorem establishes that as Algorithm 1, Algorithm 2 always terminates after finitely many iterations.

Theorem 6 *The Algorithm 2 terminates after at most*

$$2m - k^1 + \frac{(1 - \hat{\Delta}^1)}{\tilde{\Delta}}$$

iterations, where m is the number of unlabeled data points, k^1 is the number of initial clusters, and $\hat{\Delta}^1, \tilde{\Delta}$ are inputs of Algorithm 2.

Proof In Algorithm 2, the number of iterations can only be greater as in Algorithm 1 if there is some iteration t for which $\mathcal{J}^t \neq \emptyset$ holds but the hyperplane does not cut any cluster. At each iteration in which this happens, $\hat{\Delta}^t$ is increased and, in the worst case, i.e.,

$$\hat{t} := m - k^1 + \frac{(1 - \hat{\Delta}^1)}{\tilde{\Delta}},$$

we get $\hat{\Delta}^{\hat{t}} = 1$. This implies that for all further iterations t ,

$$\Delta^t = \max_{j \in [1, k^t]} |(\omega^t)^\top c^j + b^t|$$

holds. Thus, no cluster is added to the set \mathcal{G}^t . Since $|\mathcal{G}^{\hat{t}}| \leq m$ and $\mathcal{J}^t \subset \mathcal{G}^{\hat{t}}$, Algorithm 2 can only have m more iterations with $\mathcal{J}^t \neq \emptyset$. This means that the maximum number of iterations is $2m - k^1 + (1 - \hat{\Delta}^1)/\tilde{\Delta}$. \square

Although Theorem 6 shows that, in the worst case, Algorithm 2 can take more iterations than Algorithm 1 to terminate, Algorithm 2 solves problems with less binary variable in every iteration, which means that the time per iteration will be lower compared to Algorithm 1.

Note that the objective function value obtained by Algorithm 2 is an upper bound for the objective function value of Problem (P3).

Theorem 7 *Let $(\bar{\omega}, \bar{b}, \bar{\xi}, \bar{\eta}, \bar{z})$ be the point returned by Algorithm 2. Then, $(\bar{\omega}, \bar{b}, \bar{\xi}, \bar{\eta}, \bar{z})$ is feasible for Problem (P3) with*

$$M = 2\sqrt{2\bar{f}} \max_{i \in [1, N]} \|x^i\| + 1$$

and, consequently,

$$\bar{f} := \frac{\|\bar{\omega}\|^2}{2} + C_1 \sum_{i=1}^n \bar{\xi}_i + C_2(\bar{\eta}_1 + \bar{\eta}_2)$$

is an upper bound of Problem (P3).

Proof Since Algorithm 2 terminates when no cluster changes the side and no cluster is cut by the hyperplane, the proof is the same as for Theorem 5. \square

As before, we can use the point obtained from Algorithm 2 to warm start Problem (P3).

5 Using IRCM for warm-starting

As stated in Theorem 7, the solution found by Algorithm 2 is feasible for Problem (P3). Hence, we can use it for warm-starting the solution process of Problem (P3). The next lemma establishes that unlabeled points can be fixed to be in one side of the hyperplane.

Lemma 8 *Let $(\bar{\omega}, \bar{b}, \bar{\xi}, \bar{\eta}, \bar{z})$ be a feasible point of Problem (P3) with objective function value \bar{f} . Furthermore, let $(\omega^*, b^*, \xi^*, \eta^*, z^*)$ be an optimal solution of Problem (P3) with objective function value f^* . Set*

$$P_u := \left\{ i \in [n + 1, N] : (\omega^*)^\top x^i + b^* > 0 \right\},$$

$$N_u := \left\{ i \in [n + 1, N] : (\omega^*)^\top x^i + b^* < 0 \right\},$$

and let $S_p \subseteq P_u, S_n \subseteq N_u$ be arbitrarily chosen subsets and let $x^s \notin S_n$ be an unlabeled point with $\bar{\omega}^\top x^s + \bar{b} < 0$. Then, the objective function value \tilde{f} given by any feasible point of the problem

$$\min_{\omega, b, \xi, \eta, z} \frac{\|\omega\|^2}{2} + C_1 \sum_{i=1}^n \xi_i + C_2(\eta_1 + \eta_2) \tag{P5a}$$

$$s.t \quad y_i(\omega^\top x^i + b) \geq 1 - \xi_i, \quad i \in [1, n], \tag{P5b}$$

$$\omega^\top x^i + b \leq z_i M, \quad i \in [n + 1, N] \setminus (\{s\} \cup S_p \cup S_n), \tag{P5c}$$

$$\omega^\top x^i + b \geq -(1 - z_i)M, \quad i \in [n + 1, N] \setminus (\{s\} \cup S_p \cup S_n), \tag{P5d}$$

$$\omega^\top x^i + b \geq 0, \quad i \in S_p, \tag{P5e}$$

$$\omega^\top x^i + b \leq 0, \quad i \in S_n, \tag{P5f}$$

$$0 \leq \omega^\top x^s + b \leq z_s M, \tag{P5g}$$

$$\tau - \eta_1 \leq |S_p| + \sum_{i \in [n+1, N] \setminus (S_p \cup S_n)} z_i \leq \tau + \eta_2, \tag{P5h}$$

$$\xi_i \geq 0, \quad i \in [1, n], \tag{P5i}$$

$$\eta_1, \eta_2 \geq 0, \tag{P5j}$$

$$z_i \in \{0, 1\}, \quad i \in [n + 1, N] \setminus (S_p \cup S_n), \tag{P5k}$$

with M as defined in (8), satisfies the following properties:

- (a) \tilde{f} is an upper bound for f^* ,

- (b) if \tilde{f} is the optimal objective function value of Problem (P5) and $\bar{f} < \tilde{f}$ is satisfied, it holds $(\omega^*)^\top x^s + b^* < 0$, i.e., $x^s \in N_u$.

Proof (a) The points that satisfy Constraints (P5b)–(P5k) are feasible for Problem (P3) and provide an objective function value \tilde{f} . Since f^* is the optimal objective function value of Problem (P3), $f^* \leq \tilde{f}$ holds.

- (b) Consider by contradiction that $(\omega^*)^\top x^s + b^* \geq 0$ holds. This means that $(\omega^*, b^*, \xi^*, \eta^*, z^*)$ satisfies (P5b)–(P5k). Moreover, since \tilde{f} is the objective function for Problem (P5), we get $f^* = \tilde{f}$. However, $f^* \leq \bar{f}$ holds. Thus,

$$f^* \leq \bar{f} < \tilde{f} = f^*$$

yields a contradiction. □

Note that the last lemma can be adapted for the case $\bar{\omega}^\top x^s + \bar{b} > 0$. In this case, the constraints (P5g) need to be replaced with

$$-(1 - z_s)M \leq \omega^\top x^s + b \leq 0 \tag{9}$$

and (b) needs to be replaced with $(\omega^*)^\top x^s + b^* > 0$, i.e., $x^s \in P_u$. Note that the more points we have fixed on one side, the solution of Problem (P3) tends to be faster as there are fewer binary variables.

Moreover, the solution of Problem (P3) can be found by solving the problem

$$\min_{\omega, b, \xi, \eta, z} \frac{\|\omega\|^2}{2} + C_1 \sum_{i=1}^n \xi_i + C_2(\eta_1 + \eta_2) \tag{P6a}$$

$$\text{s.t. } y_i(\omega^\top x^i - b) \geq 1 - \xi_i, \quad i \in [1, n], \tag{P6b}$$

$$\omega^\top x^i + b \leq z_i M, \quad i \in [n + 1, N] \setminus (S_p \cup S_n), \tag{P6c}$$

$$\omega^\top x^i + b \geq -(1 - z_i)M, \quad i \in [n + 1, N] \setminus (S_p \cup S_n), \tag{P6d}$$

$$\omega^\top x^i + b \geq 0, \quad i \in S_p, \tag{P6e}$$

$$\omega^\top x^i + b \leq 0, \quad i \in S_n, \tag{P6f}$$

$$\tau - \eta_1 \leq |S_p| + \sum_{i \in [n+1, N] \setminus (S_p \cup S_n)} z_i \leq \tau + \eta_2, \tag{P6g}$$

$$\xi_i \geq 0, \quad i \in [1, n], \tag{P6h}$$

$$\eta_1, \eta_2 \geq 0 \tag{P6i}$$

$$z_i \in \{0, 1\}, \quad i \in [n + 1, N] \setminus (S_p \cup S_n), \tag{P6j}$$

where S_p and S_n are subsets of P_u and N_u , respectively.

Based on these results, we propose the following. We compute the point $(\bar{\omega}, \bar{b}, \bar{\xi}, \bar{\eta}, \bar{z})$ using Algorithm 2, leading to an objective function value \bar{f} for Problem (P3). Afterward, we sort the indices $i \in \{n + 1, N\}$, indicated by the permutation $\alpha: \{n + 1, N\} \rightarrow \{n + 1, N\}$, so that $|\bar{\omega}^\top x^{\alpha(i)} + \bar{b}| \geq |\bar{\omega}^\top x^{\alpha(i)+1} + \bar{b}|$ holds.

Consider now a given and fixed parameter B_{\max} , a factor $\gamma \in (1, m/B_{\max}]$, and let β be γB_{\max} rounded to the next integer. While the number of fixed points is smaller than B_{\max} , we do the following. For $i \in \{1, \dots, \beta\}$, if $\bar{\omega}^\top x^{\alpha(i)} + \bar{b} < 0$ holds, we try to solve Problem (P5) using the limit time of T_{\max} and the upper bound \bar{f} . If there is a feasible point of this problem, we set $(\bar{\omega}, \bar{b}, \bar{\xi}, \bar{\eta}, \bar{z})$ to this point and update the objective function value \bar{f} accordingly. If no feasible point could be computed and if the limit time was not reached, we fix x^s to be in the negative side.

Similarly, we do the same if $\bar{\omega}^\top x^{d_i} + \bar{b} > 0$ holds with (P5g) replaced by (9). The method is formally described in Algorithm 3. Finally note that, although Problem (P5) is an MIQP, it is a feasibility problem, which is often easier to solve than an optimization problem in practice. Besides that, if the point obtained from Algorithm 2 is close to the optimum of Problem (P3), many unlabeled points will be fixed and Problem (P3) will be faster to solve.

Algorithm 3: Improved & Warm-Started Re-Clustering Method (WIRCM)

Input: $X \in \mathbb{R}^{d \times N}$, $y \in \{-1, 1\}^n$, $k^1 \in \mathbb{N}$, $C_1 > 0$, $C_2 > 0$, $\tau \in \mathbb{N}$, $\hat{\Delta}^1 \in (0, 1)$, $\tilde{\Delta} \in (0, 1)$, $G^1 = \emptyset$, $k^+ \in \mathbb{N}$, $T_{\max} > 0$, $B_{\max} \in \mathbb{N}$, and $\gamma \in (1, m/B_{\max}]$.

- 1 Compute the hyperplane $(\bar{\omega}, \bar{b})$ and $\bar{\xi}, \bar{\eta}, \bar{z}$ using Algorithm 2, leading to the objective function value \bar{f} . Let M be the last M^t of Algorithm 2.
- 2 Sort the indices $i \in \{n + 1, N\}$ such that $|\bar{\omega}^\top x^{\alpha(i)} + \bar{b}| \geq |\bar{\omega}^\top x^{\alpha(i)+1} + \bar{b}|$ holds and set β to be γB_{\max} rounded to the next integer.
- 3 **for** $i \in \{1, \dots, \beta\}$ **do**
- 4 **if** $|S_p| + |S_n| \leq B_{\max}$ **then**
- 5 **if** $\bar{\omega}^\top x^s + \bar{b} < 0$ **then**
- 6 Solve Problem (P5) with upper bound \bar{f} and a time limit T_{\max} .
- 7 **if** there is a feasible point **then**
- 8 | update $(\bar{\omega}, \bar{b}, \bar{\xi}, \bar{\eta}, \bar{z})$, and \bar{f}
- 9 **else if** T_{\max} was not reached **then**
- 10 | $S_n \leftarrow S_n \cup \{s\}$
- 11 **end**
- 12 **else if** $\bar{\omega}^\top x^s + \bar{b} > 0$ **then**
- 13 Solve the problem (P5) with (P5g) replaced by (9), using \bar{f} as an upper bound and a time limit of T_{\max} .
- 14 **if** there is a feasible point **then**
- 15 | update $(\bar{\omega}, \bar{b}, \bar{\xi}, \bar{\eta}, \bar{z})$, and \bar{f}
- 16 **else if** T_{\max} was not reached **then**
- 17 | $S_p \leftarrow S_p \cup \{s\}$
- 18 **end**
- 19 **end**
- 20 **end**
- 21 **end**
- 22 Compute the solution $(\omega^*, b^*, \xi^*, \eta^*, z^*)$ of Problem (P6) with $(\bar{\omega}, \bar{b}, \bar{\xi}, \bar{\eta}, \bar{z})$ value and \bar{f} as an upper bound.

6 Numerical results

In this section, we present and discuss our computational results that illustrate the benefits of knowing the total amount of each class of unlabeled data and of using our approaches to speed up the solution process. We evaluate this on different test sets from the literature. The test sets are described in Sect. 6.1, while the computational setup is depicted in Sect. 6.2. The evaluation criteria are described in Sect. 6.3 and the numerical results are discussed in Sect. 6.4.

6.1 Test sets

For the computational analysis of the proposed approaches, we consider the subset of instances presented by Olson et al. (2017) that are suitable for classification problems and that have at most three classes. We restrict ourselves to instances of at most three classes to obtain an overall test set of manageable size. Repeated instances are removed and instances with missing information are reduced to the observations without missing information. If three classes are given in an instance, we transform them into two classes such that the class with label 1 represents the positive class, and the other two classes represent the negative class. This results in a final test set of 97 instances; see Table 1 in “Appendix A”.

To avoid numerical instabilities, we re-scale all data sets as follows. For each coordinate $j \in [1, d]$, we compute

$$l_j = \min_{i \in [1, N]} \{x_j^i\}, \quad u_j = \max_{i \in [1, N]} \{x_j^i\}, \quad m_j = 0.5 (l_j + u_j)$$

and shift each coordinate j of all data points x^i via $\bar{x}_j^i = x_j^i - m_j$. If we do this for all data points, they get centered around the origin. Moreover, if a coordinate j of the re-scaled points is still large, i.e., if $\tilde{l}_j = l_j - m_j < -10^2$ or $\tilde{u}_j = u_j - m_j > 10^2$ holds, it is re-scaled via

$$\tilde{x}_j^i = (\bar{v} - \underline{v}) \frac{\bar{x}_j^i - \tilde{l}_j}{\tilde{u}_j - \tilde{l}_j} + \bar{v},$$

with $\bar{v} = 10^2$ and $\underline{v} = -10^2$. The corresponding 29 instances that we re-scaled are marked with an asterisk in Table 1. Note that we use a linear transformation to scale the datasets. Hence, after computing the hyperplane for the scaled data, the respective hyperplane for the original data can also be computed ex post by applying another suitably chosen linear transformation as well.

In our computational study, we want to highlight the importance of cardinality constraints, especially for the case of non-representative biased samples. Biased samples occur frequently in non-probability surveys, which are surveys for which the inclusion process is not monitored and, hence, the inclusion probabilities are unknown as well. Correction methods like inverse inclusion probability weighting are therefore

not applicable. For an insight into inverse inclusion probability weighting, see Skinner and D'arrigo (2011) and references therein.

To mimic this situation, we create 5 biased samples with 10 % of the data being labeled for each instance. Different from a simple random sample in which each point has an equal probability of being chosen as labeled data, in the biased sample, the labeled data is chosen with probability 85 % for being on the positive side of the hyperplane. Then, for each instance, with a time limit of 3600 s, we apply the approaches listed in Sect. 6.2. In Appendix C, we also provide the results under simple random sampling, which produces unbiased samples. We see that the results from the proposed methods are similar to the plain SVM in that setting. Hence, besides the additional computational burden, there is no downside to use the proposed method in case of an unknown sampling process.

6.2 Computational setup

Our algorithm has been implemented in Julia 1.8.5 and we use Gurobi 9.5.2 and JuMP (Dunning et al. 2017) to solve Problem (P1), (P3), and (P4). All computations were executed on the high-performance cluster “Elwetritsch”, which is part of the “Alliance of High-Performance Computing Rheinland-Pfalz” (AHRP). We used a single Intel XEON SP 6126 core with 2.6 GHz and 64 GB RAM.

For each one of the 485 instances described in Sect. 6.1, the following approaches are compared:

- (a) SVM as given in Problem (P1), where only labeled data are considered;
- (b) CS³VM as given in Problem (P3) with M as given in (1);
- (c) IRCM as described in Algorithm 2;
- (d) WIRCM as described in Algorithm 3.

Based on our preliminary experiments, we set the penalty parameters $C_1 = C_2 = 1$. For WIRCM, we impose a time limit for solving Problem (P5) of $T_{\max} = 40$ s. Moreover, we choose $\gamma = 1.2$ and the maximum number B_{\max} of unlabeled points that can be fixed as

$$B_{\max} = \begin{cases} 0.2m, & \text{if } m \in [1, 100], \\ 0.25m, & \text{if } m \in (100, 500], \\ 0.35m, & \text{if } m \in (500, 1000], \\ 0.45m, & \text{otherwise.} \end{cases}$$

Finally, for IRCM and WIRCM, we set $\hat{\Delta}^1 = 0.8$, $\tilde{\Delta} = 0.1$, $k^+ = 50$, and the initial number of clusters is set to

$$k^1 = \begin{cases} 10, & \text{if } m \in [1, 500], \\ 20, & \text{if } m \in (500, 1000], \\ 50, & \text{otherwise.} \end{cases}$$

A more detailed discussion of the choice of hyperparameters is given in Appendix D.

6.3 Evaluation criteria

The first evaluation criterion is the run time of SVM, CS³VM, IRCM, and WIRCM. The results will help to contextualize other evaluation criteria such as accuracy and precision. To compare run times, we use empirical cumulative distribution functions (ECDFs). Specifically, for S being a set of solvers (or approaches as above) and for P being a set of problems, we denote by $t_{p,s} \geq 0$ the run time of approach $s \in S$ applied to problem $p \in P$ in seconds. If $t_{p,s} > 3600$, we consider problem p as not being solved by approach s . With these notations, the performance profile of approach s is the graph of the function $\gamma_s: [0, \infty) \rightarrow [0, 1]$ given by

$$\gamma_s(\sigma) = \frac{1}{|P|} |\{p \in P: t_{p,s} \leq \sigma\}|. \quad (10)$$

The second evaluation criterion is based on Theorem 5, where we show that the objective function value of the point obtained by IRCM is an upper bound for CS³VM, and consequently for Problem (P4) that is solved with WIRCM. Note that SVM also provides a feasible point for CS³VM and, consequently, provides an upper bound as well. Consider (ω, b, ξ) the solution of SVM, we compute the binary variables z_i , $i \in [n+1, N]$ as follows:

$$z_i = \begin{cases} 1, & \text{if } \omega^\top x^i + b > 0, \\ 0, & \text{if } \omega^\top x^i + b < 0. \end{cases}$$

If $\omega^\top x^i + b = 0$ for some x^i , we set

$$z_i = \begin{cases} 1, & \text{if } \sum_{j \in [n+1, N]: \omega^\top x^j + b \neq 0} z_j \leq \tau, \\ 0, & \text{otherwise.} \end{cases}$$

Finally, we set

$$\eta_1 = \max \left\{ 0, \tau - \sum_{i=n+1}^N z_i \right\}, \quad \eta_2 = \max \left\{ 0, \sum_{i=n+1}^N z_i - \tau \right\},$$

and the objective function value can be computed as

$$\frac{\|\omega\|^2}{2} + C_1 \sum_{i=1}^n \xi_i + C_2(\eta_1 + \eta_2).$$

Based on that, we compare how close the objective function values obtained from SVM, CS³VM, IRCM, and WIRCM are to the optimal solution. To this end, we use ECDFs, for which we replace $t_{p,s}$ by $f_{p,s}$ in Eq. (10) with

$$f_{p,s} := \frac{b_{p,s} - f_p^*}{f_p^*}, \tag{11}$$

where f_p^* is the optimal objective function value of problem p and $b_{p,s}$ is the objective function value obtained by approach s .

Besides that, for each instance and for each approach described in Sect. 6.2, after computing the hyperplane (ω, b) , we classify all points x^i as being on the positive side if $\omega^\top x^i + b > 0$ and as being on the negative side if $\omega^\top x^i + b < 0$ holds. For CS³VM and WIRCM, if the hyperplane (ω, b) satisfies $\omega^\top x^i + b = 0$ for some unlabeled point x^i , we classify this point as positive or negative depending on the respective binary variable z_i . On the other hand, for IRCM, if $\omega^\top x^i + b = 0$ for some unlabeled point x^i , we classify this point as positive or negative depending on z_j with j so that $x^i \in \mathcal{C}_j$. For the labeled points in these three approaches and for all points in the SVM, if $\omega^\top x^i + b = 0$ holds, we classify the point on the correct side. Note that for the cases in which the IRCMs take more than 3600 s to solve the instance, we use the last hyperplane found by the algorithm. If we hit the time limit in Gurobi when solving CS³VM (either standalone or in the final phase of the WIRCM), we take the best solution found so far.

Knowing the true label of all points, we then distinguish all points in four categories: true positive (TP) or true negative (TN) if the point is classified correctly in the positive or negative class, respectively, as well as false positive (FP) if the point is misclassified in the positive class and as false negative (FN) if the point is misclassified in the negative class. Based on that we compute two classification metrics, for which a higher value indicates a better classification. The first one is accuracy (AC). It measures the proportion of correctly classified points and is given by

$$AC := \frac{TP + TN}{TP + TN + FP + FN} \in [0, 1]. \tag{12}$$

The second metric is precision (PR). It measures the proportion of correctly classified points among all positively classified points and is computed by

$$PR := \frac{TP}{TP + FP} \in [0, 1]. \tag{13}$$

The main comparison in terms of accuracy and precision is w.r.t. the “true hyperplane”, i.e., the solution of Problem (P1) on the complete data with all N points and all labels available. The main question is how close the accuracy and precision is to the one of the true hyperplane. Hence, we compute the ratios of the accuracy and precision according to

$$\widehat{AC} := \frac{AC}{AC_{\text{true}}}, \quad \widehat{PR} := \frac{PR}{PR_{\text{true}}}, \tag{14}$$

where AC_{true} and PR_{true} are computed as in Eqs. (12) and (13) for the true hyperplane.

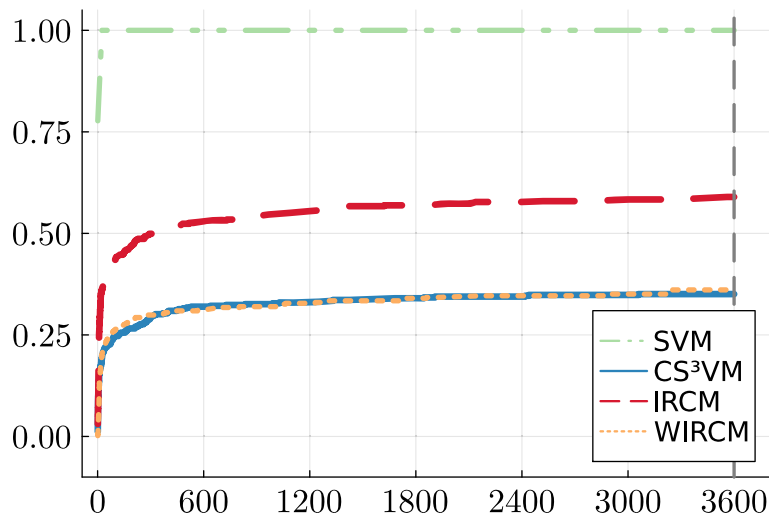


Fig. 2 ECDFs for run time (in seconds)

We also compare the measures with the SVM method, which only considers the information of the labeled data. For this purpose, we compute

$$\overline{AC} := \frac{AC - AC_{SVM}}{AC_{SVM}}, \quad \overline{PR} := \frac{PR - PR_{SVM}}{PR_{SVM}}, \quad (15)$$

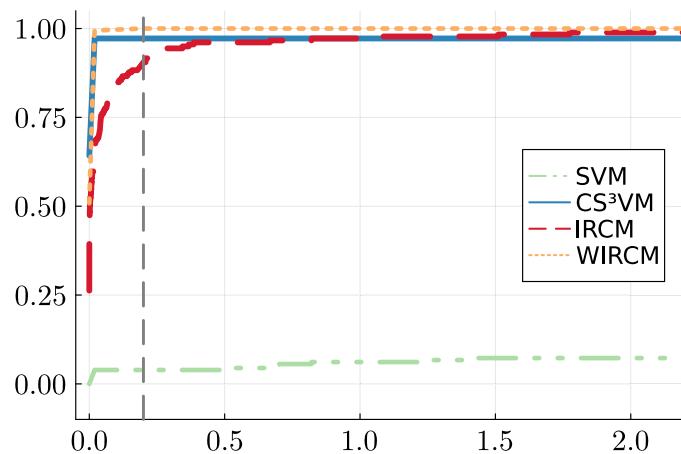
where AC_{SVM} and PR_{SVM} are computed as in (12) and (13) for the SVM hyperplane. To keep the numerical results section concise, we report on recall and the false positive rate in Appendix B.

6.4 Numerical results

6.4.1 Run time

Figure 2 shows the ECDFs for the measured run times. Clearly, SVM is the fastest algorithm. This is expected as the SVM does not include any binary variables related to the unlabeled points, which is in contrast to other approaches. It can be seen that the IRCM outperforms both CS^3VM and WIRCM. This shows that the idea to cluster unlabeled data points significantly decreases the run time. However, we need to be careful with the interpretation of these run times since termination of SVM and IRCM does not imply that a globally optimal point is found, whereas this is guaranteed CS^3VM and the WIRCM. The quality of the points found by SVM and IRCM will be discussed in the next section. The figure also clearly indicates that Problem (P2) is rather challenging: Even IRCM, which terminates for the most instances within the time limit (indicated by the gray and dashed vertical line) only does so for 57% of the instances. Note that the WIRCM has the worst efficiency. This obviously needs to be the case since due to Step 1 of Algorithm 3, its run time always includes the run time of the IRCM. To shed some light on the scalability of the approaches, we also present a brief analysis of the run times in dependence of the number of samples in Appendix E.

Fig. 3 ECDFs for the quality of the obtained upper bounds



6.4.2 Quality of the obtained upper bounds

As discussed in the last section, for some instances none of the three approaches that actually consider the unlabeled data terminate within the given time limit. This means we do not obtain the optimal objective function value for these instances, which we, moreover, can only provably obtain by CS³VM and the WIRCM. In fact, we have the optimal solution for 179 instances. These are the baseline instances for Fig. 3, which shows the ECDFs for the upper bound quality, as defined in (11). Note that the objective function value obtained by SVM is very far from the optimal value, while the IRCM finds an objective function value rather close to the optimal value (with $f_{ps} \leq 0.2$, see the gray dashed vertical line) in 90 % of these instances. Besides that, the WIRCM outperforms CS³VM in this comparison, which means using the IRCM as a warm start improves the result.

The consequences of the results so far are the following. If one is interested in getting a rather good feasible point as quickly as possible, one should use the IRCM. If one is able to spend some more run time, one should use the WIRCM. Hence, both novel methods derived in this paper have their advantage over just solving the CS³VM with a standard MIQP solver.

6.4.3 Accuracy

For some instances, none of the three approaches that actually tackle the unlabeled data terminate within the given time limit. Hence, our first comparison only considers instances for which CS³VM terminates within the time limit.

As can be seen in Fig. 4, the relative accuracy \widehat{AC} (w.r.t. the true hyperplane) of CS³VM, is closer to 1 than the relative accuracy of SVM—especially for the unlabeled data. This means that using the unlabeled points as well as the cardinality constraint allows to re-produce the classification of the true hyperplane with higher accuracy than the standard SVM does. Besides that, the relative accuracy of the SVM is more spread than the one of the other approaches, indicating that there is comparable more variation in the results as compared the results of CS³VM. The box in the boxplot depicts the range of the medium 50 % of the values; 25 % of the values are below and 25 % are above the box.

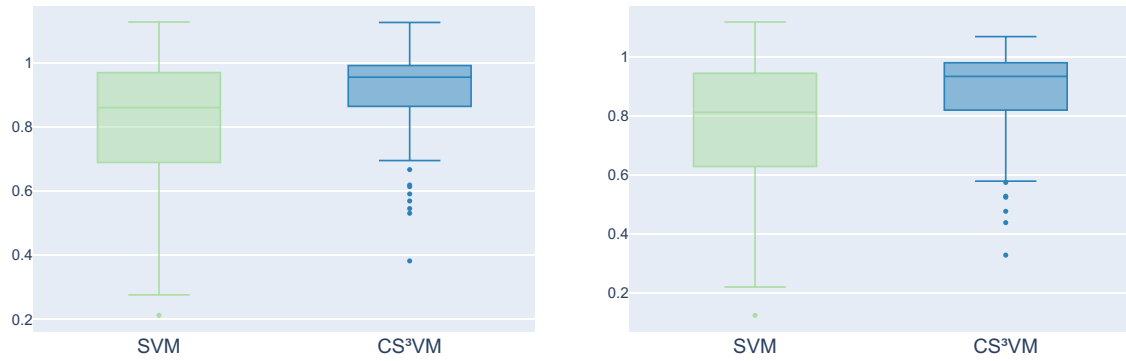


Fig. 4 Relative accuracy \widehat{AC} w.r.t. the true hyperplane; see (14). Only those instances are considered for which CS^3VM terminated. Left: Comparison for all data points. Right: Comparison only for unlabeled data points

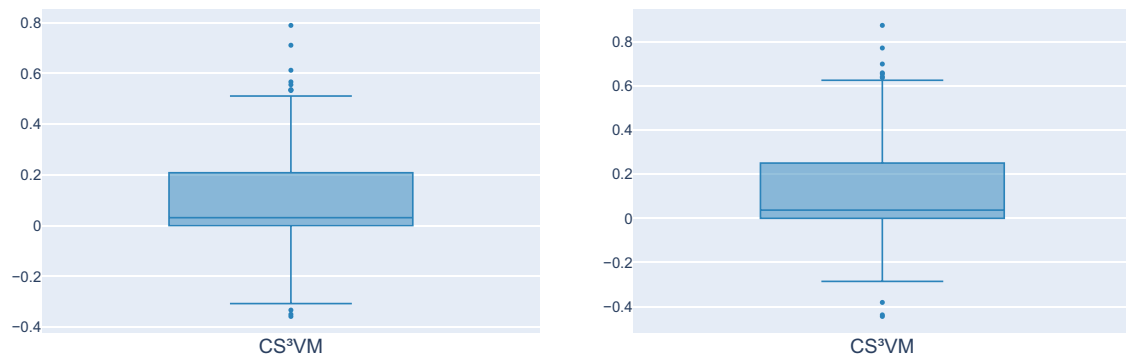


Fig. 5 Accuracy values \overline{AC} w.r.t. the SVM; see (15) only consider the instances that CS^3VM terminated. Left: Comparison for all data points. Right: Comparison only for unlabeled data points

Figure 5 shows that, in almost 75% of the cases, CS^3VM , has \overline{AC} values larger than zero, where zero means the same accuracy as the SVM itself. In the others 25% of the cases, the \overline{AC} of CS^3VM is slightly smaller than SVM.

The second comparison considers only those three approaches that actually consider the unlabeled data, i.e., CS^3VM , IRCM, and WIRCM for all instances. As can be seen in Fig. 6, even though IRCM does not have an optimality guarantee, it has a better relative accuracy \widehat{AC} than the hyperplane obtained from CS^3VM within the time limit. Consequently, as the hyperplane obtained from IRCM is used as a warm-start in WIRCM, it also has better accuracy.

Figure 7 shows that, in almost 75% of the cases, CS^3VM , the IRCM, and the WIRCM have \overline{AC} values larger than zero. That is, in general, our methods have greater accuracy than the SVM. Though, some cases indicate worse \overline{AC} values for our methods than for the SVM. This happens because for some instances, the methods (mainly for CS^3VM ; see also Fig. 2) do not terminate within the time limit. Hence, we expect that the number of negative values will decrease if we would increase the time limit.

6.4.4 Precision

We again separate the comparisons as in Sect. 6.4.3. Figure 8 shows that the SVM's relative precision \widehat{PR} is lower than the relative precision of CS^3VM . This means that

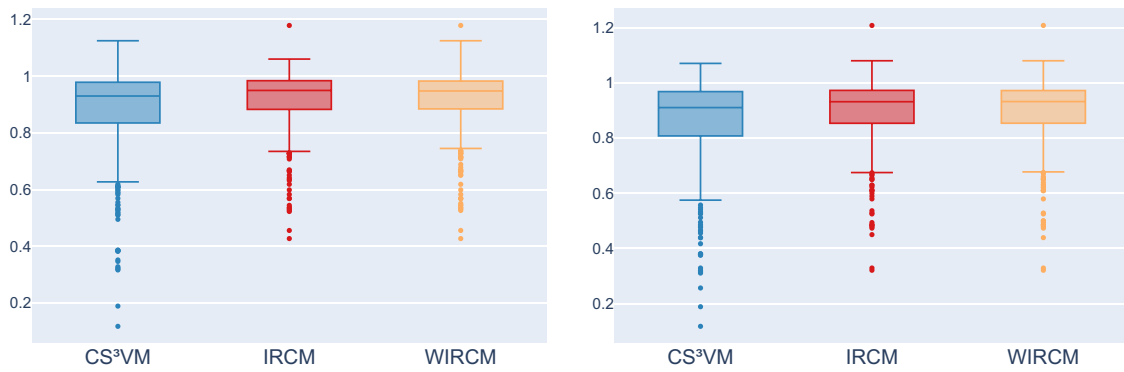


Fig. 6 Relative accuracy \widehat{AC} w.r.t. the true hyperplane; see (14). Left: Comparison for all data points. Right: Comparison only for unlabeled data points

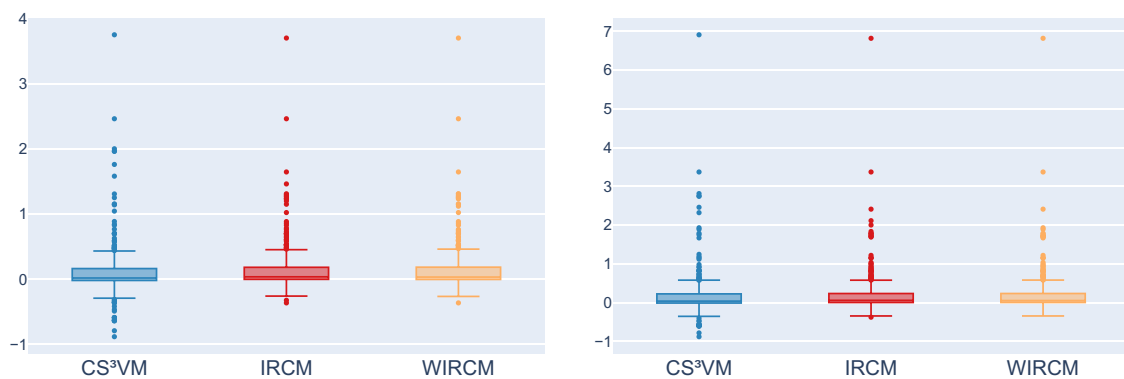


Fig. 7 Accuracy values \overline{AC} w.r.t. the SVM; see (15) consider all instances. Left: Comparison for all data points. Right: Comparison only for unlabeled data points

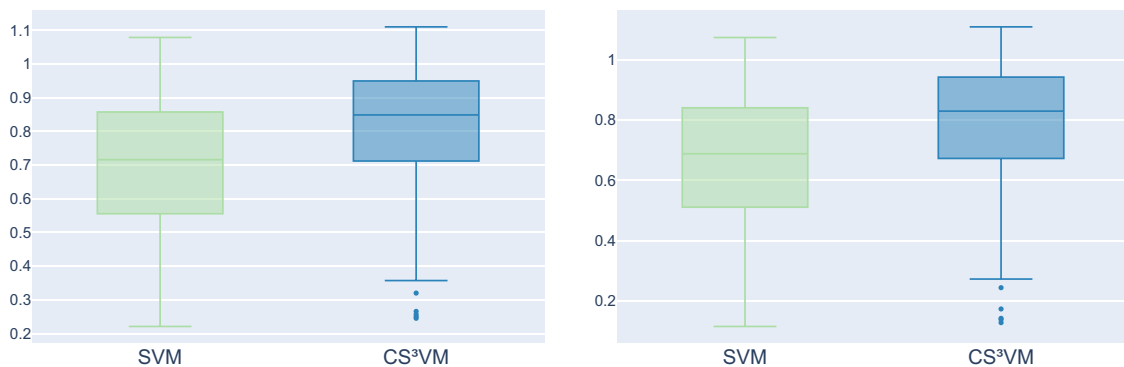


Fig. 8 Relative precision \widehat{PR} w.r.t. the true hyperplane as; see (14). Only those instances are considered for which CS^3VM terminated. Left: Comparison for all data points. Right: Comparison only for unlabeled data points

CS^3VM re-produces the classification of the true hyperplane with higher precision than the original SVM. Hence, SVM has more false-positive results. This happens because the biased sample is more likely to have positively labeled data and due to having no information about the unlabeled data, the SVM ends up classifying points on the positive side. As can be seen in Fig. 9, CS^3VM has slightly higher \overline{PR} values than 0, which is the baseline here that refers to the SVM itself. This means, CS^3VM is slightly more precise than the SVM.

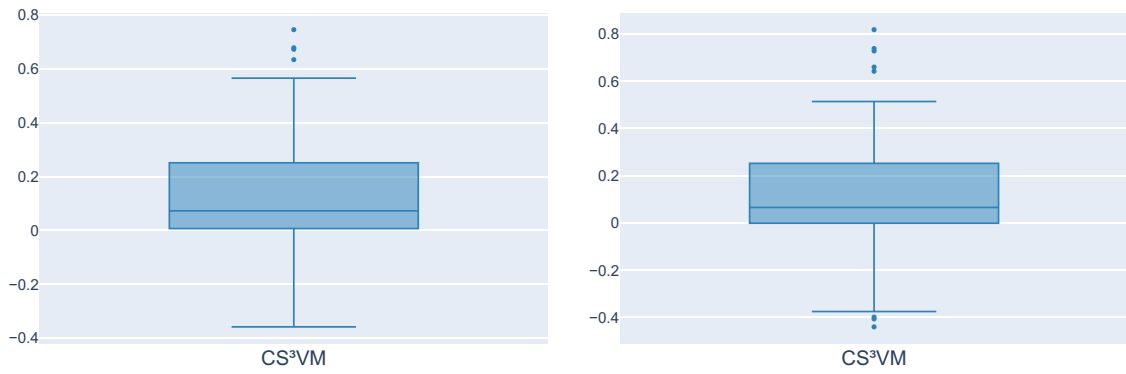


Fig. 9 Precision values \overline{PR} w.r.t. the SVM; see (15). Only those instances are considered for which CS^3VM terminated. Left: Comparison for all data points. Right: Comparison only for unlabeled data points

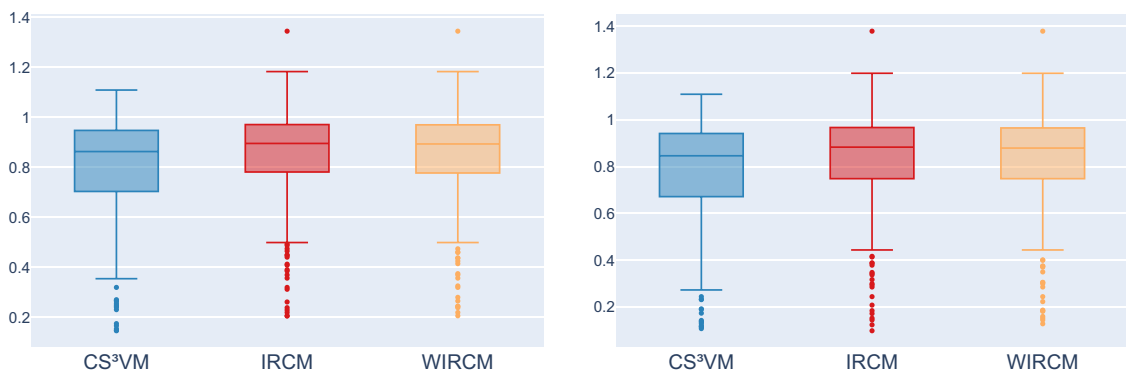


Fig. 10 Relative precision \widehat{PR} w.r.t. the true hyperplane as; see (14). Left: Comparison for all data points. Right: Comparison only for unlabeled data points

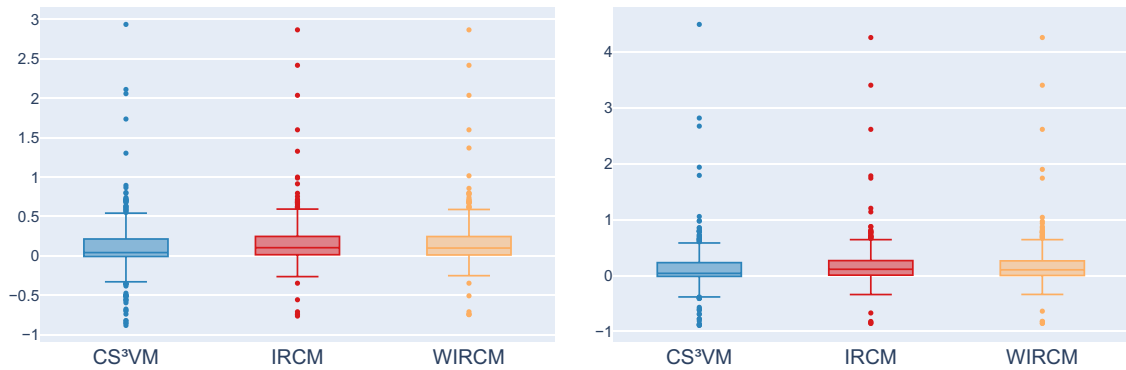


Fig. 11 Precision values \overline{PR} w.r.t. the SVM; see (15). Left: Comparison for all data points. Right: Comparison only for unlabeled data points

Figure 10 shows that the \widehat{PR} values of the IRCM and the WIRCM are less spread than the ones of CS^3VM . The reason most likely is that the CS^3VM approach terminates on fewer instances than the IRCM and the WIRCM. As can be seen in Fig. 11, the IRCM and the WIRCM also have slightly higher \overline{PR} values than 0. This means that our methods are slightly more precise than the SVM. The negative outliers most likely are due to the same reason as those for the respective accuracy values.

7 Conclusion

For many classification problems, it can be costly to obtain labels for the entire population of interest. However, aggregate information on how many points are in each class can be available from external sources. For this situation, we proposed a semi-supervised SVM that can be modeled via a big- M -based MIQP formulation. We also presented a rule for updating the big- M in an iterative re-clustering method and derived further computational techniques such as tailored dimension reduction and warm-starting to reduce the computational cost.

In case of simple random samples, our proposed semi-supervised methods perform as good as the classic SVM approach. However, in many applications, the available data is coming from non-probability samples. Hence, there is the risk of obtaining biased samples. Our numerical study shows that our approaches have better accuracy and precision than the original SVM formulation in this setting.

The problem of considering a cardinality constraint is computationally challenging. Our proposed clustering approach significantly helps to decrease the run time and to find an objective function value that is very close to the optimal value. Besides that, the clustering approach maintains the same accuracy and precision as the MIQP formulation. Moreover, using the clustering approach as a warm-start and fixing some unlabeled points on one side of the hyperplane helps to improve the quality of the objective function value again. Hence, the newly proposed methods lead to a significant improvement compared to just solving the classic MIQP formulation using a standard solver.

Despite these contributions, there is still room for improvement and future work. First, we only considered the linear SVM kernel. For future work, the development of methods for other kernels, such as a Gaussian kernel, can be a valuable topic. Moreover, the use of other norms than the 2-norm could be analyzed as well and the formal hardness of the considered problem should be settled. Finally, the adaptation of our approaches for multiclass SVMs using a one-vs.-rest strategy may be another reasonable future work.

Appendix A: Detailed information on the instances

See Table 1.

Table 1 Overview over the entire test set with the number of points (N) and the dimension (d)

ID	Instance	N	d
1	prnn_synth	250	2
2*	analcata_data_asbestos	73	3
3*	lupus	87	3
4	analcata_data_boxing1	120	3
5	analcata_data_boxing2	132	3
6	haberman	289	3
7	analcata_data_happiness	60	3

Table 1 continued

ID	Instance	N	d
8*	analcataiddata_aids	50	4
9	analcataiddata_lawsuit	263	4
10	iris	147	4
11	hayes_roth	93	4
12	balance_scale	625	4
13	parity5	32	5
14*	bupa	341	5
15	irish	470	5
16	phoneme	5349	5
17	tae	110	5
18	new_thyroid	215	5
19*	analcataiddata_bankruptcy	50	6
20*	analcataiddata_creditscore	100	6
21	mux6	64	6
22	monk3	357	6
23	monk1	432	6
24	monk2	432	6
25	appendicitis	106	7
26	prnn_crabs	200	7
27*	penguins	333	7
28	postoperative_patient_data	78	8
29*	biomed	209	8
30*	pima	768	8
31*	cars	392	8
32	analcataiddata_japansolvent	52	9
33	glass2	162	9
34	breast_cancer	272	9
35	saheart	462	9
36	threeOf9	512	9
37	profb	672	9
38	breast_w	463	9
39	tic_tac_toe	958	9
40	xd6	512	9
41	cmc	1425	9
42	analcataiddata_cyyoung9302	92	10
43	analcataiddata_cyyoung8092	97	10
44	breast	691	10
45	flare	315	10
46	parity5+5	1024	10
47	magic	18,905	10

Table 1 continued

ID	Instance	<i>N</i>	<i>d</i>
48	analcata_data_fraud	42	11
49	heart_statlog	270	13
50	heart_h	293	13
51	hungarian	293	13
52*	cleve	302	13
53*	heart_c	302	13
54	wine_recognition	178	13
55*	australian	690	14
56*	adult	48,790	14
57*	schizo	340	14
58*	buggyCrx	690	15
59	labor	57	16
60	house_votes_84	342	16
61	hepatitis	155	19
62*	credit_g	1000	20
63	gametes_e_0.1H	1599	20
64	gametes_e_0.4H	1600	20
65	gametes_e_0.2H	1600	20
66	gametes_h_50	1592	20
67	gametes_h_75	1599	20
68*	churn	5000	20
69*	ring	7400	20
70	twonorm	7400	20
71	waveform_21	5000	21
72	ann_thyroid	7129	21
73	spect	228	22
74	horse_colic	357	22
75	agaricus_lepiota	8124	22
76*	hypothyroid	3086	25
77*	dis	3711	29
78*	allhypo	3709	29
79*	allbp	3711	29
80*	breast_cancer_wisconsin	569	30
81	backache	180	32
82	ionosphere	351	34
83	chess	3196	36
84	waveform_40	5000	40
85	connect_4	67,557	42
86	spectf	267	44

Table 1 continued

ID	Instance	N	d
87*	tokyo1	959	44
88	molecular_biology_promoters	106	57
89*	spambase	4210	57
90	sonar	208	60
91	splice	2903	60
92	coil2000	8380	85
93*	Hill_Valley_without_noise	1212	100
94*	clean1	476	168
95*	clean2	6598	168
96	dna	3002	180
97	gametes_e_1000atts	1600	1000

Appendix B: Further numerical results

Besides the measures of accuracy and precision, we compare two further measures in this section. First, recall (RE) measures the percentage of points with positive label that are actually classified as positive. It is formally given by

$$\text{RE} := \frac{\text{TP}}{\text{TP} + \text{FN}}. \quad (16)$$

Note that for applications such as cancer diagnosis, it is relevant to evaluate recall because it is more important to flag cancer rather than to do not. Also in cases of rare positive labels, recall is often the favored metric. Note that values close to 1 indicate a better classification here.

Second, we also compare the false positive rate (FPR), which measures the probability of points with negative labels being classified as positive:

$$\text{FPR} := \frac{\text{FP}}{\text{TN} + \text{FP}}. \quad (17)$$

This quantity is important in some applications such as quality control, where a false positive can cause more issues than a false negative. Note that for FPR, the lower the value, the better the classification.

The main comparison in terms of recall and false positive rate is w.r.t. the “true hyperplane”, i.e., the solution of Problem (P1) on the complete data with all N points and all labels available. The main question is how close the recall and false positive rate is to the one of the true hyperplane. Hence, we compute the ratios of the recall and false positive rate according to

$$\widehat{\text{RE}} := \frac{\text{RE}}{\text{RE}_{\text{true}}}, \quad \widehat{\text{FPR}} := \frac{\text{FPR}}{\text{FPR}_{\text{true}}}, \quad (18)$$

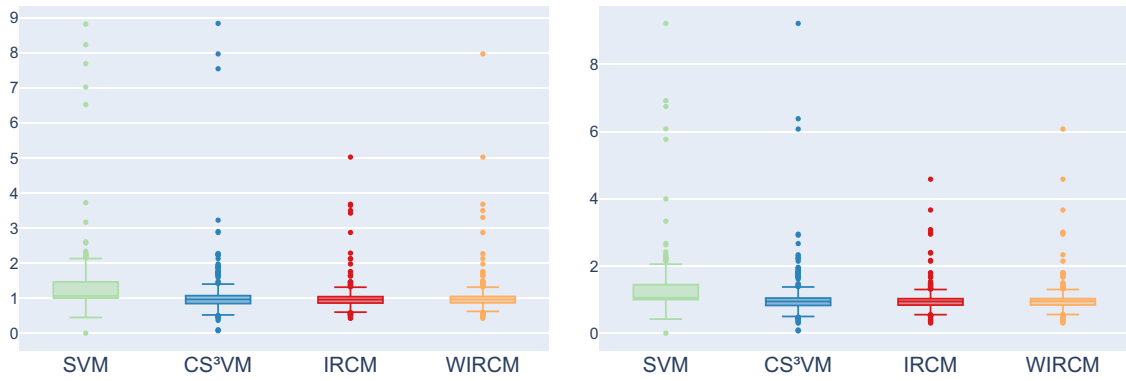


Fig. 12 Relative recall \widehat{RE} w.r.t. the true hyperplane; see (18). Left: Comparison for all data points. Right: Comparison only for unlabeled data points



Fig. 13 Relative false positive rate \widehat{FPR} w.r.t. the true hyperplane; see (18). Left: Comparison for all data points. Right: Comparison only for unlabeled data points

where RE_{true} and FPR_{true} are computed as in (16) and (17) for the true hyperplane.

As can be seen in Fig. 12, the SVM’s relative recall is a little bit larger than the one of the other methods. As in Sect. 6.4.4, this happens because the biased sample is more likely to have positive labeled data and having no information about the unlabeled data, the SVM ends up classifying points on the positive side.

Figure 13 shows that CS³VM, the IRCM, and the WIRCM have lower \widehat{FPR} values than the original SVM. This means that the newly proposed methods have a lower false positive rate than the original SVM. The fact that CS³VM terminates for less instances than the IRCM explains why the IRCM has a lower relative false positive rate than CS³VM. Finally, since the WIRCM uses the IRCM for warm-starting, the WIRCM also has better relative false positive rates than CS³VM.

Appendix C: Numerical results for simple random samples

In Sect. 6, we focused our computational study on non-representative, biased samples. The common baseline scenario to check the performance of estimators is to apply them on simple random samples. Hence, for completeness, we also present the results under simple random sampling. That is, each unit in the data set has the same probability $\pi_i = n/N$ to be included into the sample of size n . The instances are the same as

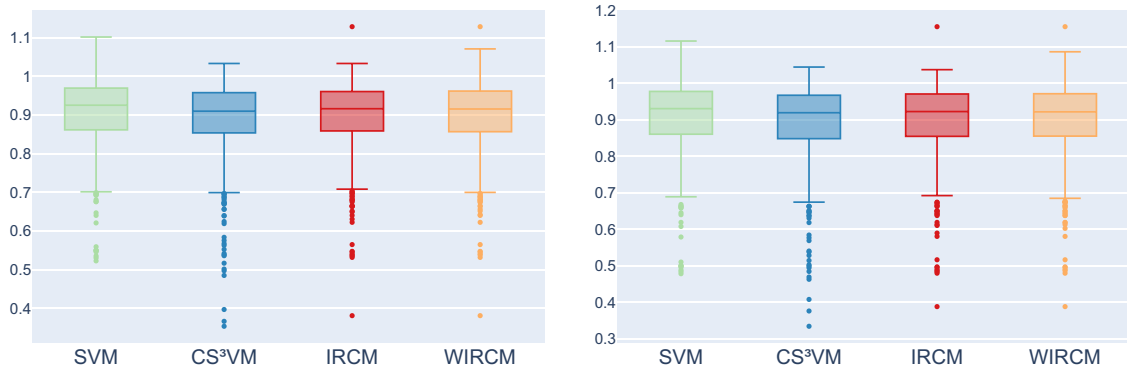


Fig. 14 Relative accuracy \widehat{AC} w.r.t. the true hyperplane; see (14), for the simple random samples. Left: Comparison for all data points. Right: Comparison only for unlabeled data points

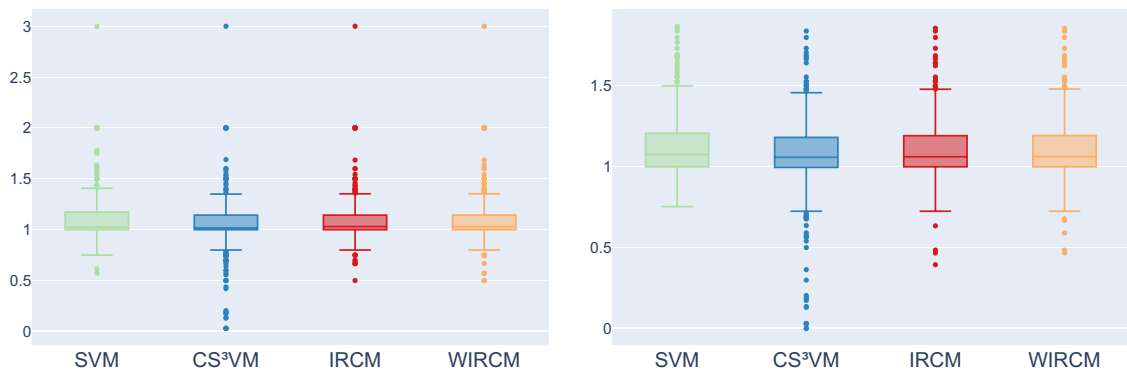


Fig. 15 Relative precision \widehat{PR} w.r.t. the true hyperplane; see (14), for the simple random samples. Left: Comparison for all data points. Right: Comparison only for unlabeled data points

described in Sect. 6.1. The computational setup follows the description in Sect. 6.2. As before, the used evaluation criteria are \widehat{AC} , \widehat{PR} as in (14) and \widehat{RE} , \widehat{FPR} as in (18).

Figures 14 and 15 show similar accuracy and precision performance for all approaches. This is as expected, as the sample is not biased and hence the cardinality constraint does not contribute relevant additional information to the problem. Therefore, the SVM does not tend to classify the points as positive as it is the case for the biased samples. The outliers, mainly present for CS^3VM , are due those instances that are not solved within the time limit. As can be seen in Figs. 16 and 17, recall and false positive rate are also similar for all approaches.

Hence, for the simple random samples our approaches have almost the same results as the SVM. Note that for the biased samples, they outperformed the SVM. Hence, in cases for which the type of sample is not known, it is “safe” to use the newly proposed approaches for classification.

Appendix D: Choosing the hyperparameters

Each parameter of Algorithm 2 and 3 as well as in Problem (P1) and (P3) can be chosen from a range. In Table 2 we present plausible ranges for these parameters.

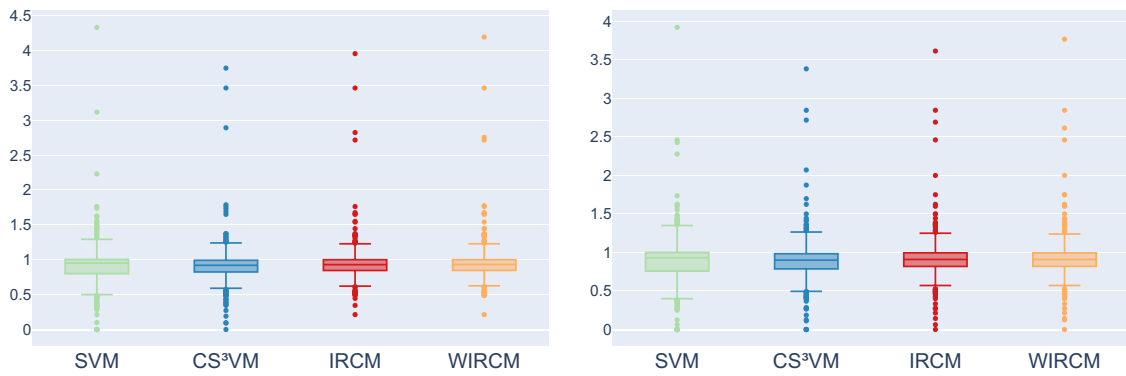


Fig. 16 Relative recall \widehat{RE} w.r.t. the true hyperplane; see (18), for the simple random samples. Left: Comparison for all data points. Right: Comparison only for unlabeled data points

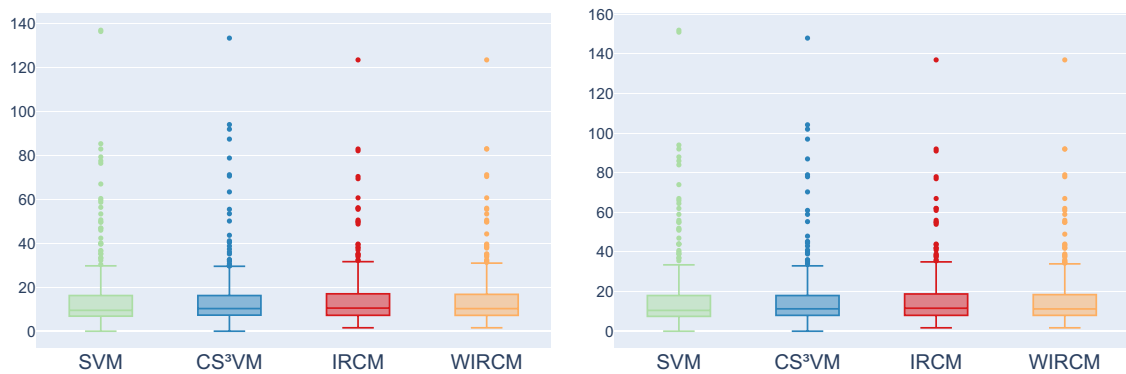


Fig. 17 Relative false positive rate \widehat{FPR} w.r.t. the true hyperplane; see (18), for the simple random samples. Left: Comparison for all data points. Right: Comparison only for unlabeled data points

Table 2 Plausible ranges for the hyperparameters

Parameter	Plausible range	Current choice
C_1	$\mathbb{R}_{\geq 0}$	1
C_2	$[0.5C_1, 2C_1]$	1
k^1	$[2, m]$	10, 20, 50
k^+	$[k^1, m]$	50
$\hat{\Delta}^1$	$[0.5, 0.9]$	0.8
$\tilde{\Delta}$	$[0.1, 1 - \hat{\Delta}^1]$	0.1
B_{\max}	$[1, m]$	$0.2m, 0.25m, 0.35m, 0.45m$
γ	$[1.1, m/B_{\max}]$	1.2
T_{\max}	$[10, 100]s$	40s

Clearly, $C_1 \in \mathbb{R}_{\geq 0}$ holds. However, the closer the value is to 1, the more equally important are maximizing the margin and minimizing the classification error for the labeled data. The range of C_2 is based on C_1 in order to indicate how much more important the unlabeled data is compared to the labeled data. Again, we choose $C_2 = 1$ so that both data have the same importance. Besides that, if C_2 is much bigger than C_1 , our preliminary tests showed that this leads to focus on minimizing the classification

error for the unlabeled data, which implies focusing on the binary variable and, hence, leads to larger run times.

For choosing the other parameters, we consider the first 3 datasets presented in Table 1 and varied the parameter choices in a preliminary numerical study. Based on the results, we now discuss how to choose the remaining parameters. The parameter k^1 can be between 2 and m since we cluster m unlabeled points. Note that, the smaller k^1 , the less time per iteration is needed since we have fewer binary variables. However, more iterations may be needed to find the solution. On the other hand, the bigger k^1 , the more time per iteration is required. We choose to start with a small value of k^1 because in preliminary numerical tests, when the algorithm terminated, the number of clusters never exceeded $m/3$. Moreover, in our preliminary tests, if the algorithm exceeds $k^t = 50$ for some iteration t , it takes a lot of time to solve Problem (P4). To decrease this time, we reduced the number of clusters, eliminating the ones being far from the hyperplane. This is the reason why we choose $k^+ = 50$.

The parameter $\hat{\Delta}^1$ indicates that clusters with a distance to the hyperplane greater than the $\hat{\Delta}^1$ -quantile of all distances will be deactivated. It is between 0.5 and 0.9 because a smaller value than 0.5 means removing points that are too close to the hyperplane. This implies that in next iterations many clusters can be reactivated. On the other hand, if it is larger than 0.9, it means that almost no clusters can be deactivated. We choose 0.8 because in our preliminary numerical tests we noticed that with a smaller value, many clusters were activated again, which increased the required time per iteration. The range of $\tilde{\Delta}$ is justified by the fact that for all t , the maximum value of $\hat{\Delta}^t$ is 1. We chose 0.1 because the higher the value we choose, the smaller the possibility to eliminate clusters becomes. If chosen smaller, $\hat{\Delta}^t$ and $\hat{\Delta}^{t+1}$ would be very similar and some clusters would be deactivated and reactivated several times.

Because we have m unlabeled points, we can fix at most m unlabeled points, which justifies the range of B_{\max} and the maximum value of γ . Since some points are not fixed on some side—they may be on the wrong side or it could take more than T_{\max} to solve Problem (P5)—we try to fix at least more than 10% of B_{\max} many unlabeled points. This is why the minimum value of γ is 1.1. The maximum value of T_{\max} is 100s because, if chosen smaller, we observe that there is often not enough time to solve Problem (P5). On the other hand, if it is larger, we observe that the time needed to solve the Algorithm 3 increases.

Appendix E: Run times in dependence of the number of data points

In this section, we complement Sect. 6 by presenting the run times in dependence of the number of points in the data set in order to shed some light on the scalability of our approaches. To this end, we split the entire data set in three subsets.

The first subset only considers those 46 data sets with $N \leq 500$. As can be seen in Fig. 18 (top), IRCM solves more than 75% of the instances while CS³VM and WIRCM solve more than 50%. The second subset contains 11 data sets with $N \in (500, 1500]$. Figure 18 (middle) shows that for these test sets, IRCM solves about 40% of the instances while CS³VM and WIRCM solve more than 10%. The last subset contains those 21 data sets with $N > 1500$. Figure 18 (bottom) shows that CS³VM and WIRCM

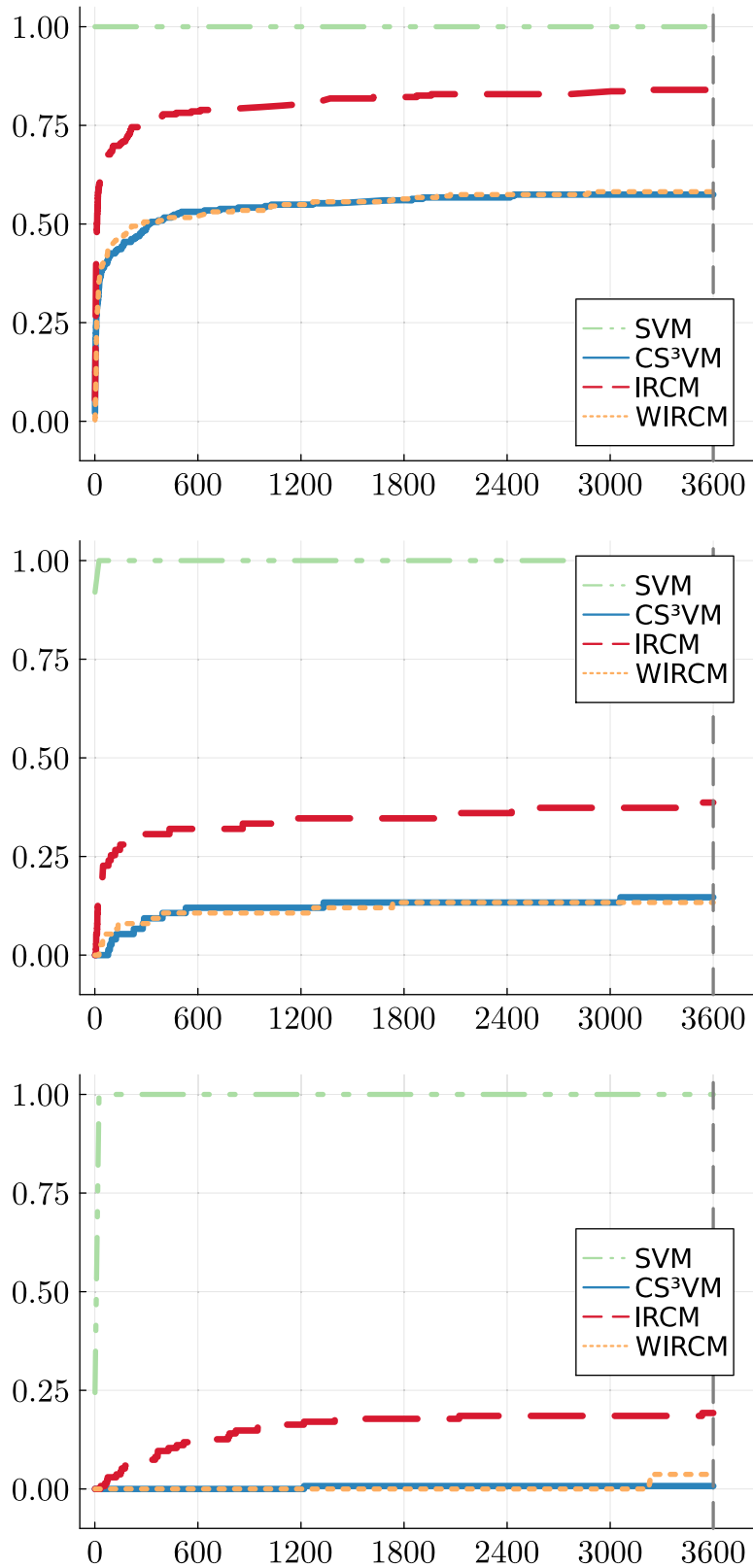


Fig. 18 ECDFs for run time (in seconds). Top: Instances with $N \geq 500$. Middle: Instances with $N \in (500, 1500]$. Bottom: Instances with $N > 1500$

do not solve any of these instances and IRCM solves about 20%. As expected, the larger the number of points and, thus, the larger the number of binary variables, the more challenging it is to solve the instances. Besides that, SVM solves all instances, which is expected since it does not include any binary variables.

Acknowledgements The authors thank the DFG for their support within RTG 2126 “Algorithmic Optimization”.

Funding Open Access funding enabled and organized by Projekt DEAL

Data availability We use the data from the literature that we properly cite.

Declarations

Conflict of interest The authors declare no conflict of interest.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article’s Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article’s Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

- Almasi ON, Rouhani M (2016) Fast and de-noise support vector machine training method based on fuzzy clustering method for large real world datasets. *Turk J Electr Eng Comput Sci* 24:219–233. <https://doi.org/10.3906/elk-1304-139>
- Aloise D, Deshpande A, Hansen P, Popat P (2009) NP-hardness of Euclidean sum-of-squares clustering. *Mach Learn* 75(2):245–248. <https://doi.org/10.1007/s10994-009-5103-0>
- Belkin M, Niyogi P, Sindhvani V (2006) Manifold regularization: a geometric framework for learning from labeled and unlabeled examples. *J Mach Learn Res* 7:2399–2434
- Bennett KP, Demiriz A (1998) Semi-supervised support vector machines. In: Proceedings of the 11th international conference on neural information processing systems. NIPS’98. MIT Press, Cambridge, pp 368–374. <https://proceedings.neurips.cc/paper/1998/file/b710915795b9e9c02cf10d6d2bdb688c-Paper.pdf>
- Birzhandi P, Youn HY (2019) CBCH (clustering-based convex hull) for reducing training time of support vector machine. *J Supercomput* 75(8):5261–5279. <https://doi.org/10.1007/s11227-019-02795-9>
- Birzhandi P, Kim KT, Youn HY (2002) Reduction of training data for support vector machine: a survey. *Soft Comput* 26(8):3729–3742. <https://doi.org/10.1007/s00500-022-06787-5>
- Boser BE, Guyon IM, Vapnik VN (1992) A training algorithm for optimal margin classifiers. In: Proceedings of the fifth annual workshop on computational learning theory. COLT ’92. ACM Press, Pittsburgh, pp 144–152. <https://doi.org/10.1145/130385.130401>
- Burgard JP, Krause J, Schmaus S (2021) Estimation of regional transition probabilities for spatial dynamic microsimulations from survey data lacking in regional detail. *Comput Stat Data Anal* 154:107048. <https://doi.org/10.1016/j.csda.2020.107048>
- Cervantes J, Li X, Yu W (2006) Support vector machine classification based on fuzzy clustering for large data sets, vol 4293. Springer, Berlin, pp 572–582. https://doi.org/10.1007/11925231_54
- Chapelle O, Zien A (2005) Semi-supervised classification by low density separation. In: Cowell RG, Ghahramani Z (eds) Proceedings of the tenth international workshop on artificial intelligence and statistics, vol R5. Proceedings of machine learning research. PMLR, pp 57–64. <http://proceedings.mlr.press/r5/chapelle05b/chapelle05b.pdf>

- Chapelle O, Chi M, Zien A (2006) A continuation method for semi-supervised SVMs. In: Proceedings of the 23rd international conference on machine learning. ICML '06. Association for Computing Machinery, New York, pp 185–192 <https://doi.org/10.1145/1143844.1143868>
- Cortes C, Vapnik V (1995) Support vector networks. *Mach Learn* 20:273–297. <https://doi.org/10.1007/BF00994018>
- Dasgupta S (2007) The hardness of k-means clustering. <https://cseweb.ucsd.edu/~dasgupta/papers/kmeans.pdf>
- de Almeida MB, de Pádua Braga A, Braga JP (2000) SVM-KM: speeding SVMs learning with a priori cluster selection and k-means. Proceedings of the sixth Brazilian symposium on neural networks 1:162–167. <https://doi.org/10.1109/SBRN.2000.889732>
- Dunning I, Huchette J, Lubin M (2017) JuMP: a modeling language for mathematical optimization. *SIAM Rev* 59(2):295–320. <https://doi.org/10.1137/15M1020575>
- Hyndman RJ, Fan Y (1996) Sample quantiles in statistical packages. *Am Stat* 50(4):361–365. <https://doi.org/10.2307/2684934>
- Joachims T (2002) Training transductive support vector machines. In: Learning to classify text using support vector machines. Springer, New York, pp 163–174. https://doi.org/10.1007/978-1-4615-0907-3_9
- Kontonatsios G, Brockmeier AJ, Przybyła P, McNaught J, Mu T, Goulermas JY, Ananiadou S (2017) A semi-supervised approach using label propagation to support citation screening. *J Biomed Inf* 72:67–76. <https://doi.org/10.1016/j.jbi.2017.06.018>
- Lloyd S (1982) Least squares quantization in PCM. *IEEE Trans Inf Theory* 28(2):129–137. <https://doi.org/10.1109/TIT.1982.1056489>
- MacQueen J (1967) Classification and analysis of multivariate observations. In: Proceedings of the fifth Berkeley symposium on mathematical statistics and probability, pp 281–297
- Mahajan M, Nimbhorkar P, Varadarajan K (2012) The planar k-means problem is NP-hard. *Theor Comput Sci* 442:13–21. <https://doi.org/10.1016/j.tcs.2010.05.034>
- Melacci S, Belkin M (2009) Laplacian support vector machines trained in the primal. *J Mach Learn Res*. [arXiv:0909.5422](https://arxiv.org/abs/0909.5422)
- Olson RS, La Cava W, Orzechowski P, Urbanowicz RJ, Moore JH (2017) PMLB: a large benchmark suite for machine learning evaluation and comparison. *BioData Min* 10(36):1–13. <https://doi.org/10.1186/s13040-017-0154-4>
- Skinner CJ, D'arrigo, (2011) Inverse probability weighting for clustered nonresponse. *Biometrika* 98(4):953–966. <https://doi.org/10.1093/biomet/asr058>
- Yao Y, Liu Y, Yu Y, Xu H, Lv W, Li Z, Chen X (2013) K-SVM: an effective SVM algorithm based on K-means clustering. *J Comput*. <https://doi.org/10.4304/jcp.8.10.2632-2639>
- Yu X, Yang J, Zhan J-P (2012) A transductive support vector machine algorithm based on spectral clustering. *AASRI Proc* 1:384–388. <https://doi.org/10.1016/j.aasri.2012.06.059>

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Paper 2

Mixed-Integer Linear Optimization for Semi-Supervised Optimal Classification Trees

Jan Pablo Burgard, Maria Eduarda Pinheiro, Martin Schmidt

Preprint under review

<https://arxiv.org/abs/2401.09848>

MIXED-INTEGER LINEAR OPTIMIZATION FOR SEMI-SUPERVISED OPTIMAL CLASSIFICATION TREES

JAN PABLO BURGARD, MARIA EDUARDA PINHEIRO, MARTIN SCHMIDT

ABSTRACT. Decision trees are one of the most famous methods for solving classification problems, mainly because of their good interpretability properties. Moreover, due to advances in recent years in mixed-integer optimization, several models have been proposed to formulate the problem of computing optimal classification trees. The goal is, given a set of labeled points, to split the feature space with hyperplanes and assign a class to each partition. In certain scenarios, however, labels are exclusively accessible for a subset of the given points. Additionally, this subset may be non-representative, such as in the case of self-selection in a survey. Semi-supervised decision trees tackle the setting of labeled and unlabeled data and often contribute to enhancing the reliability of the results. Furthermore, undisclosed sources may provide extra information about the size of the classes. We propose a mixed-integer linear optimization model for computing semi-supervised optimal classification trees that cover the setting of labeled and unlabeled data points as well as the overall number of points in each class for a binary classification. Our numerical results show that our approach leads to a better accuracy and a better Matthews correlation coefficient for biased samples compared to other optimal classification trees, even if only few labeled points are available.

1. INTRODUCTION

Decision trees are among the most popular approaches for supervised classification (Breiman et al. 1984; Quinlan 1986). One of the main reasons for this is that they are easy to interpret compared to other machine learning models. The core idea is to recursively partition the feature space, according to branching rules, and assign a label to each resulting part of the partition.

One way to partition the data is to use hyperplanes involving a single feature, which then leads to so-called univariate trees; see, e.g., Kotsiantis (2014) and Yildiz and Dikmen (2007). In a multivariate tree, these hyperplanes involve more than one feature and some approaches for this setting are given in Altincay (2007), Bennett and Blue (1996), and Orsenigo and Vercellis (2003). In many algorithms for univariate or multivariate trees, each separate hyperplane is generated by minimizing a local impurity function, i.e., they do not build the tree by solving just a single optimization problem.

In recent years, due to the advancement of algorithms for mixed-integer programming (MIP), many strategies for computing optimal classification trees (OCT) by globally solving an optimization problem using MIP techniques have been proposed. Some techniques are discussed in the recent surveys by Carrizosa et al. (2021) and Gambella et al. (2021). The first approaches were proposed by Bertsimas and Dunn (2017). They present two mixed-integer linear programming (MILP) models based on univariate and multivariate trees. Verwer and Zhang (2019) propose a binary

Date: January 17, 2024.

2020 Mathematics Subject Classification. 90C11, 90C90, 90-08, 68T99.

Key words and phrases. Semi-supervised learning, Optimal classification trees, Mixed-integer linear optimization.

linear formulation in which the problem size is largely independent of the size of the training data. MIP approaches that consider support vector machines (Cortes and Vapnik 1995) to split the tree also have been explored, as can be seen in Blanco et al. (2022a), Blanco et al. (2022b), and D’Onofrio et al. (2023).

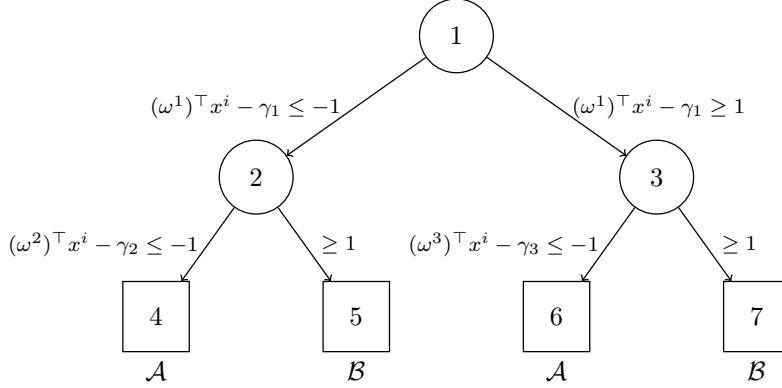
Besides the MIP models to solve an OCT, Blanquero et al. (2021) proposed a nonlinear optimization approach to compute an optimal “randomized” classification tree. In their approach, each data point is assigned to a class only with a given probability. The model uses oblique cuts with the goal of utilizing fewer predictor variables in the splits of the tree. Furthermore, Blanquero et al. (2019) aggregate local and global sparsity in the randomized tree by means of regularization with polyhedral norms.

All the strategies presented so far exclusively focus on labeled data. However, acquiring labels for every unit of interest can be expensive—in particular if classic surveys are used to obtain the labels. In this case, it would be beneficial to train the decision tree on only partly labeled data. This yields a semi-supervised learning setting (Zhu and Goldberg 2009). Algorithms for semi-supervised learning have already been proposed for SVM (Chapelle et al. 2006; Melacci and Belkin 2009), neural networks (Lee 2013; Nguyen et al. 2023; Oliver et al. 2018), and logistic regression (Amini and Gallinari 2002; Bzdok et al. 2015).

In the field of semi-supervised decision trees, Kim (2016) splits internal nodes by utilizing the structural characteristics of the data for subspace partition. Moreover, Kocev et al. (2017) consider minimizing a local impurity function and Tanha et al. (2017) propose self-training as base learners. Recently, Santhiappan and Ravindran (2021) consider a maximum mean discrepancy to estimate the class ratio at every splitting rule in a univariate decision tree. Furthermore, Zharmagambetov and Carreira-Perpinan (2022) present a graph Laplacian approach to deal with the unlabeled data. Hence, although they present different ideas, none of the approaches considers globally solving a single optimization problem.

Moreover, in many cases, external sources provide information about the total amount of elements in each class within a population. For example, in some businesses, the number of positive labels might be available, but the identification of which customer has a positive or negative label is unknown. An intuitive example is a supermarket for which the amount of cash payments is known. However, this information is not attributable to the individual customers ex-post. Another example is population surveys, where statistics agencies can provide how many people are employed. For logistic regression, the idea of aggregating this extra information is proposed by Burgard et al. (2021), who develop a cardinality-constrained multinomial logit model. In the SVM setting, Burgard et al. (2023) present a mixed-integer quadratic optimization model and iterative clustering techniques to tackle cardinality constraints for each class. Our contribution here is to propose to add this aggregated additional information to a multivariate OCT model by imposing a cardinality constraint on the predicted labels for the unlabeled data.

We develop a big- M -based MILP to solve the semi-supervised optimal classification tree (S²OCT) problem that deals with the cardinality constraint for the unlabeled data. The cardinality constraint helps to account for biased samples since the number of predictions in each class on the population is bounded by the constraint. This paper is organized as follows. In Section 2 we present preliminary concepts and our optimization problem. Afterward, the big- M -based MILP formulation is presented in Section 3. In Section 4, numerical results are reported and discussed and we conclude in Section 5.

FIGURE 1. A classification tree with depth $D = 2$

2. PRELIMINARY CONCEPTS

Let $X \in \mathbb{R}^{p \times N} = [X_l, X_u]$ be the data matrix with labeled data $X_l = [x^1, \dots, x^n]$ and unlabeled data $X_u = [x^{n+1}, \dots, x^N]$. Hence, we observe $x^i \in \mathbb{R}^p$ for all $i \in [1, N] := \{1, \dots, N\}$. We set $m := N - n$, $\mathcal{U} := [n + 1, \dots, N]$, and $c \in \{\mathcal{A}, \mathcal{B}\}^n$ as the vector of class labels for the labeled data.

In a multivariate optimal classification tree, each decision consists of a linear combination of the p components of a point x^i . Considering only the labeled data, the goal is to split the feature space into distinct regions to correctly classify each point. However, in many applications, aggregated information on the labels is available, e.g., from census data. In the following, we know the total number $\lambda \in \mathbb{N}$ of unlabeled points that belong to the class \mathcal{A} and adapt the idea of optimal classification trees such that we can use the unlabeled data as well as λ as additional information.

Given a depth $D \in \mathbb{N}$, a classification tree has $2^{D+1} - 1$ nodes, categorized in two types; the branch nodes $\mathcal{T}_B = [1, 2^D - 1]$ and the leaf nodes $\mathcal{T}_L = [2^D, 2^{D+1} - 1]$. Each branch node $b \in \mathcal{T}_B$ provides a hyperplane parameterized by (ω^b, γ_b) that splits the feature space into half-spaces. As suggested by Bertsimas and Dunn (2017), if a point x^i , $i \in [1, N]$, satisfies $(\omega^b)^\top x^i - \gamma_b \leq 0$, then x^i follows the left branch of the node b . If $(\omega^b)^\top x^i - \gamma_b > 0$ holds, the point x^i follows the right branch of the node b . For an optimization formulation, the strict inequality needs to be re-written as $(\omega^b)^\top x^i - \gamma_b \geq \varepsilon$ for a sufficiently small $\varepsilon > 0$. However, a very small value of ε might lead to numerical instabilities. To avoid this, we replace $(\omega^b)^\top x^i - \gamma_b \leq 0$ by $(\omega^b)^\top x^i - \gamma_b \leq -1$, and $(\omega^b)^\top x^i - \gamma_b \geq \varepsilon$ by $(\omega^b)^\top x^i - \gamma_b \geq 1$. Note that this change is always possible except for the rare cases that a data point actually lies on a hyperplane.

In each leaf node, $t \in \mathcal{T}_L$, all points x^i , $i \in [1, N]$, are classified as \mathcal{A} or \mathcal{B} . In a simple example with $D = 2$, the classification tree has $\mathcal{T}_B = [1, 3]$ and $\mathcal{T}_L = [4, 7]$; see Figure 1. Regarding the classification at the leaf nodes, let $\mathcal{T}_L^{\mathcal{A}} = \{t \in \mathcal{T}_L : t \text{ is even}\}$ be the set of leaf nodes that are classified as \mathcal{A} and $\mathcal{T}_L^{\mathcal{B}} = \{t \in \mathcal{T}_L : t \text{ is odd}\}$ be the set of leaf nodes that are classified as \mathcal{B} . For the classification tree with $D = 2$, see Figure 1 again, $\mathcal{T}_L^{\mathcal{A}} = \{4, 6\}$ and $\mathcal{T}_L^{\mathcal{B}} = \{5, 7\}$ holds.

For each leaf node $t \in \mathcal{T}_L$, define $\mathcal{N}_R(t)$ as the index set of the branch nodes in which the right or “greater than” branch is traversed to reach leaf t . Moreover, we define $\mathcal{N}_L(t)$ as the index set of the branch nodes in which the left or “less than”

branch is traversed to reach leaf t . For the classification tree in Figure 1 we have

$$\begin{aligned}\mathcal{N}_L(4) &= \{1, 2\}, & \mathcal{N}_R(4) &= \emptyset, & \mathcal{N}_L(5) &= \{1\}, & \mathcal{N}_R(5) &= \{2\}, \\ \mathcal{N}_L(6) &= \{3\}, & \mathcal{N}_R(6) &= \{1\}, & \mathcal{N}_L(7) &= \emptyset, & \mathcal{N}_R(7) &= \{1, 3\}.\end{aligned}$$

Given a fixed depth D , a point x^i , $i \in [1, n]$, with $c_i \in \{\mathcal{A}, \mathcal{B}\}$, is correctly classified if all inequalities from the root to the leaf node are satisfied for some leaf node $t \in \mathcal{T}_L^{c_i}$. Hence, a point x^i is correctly classified if

$$\bigvee_{t \in \mathcal{T}_L^{c_i}} \left\{ \left\{ \bigwedge_{b \in \mathcal{N}_R(t)} [(\omega^b)^\top x^i - \gamma_b \geq 1] \right\} \wedge \left\{ \bigwedge_{b \in \mathcal{N}_L(t)} [(\omega^b)^\top x^i - \gamma_b \leq -1] \right\} \right\}. \quad (1)$$

is satisfied.

In our running example with $D = 2$, a point x^i with $c_i = \mathcal{A}$ is correctly classified if

$$\begin{aligned}& \left\{ [(\omega^1)^\top x^i - \gamma_1 \leq -1] \wedge [(\omega^2)^\top x^i - \gamma_2 \leq -1] \right\} \\ & \vee \left\{ [(\omega^1)^\top x^i - \gamma_1 \geq 1] \wedge [(\omega^3)^\top x^i - \gamma_3 \leq -1] \right\}\end{aligned}$$

holds.

An unlabeled point x^i , $i \in [n+1, N]$, is classified as \mathcal{A} if

$$\bigvee_{t \in \mathcal{T}_L^{\mathcal{A}}} \left\{ \left\{ \bigwedge_{b \in \mathcal{N}_R(t)} [(\omega^b)^\top x^i - \gamma_b \geq 1] \right\} \wedge \left\{ \bigwedge_{b \in \mathcal{N}_L(t)} [(\omega^b)^\top x^i - \gamma_b \leq -1] \right\} \right\} \quad (2)$$

is true. Thus, our goal is to find ω and γ such that Expression (1) is satisfied for all labeled data points x^i , $i \in [1, n]$, and such that the number of unlabeled points x^i , $i \in [n+1, N]$, that satisfy Expression (2) is as close as possible to λ .

For doing so, we first need to define suitable error measures that then have to be minimized. For this purpose, we define the branch and leaf error according to the decision error and leaf error proposed by Bennett and Blue (1996). The first error is related to branch nodes. For each labeled point, at each branch node, we consider the inequality that must be satisfied and then measure by how much it is violated.

Definition 1 (Branch Error). *Given a labeled point x^i , $i \in [1, n]$, in any branch node $b \in \mathcal{T}_B$, the branch errors $(y_b^R)_i$ and $(y_b^L)_i$ are defined by*

$$(y_b^R)_i := \left[-(\omega^b)^\top x^i + \gamma_b + 1 \right]^+ \quad \text{and} \quad (y_b^L)_i := \left[(\omega^b)^\top x^i - \gamma_b + 1 \right]^+$$

with $[v]^+ := \max\{0, v\}$.

The definition above can be interpreted as follows. If the point x^i satisfies $(\omega^b)^\top x^i - \gamma_b \geq 1$, and therefore follows the right branch in some node $b \in \mathcal{T}_B$, it holds $(y_b^R)_i = 0$ and $(y_b^L)_i > 0$. However, if the point follows the left branch in some node $b \in \mathcal{T}_B$, i.e., if $(\omega^b)^\top x^i - \gamma_b \leq -1$ holds, we obtain $(y_b^R)_i > 0$ and $(y_b^L)_i = 0$.

The next definition represents the error in each leaf node t . For each labeled point, we sum over the branch errors along the path from the root to the leaf node t .

Definition 2 (Leaf Error). *The leaf error of a labeled point x^i , $i \in [1, n]$, at a leaf node $t \in \mathcal{T}_L$ is defined by*

$$LE(x^i, t) := \sum_{b \in \mathcal{N}_R(t)} (y_b^R)_i + \sum_{b \in \mathcal{N}_L(t)} (y_b^L)_i. \quad (3)$$

Note that $\text{LE}(x^i, t)$ is a linear expression and for each labeled point x^i , $i \in [1, n]$, Expression (1) is satisfied if $\text{LE}(x^i, t) = 0$ for some $t \in \mathcal{T}_L^{c_i}$. Additionally, $\text{LE}(x^i, t) \geq 0$ holds for all $t \in \mathcal{T}_L$. Hence, each labeled point x^i is correctly classified if the minimum value of all leaf errors is zero, i.e., if

$$\min_{t \in c_i} \{\text{LE}(x^i, t)\} = 0$$

holds. Besides that, we want to classify λ unlabeled points as \mathcal{A} , which means λ unlabeled points must end up in some $t \in \mathcal{T}_L^{\mathcal{A}}$.

To sum up, given the data matrix $X \in \mathbb{R}^{p \times N}$ and $\lambda \in \mathbb{N}$ as well as some $s > 0$, our goal is to find optimal parameters $\omega \in \mathbb{R}^{p \times 2^D - 1}$, $\gamma \in \mathbb{R}^{2^D - 1}$ as well as $y^R, y^L \in \mathbb{R}^{2^{D-1} \times n}$ and $\xi \in \mathbb{R}$ that solve the optimization problem

$$\min_{\omega, \gamma, y^R, y^L, \xi} \sum_{i=1}^n \min_{t \in \mathcal{T}_L^{c_i}} \{\text{LE}(x^i, t)\} + C\xi \quad (\text{P1a})$$

$$\text{s.t. } (y_b^R)_i \geq -(\omega^b)^\top x^i + \gamma_b + 1, \quad b \in \mathcal{T}_B, \quad i \in [1, n], \quad (\text{P1b})$$

$$(y_b^L)_i \geq (\omega^b)^\top x^i - \gamma_b + 1, \quad b \in \mathcal{T}_B, \quad i \in [1, n], \quad (\text{P1c})$$

$$-s \leq \omega_j^b \leq s, \quad b \in \mathcal{T}_B, \quad j \in [1, p], \quad (\text{P1d})$$

$$(y_b^R)_i, (y_b^L)_i \geq 0, \quad b \in \mathcal{T}_B, \quad i \in [1, n], \quad (\text{P1e})$$

$$\lambda - \xi \leq \sum_{i=n+1}^N \sum_{t \in \mathcal{T}_L^{\mathcal{A}}} (\psi(x^i, t)) \leq \lambda + \xi, \quad (\text{P1f})$$

$$\xi \geq 0 \quad (\text{P1g})$$

where $\text{LE}(x^i, t)$ is defined in (3) and

$$\psi(x^i, t) = \begin{cases} 1, & \text{if } x^i \text{ ends in the leaf node } t, \\ 0, & \text{otherwise.} \end{cases} \quad (4)$$

Note that the objective function in (P1a) models a compromise between minimizing the classification error for the labeled and unlabeled data. The penalty parameter $C > 0$ aims to control the importance of the slack variable ξ . Constraints (P1b) and (P1c) enforce on which branch (left or right) the labeled point x^i should traverse in branch node b . Constraint (P1d) defines the domain of each variable ω^b . This constraint is not necessary for the correctness of the model but will serve as a big- M -type parameter that is later used for deriving certain bounds; see Section 3. Constraint (P1f) ensures that the number of unlabeled data classified as \mathcal{A} is as close to λ as possible.

The functions $\min\{\text{LE}(x^i, t) : t \in \mathcal{T}_L^{c_i}\}$ in the objective function (P1a) and $\psi(x^i, t)$ in Constraint (P1g) are discontinuous, which means that Problem (P1) cannot be solved easily by standard solvers as such. Hence, we will present a mixed-integer linear programming (MILP) formulation using binary variables to re-model the objective function (P1a) and to count the classification of unlabeled points.

3. THE MILP MODEL

We start the development of a MILP model by using classic SOS1-techniques (Beale and Tomlin 1969) and McCormick inequalities (McCormick 1976) to rephrase a min min problem as an MILP formulation.

Lemma 1. Consider a set of continuous functions $f_k : \mathbb{R}^p \rightarrow \mathbb{R}$, $k \in [1, d]$, for some $d \in \mathbb{N}$, and let $\Omega \subseteq \mathbb{R}^p$ be given. Suppose further that there exist values $u_k > 0$ such that

$$0 \leq f_k(x) \leq u_k$$

holds for all $x \in \Omega$ and $k \in [1, d]$. Then, $x^* \in \mathbb{R}^p$ is a solution to the problem

$$\min_x \min_{k \in [1, d]} f_k(x) \quad (5a)$$

$$\text{s.t. } x \in \Omega \quad (5b)$$

if and only if there exist $\alpha^*, \beta^* \in \mathbb{R}^d$ such that (x^*, α^*, β^*) is a solution to the problem

$$\min_{x, \alpha, \beta} \sum_{k=1}^d \beta_k \quad (6a)$$

$$\text{s.t. } x \in \Omega, \quad (6b)$$

$$\sum_{k=1}^d \alpha_k = 1, \quad (6c)$$

$$\alpha_k \in \{0, 1\}, \quad k \in [1, d], \quad (6d)$$

$$\beta_k \leq u_k \alpha_k, \quad k \in [1, d], \quad (6e)$$

$$\beta_k \leq f_k(x), \quad k \in [1, d], \quad x \in \Omega, \quad (6f)$$

$$\beta_k \geq f_k(x) - u_k(1 - \alpha_k), \quad k \in [1, d], \quad x \in \Omega, \quad (6g)$$

$$\beta_k \geq 0, \quad k \in [1, d]. \quad (6h)$$

Proof. By introducing the binary variables α_k , we obtain that x^* is a solution to Problem (5) if and only if (x^*, α^*) is a solution of the problem

$$\min_{x, \alpha} \sum_{k=1}^d \alpha_k f_k(x) \\ \text{s.t. } (6b)-(6d).$$

Besides that, for any (x^*, α^*) solution of the problem above, if $\alpha_k^* = 0$ holds for some $k \in [1, d]$, Constraints (6e) and (6h) enforce that $\beta_k^* = 0 = \alpha_k^* f_k(x^*)$. On the other hand, if $\alpha_k^* = 1$ holds, by Constraints (6f) and (6g), we obtain $\beta_k^* = \alpha_k^* f_k(x^*)$.

Hence, x^* is a solution to Problem (5) if and only if (x^*, α^*, β^*) exists such that

$$\beta_k^* = \alpha_k^* f_k(x^*), \quad k \in [1, d],$$

is a solution to Problem (6). \square

To apply the previous lemma to Problem (P1) it is necessary that $LE(x^i, t)$ has lower and upper bounds, which are given in following proposition. Here and in what follows, $\|\cdot\|$ denotes the Euclidean norm.

Proposition 1. Given $[x^1, \dots, x^N]$ with $x^i \in \mathbb{R}^p$ for all $i \in [1, N]$ and $s > 0$, every optimal solution $(\omega, \gamma, y^R, y^L, \xi)$ to Problem (P1) satisfies

$$|(\omega^b)^\top x^i - \gamma_b| \leq \eta s \sqrt{p}, \quad i \in [1, N], \quad b \in \mathcal{T}_B,$$

and

$$(y_b^R)_i \leq \eta s \sqrt{p} + 1, \quad (y_b^L)_i \leq \eta s \sqrt{p} + 1, \quad i \in [1, n], \quad b \in \mathcal{T}_B,$$

where $\eta := \max_{i, k \in [1, N]} \|x^i - x^k\|$. Moreover,

$$0 \leq LE(x^i, t) \leq B(s), \quad t \in i \in [1, n], \quad t \in \mathcal{T}_L,$$

holds with

$$B(s) := D(\eta s \sqrt{p} + 1). \quad (7)$$

Proof. Let \mathcal{H} be the set of hyperplanes that separates $[x^1, \dots, x^N]$ into two sets, and let the hyperplane $(\omega^b, \gamma_b) \in \mathcal{H}$ be the hyperplane with the minimal maximum distance to any point in X . Then, according to Blanco et al. (2022a), the maximum distance from a point $x^i, i \in [1, N]$, to (ω^b, γ_b) is η , i.e.,

$$\frac{|(\omega^b)^\top x^i - \gamma_b|}{\|\omega^b\|} \leq \eta.$$

Moreover, Constraint (P1d) enforces that $\|\omega^b\| \leq s\sqrt{p}$. Hence,

$$|(\omega^b)^\top x^i - \gamma_b| \leq \eta s\sqrt{p}, \quad i \in [1, N], \quad b \in \mathcal{T}_B,$$

holds and

$$-(\omega^b)^\top x^i + \gamma_b + 1 \leq \eta s\sqrt{p} + 1 \quad \text{as well as} \quad (\omega^b)^\top x^i - \gamma_b + 1 \leq \eta s\sqrt{p} + 1$$

are satisfied for all $b \in \mathcal{T}_B$ and $i \in [1, n]$.

Note further that Problem (P1) is a minimization problem and $\text{LE}(x^i, t)$ is a sum of the D non-negatives terms y_b^R and y_b^L . Thus, Constraints (P1b) and (P1c) imply that

$$(y_b^R)_i \leq \eta s\sqrt{p} + 1, \quad (y_b^L)_i \leq \eta s\sqrt{p} + 1, \quad i \in [1, n], \quad b \in \mathcal{T}_B,$$

and

$$0 \leq \text{LE}(x^i, t) \leq B(s), \quad t \in \mathcal{T}_L,$$

holds. \square

Note that Constraint (P1d) is decisive for obtaining the upper bound $B(s)$ in the previous lemma. Finally, to overcome the discontinuity of the function $\psi(\cdot)$ defined in (4), we also add binary variables and use SOS techniques again to turn on or off the enforcement of a constraint. More formal, by introducing the matrices $\beta \in \mathbb{R}^{n \times 2^{D-1}}$, $\alpha \in \{0, 1\}^{n \times 2^{D-1}}$, $z \in \{0, 1\}^{m \times 2^{D-1}}$, and $\delta \in \{0, 1\}^{m \times 2^{D-1}}$ of binary variables, we can reformulate the optimization problem (P1) as follows:

$$\min_{\omega, \gamma, y^R, y^L, \xi, \alpha, \beta, z, \delta} \sum_{i=1}^n \sum_{t \in \mathcal{T}_L^{c_i}} \beta_t^i + C\xi \quad (\text{P2a})$$

s.t. (P1b)–(P1e),

$$\sum_{t \in \mathcal{T}_L^{c_i}} \alpha_t^i = 1, \quad i \in [1, n], \quad (\text{P2b})$$

$$\alpha_t^i \in \{0, 1\}, \quad i \in [1, n], \quad t \in \mathcal{T}_L^{c_i}, \quad (\text{P2c})$$

$$\beta_t^i \leq \text{LE}(x^i, t), \quad i \in [1, n], \quad t \in \mathcal{T}_L^{c_i}, \quad (\text{P2d})$$

$$\beta_t^i \geq \text{LE}(x^i, t) - B(s)(1 - \alpha_t^i), \quad i \in [1, n], \quad t \in \mathcal{T}_L^{c_i}, \quad (\text{P2e})$$

$$0 \leq \beta_t^i \leq B(s)\alpha_t^i, \quad i \in [1, n], \quad t \in \mathcal{T}_L, \quad (\text{P2f})$$

$$(\omega^b)^\top x^i - \gamma_b \leq z_i^b M - 1, \quad b \in \mathcal{T}_B, \quad i \in \mathcal{U}, \quad (\text{P2g})$$

$$(\omega^b)^\top x^i - \gamma_b \geq -(1 - z_i^b)M + 1, \quad b \in \mathcal{T}_B, \quad i \in \mathcal{U}, \quad (\text{P2h})$$

$$z_i^b \in \{0, 1\}, \quad b \in \mathcal{T}_B, \quad i \in \mathcal{U}, \quad (\text{P2i})$$

$$\delta_i^t \leq z_i^b, \quad b \in \mathcal{N}_R(t), \quad t \in \mathcal{T}_L^A, \quad i \in \mathcal{U}, \quad (\text{P2j})$$

$$\delta_i^t \leq -z_i^b + 1, \quad b \in \mathcal{N}_L(t), \quad t \in \mathcal{T}_L^A, \quad i \in \mathcal{U}, \quad (\text{P2k})$$

$$\delta_i^t \geq \sum_{b \in \mathcal{N}_R(t)} z_i^b + \sum_{b \in \mathcal{N}_L(t)} (-z_i^b + 1) - (D - 1), \quad t \in \mathcal{T}_L^A, \quad i \in \mathcal{U}, \quad (\text{P2l})$$

$$\delta_i^t \in \{0, 1\}, \quad t \in \mathcal{T}_L^A, \quad i \in \mathcal{U}, \quad (\text{P2m})$$

$$\lambda - \xi \leq \sum_{i=1+n}^N \sum_{t \in \mathcal{T}_L^A} \delta_i^t \leq \lambda + \xi, \quad (\text{P2n})$$

$$\xi \geq 0, \quad (\text{P2o})$$

where M needs to be chosen sufficiently large. The constraints in (P2c) enforce that the minimum value of $\text{LE}(x^i, t)$ is selected for each $x^i \in [1, n]$ and Constraints (P2d)–(P2f) ensure

$$\beta_t^i = \alpha_i^i \text{LE}(x^i, t), \quad i \in [1, n], \quad t \in \mathcal{T}_L.$$

As z_i^b is binary, Constraints (P2g) and (P2h) lead to

$$\begin{aligned} (\omega^b)^\top x^i - \gamma_b \geq 1 &\implies z_i^b = 1, \quad b \in \mathcal{T}_B, \quad i \in \mathcal{U}, \\ (\omega^b)^\top x^i - \gamma_b \leq -1 &\implies z_i^b = 0, \quad b \in \mathcal{T}_B, \quad i \in \mathcal{U}. \end{aligned}$$

Furthermore, as δ_i^t is binary as well, for all $i \in \mathcal{U}$ and $t \in \mathcal{T}_L^A$, Constraints (P2j)–(P2l) lead to

$$\delta_i^t = \begin{cases} 1, & \text{if } z_i^b = 1 \text{ for } b \in \mathcal{N}_R(t) \text{ and } z_i^b = 0 \text{ for } b \in \mathcal{N}_L(t), \\ 0, & \text{otherwise,} \end{cases}$$

i.e.,

$$\delta_i^t = \begin{cases} 1, & \text{if } x^i \text{ ends up in the leaf node } t, \\ 0, & \text{otherwise.} \end{cases}$$

Constraint (P2n) ensures that the number of unlabeled data classified as \mathcal{A} is as close to λ as possible. Reformulation (P2) is an MILP. We refer to this problem as S²OCT. As usual in mixed-integer optimization, the big- M -value is crucial as is the choice of s in Constraint (P1d). However, precisely based on s we have an exact value for M , as discussed in the following theorem.

Theorem 2. Consider $\eta := \max_{i,k \in [1, N]} \|x^i - x^k\|$. Any feasible point for Problem (P2) satisfies (P2g) and (P2h) for $M = \eta s \sqrt{p} + 1$.

Proof. Note that if $z_i^b = 1$ for some $i \in \mathcal{U}$ and $b \in \mathcal{T}_B$, Constraints (P2g) and (P2h) imply

$$M - 1 \geq (\omega^b)^\top x^i - \gamma_b \geq 1.$$

Moreover, since $(\omega^b)^\top x^i - \gamma_b \geq 0$, because of Proposition 1, we get

$$(\omega^b)^\top x^i - \gamma_b \leq \eta s \sqrt{p}.$$

This means that $M = \eta s \sqrt{p} + 1$ does not cut off any feasible solution.

On the other hand, if $z_i^b = 0$ holds for some $i \in \mathcal{U}$ and $b \in \mathcal{T}_B$, due to Constraint (P2g) and (P2h), we obtain

$$1 - M \leq (\omega^b)^\top x^i - \gamma_b \leq -1,$$

and, similarly, $M = \eta s \sqrt{p} + 1$ does not cut of any feasible solution as well. \square

4. NUMERICAL RESULTS

In this section, we present and discuss our computational results that exemplify the advantages of considering the known total amount of each class. We analyze this on different test sets from the literature. The test sets are discussed in Section 4.1, while the computational setup is described in Section 4.2. The evaluation criteria are depicted in Section 4.3. Finally, the numerical results are discussed in Section 4.4.

4.1. Tests Sets. For the computational analysis of the proposed approach, we consider the subset of instances presented by Olson et al. (2017) that are applicable to classification problems and that have at most three classes. Repeated instances are eliminated, and all instances are reduced to complete cases only. If an instance contains three classes, we convert them into two classes, such that the class with label 1 represents the class \mathcal{A} and the other two classes represent the class \mathcal{B} . This results in a final test set of 97 instances, as listed in Table 1. To avoid numerical instabilities, all data sets are scaled as follows. For each coordinate $j \in [1, p]$, we compute

$$l_j = \min_{i \in [1, N]} \{x_j^i\}, \quad u_j = \max_{i \in [1, N]} \{x_j^i\}, \quad m_j = 0.5(l_j + u_j)$$

and shift each coordinate j of all data points x^i via $\bar{x}_j^i = x_j^i - m_j$. Furthermore, if a coordinate j of the re-scaled points is still large, i.e., if $\bar{l}_j = l_j - m_j < -10^2$ or $\bar{u}_j = u_j - m_j > 10^2$ holds, it is re-scaled via

$$\tilde{x}_j^i = (\bar{v} - \underline{v}) \frac{\bar{x}_j^i - \bar{l}_j}{\bar{u}_j - \bar{l}_j} + \bar{v}$$

with $\bar{v} = 10^2$ and $\underline{v} = -10^2$. The corresponding 10 instances that we re-scale are marked with an asterisk in Table 1.

Table 1: Overview over the entire test set with number of points (N) and dimension (p)

ID	Instance	N	p
1	prnm_synth	250	2
2*	analcata_data_asbestos	73	3
3*	lupus	87	3
4	analcata_data_boxing1	120	3
5	analcata_data_boxing2	132	3
6	haberman	289	3
7	analcata_data_happiness	60	3
8*	analcata_data_aids	50	4
9	analcata_data_lawsuit	263	4
10	iris	147	4
11	hayes_roth	93	4
12	balance_scale	625	4
13	parity5	32	5
14*	bupa	341	5
15	irish	470	5
16	phoneme	5349	5
17	tae	110	5
18	new_thyroid	215	5
19*	analcata_data_bankruptcy	50	6
20*	analcata_data_creditscore	100	6
21	mux6	64	6
22	monk3	357	6
23	monk1	432	6
24	monk2	432	6
25	appendicitis	106	7
26	prnm_crabs	200	7
27*	penguins	333	7
28	postoperative_patient_data	78	8

29*	biomed	209	8
30*	pima	768	8
31*	cars	392	8
32	analcatdata_japansolvent	52	9
33	glass2	162	9
34	breast_cancer	272	9
35	saheart	462	9
36	threeOf9	512	9
37	profb	672	9
38	breast_w	463	9
39	tic_tac_toe	958	9
40	xd6	512	9
41	cmc	1425	9
42	analcatdata_cyyoung9302	92	10
43	analcatdata_cyyoung8092	97	10
44	breast	691	10
45	flare	315	10
46	parity5+5	1024	10
47	magic	18905	10
48	analcatdata_fraud	42	11
49	heart_statlog	270	13
50	heart_h	293	13
51	hungarian	293	13
52*	cleve	302	13
53*	heart_c	302	13
54	wine_recognition	178	13
55*	australian	690	14
56*	adult	48790	14
57*	schizo	340	14
58*	buggyCrx	690	15
59	labor	57	16
60	house_votes_84	342	16
61	hepatitis	155	19
62*	credit_g	1000	20
63	gametes_e_0.1H	1599	20
64	gametes_e_0.4H	1600	20
65	gametes_e_0.2H	1600	20
66	gametes_h_50	1592	20
67	gametes_h_75	1599	20
68*	churn	5000	20
69*	ring	7400	20
70	twonorm	7400	20
71	waveform_21	5000	21
72	ann_thyroid	7129	21
73	spect	228	22
74	horse_colic	357	22
75	agaricus_lepiota	8124	22
76*	hypothyroid	3086	25
77*	dis	3711	29
78*	allhypo	3709	29
79*	allbp	3711	29
80*	breast_cancer_wisconsin	569	30

81	backache	180	32
82	ionosphere	351	34
83	chess	3196	36
84	waveform_40	5000	40
85	connect_4	67557	42
86	spectf	267	44
87*	tokyo1	959	44
88	molecular_biology_promoters	106	57
89*	spambase	4210	57
90	sonar	208	60
91	splice	2903	60
92	coil2000	8380	85
93*	Hill_Valley_without_noise	1212	100
94*	clean1	476	168
95*	clean2	6598	168
96	dna	3002	180
97	gametes_e_1000atts	1600	1000

In our computational study, we aim to emphasize the significance of cardinality constraints, particularly in the context of non-representative biased samples. Biased samples are frequently observed in non-probability surveys, which are surveys where the inclusion process is not monitored and, hence, the inclusion probabilities are unknown as well. Therefore, correction methods like inverse inclusion probability weighting are not applicable. For an understanding of inverse inclusion probability weighting, see Skinner and D’arrigo (2011) and references therein.

To simulate this scenario, we create 5 biased samples with 10% of the data being labeled for each instance. In contrast to a simple random sample, where each point has an equal probability of being chosen as labeled data, in the biased sample the labeled data are chosen with probability 85% for being on class \mathcal{A} . Then, for each instance, with a time limit of 7200 s each, we apply the methods listed in Section 4.2.

Additionally, in Appendix B, we provide the results under simple random sampling, which produces unbiased samples. We see that the results from the proposed methods are similar to OCT-H (Bertsimas and Dunn 2017) in this setting. Despite the extra computational costs, we do not observe any drawbacks by using S²OCT in more simple survey designs.

4.2. Computational Setup. For each one of the 485 instances described in Section 4.1, the following approaches are compared:

- (a) OCT-H as described in Bertsimas and Dunn (2017), where only labeled data are considered.
- (b) S²OCT as given in Problem (P2) with $B(s)$ as in Expression (7) and M as given in Theorem 2.

Our comparison has been implemented in Julia 1.8.5 and we use Gurobi 9.5.2 and JuMP (Dunning et al. 2017) to solve OCT-H as well as Problem (P2). All computations were executed on the high-performance cluster “Elwetritsch”, which is part of the “Alliance of High-Performance Computing Rheinland-Pfalz” (AHRP). We used a single Intel XEON SP 6126 core with 2.6 GHz and 64 GB RAM.

To give the same importance to labeled and unlabeled data, we set the parameter $C = 1$ in S²OCT. Furthermore, we set the complexity parameter $\alpha = 0$ in OCT-H. Also, as required by OCT-H, all points x^i belong to $[0, 1]^d$. For this to hold, we

re-scaled the data as discussed in Section 4.1, with the difference that $\tilde{l}_j < 0$ and $\tilde{u}_j > 1$.

Theorem 2 establishes a relationship between s and M . To keep M at least 500 and s sufficiently large, from preliminary numerical tests we set, in S²OCT,

$$s = \begin{cases} \max\{10, 499/(\eta\sqrt{d})\}, & \text{if } N \in [1, 650), \\ \max\{20, 499/(\eta\sqrt{d})\}, & N \in [650, 1500). \\ \max\{40, 499/(\eta\sqrt{d})\}, & \text{otherwise.} \end{cases}$$

By default, MIP solvers such as Gurobi aim to achieve a balance between exploring new feasible solutions and verifying the optimality of the current solution. In preliminary numerical tests, we observed that the solver required significant time to find feasible solutions. Therefore, we selected Gurobi's parameter `MIPFocus` = 1, i.e., the solver focuses more on finding feasible solutions. Moreover, D in OCT-H and in S²OCT are fixed as

$$D = \begin{cases} 2, & \text{if } N \in [1, 1000), \\ 3, & \text{otherwise.} \end{cases}$$

4.3. Evaluation Criteria. The first evaluation criterion is the run time of OCT-H and S²OCT. To compare run times, we use empirical cumulative distribution functions (ECDFs). Specifically, for S being a set of solvers (or approaches as above) and for \bar{P} being a set of problems, we denote by $t_{\bar{p},s} \geq 0$ the run time of the approach $s \in S$ applied to the problem $\bar{p} \in \bar{P}$ in seconds. If $t_{\bar{p},s} > 7200$, we consider problem \bar{p} as not being solved by approach s . With these notations, the performance profile of approach s is the graph of the function $\gamma_s : [0, \infty) \rightarrow [0, 1]$ given by

$$\gamma_s(\sigma) = \frac{1}{|\bar{P}|} |\{\bar{p} \in \bar{P} : t_{\bar{p},s} \leq \sigma\}|.$$

Furthermore, since the true labels of all points are known in the simulation, we categorize them into four distinct categories: true positive (TP) or true negative (TN) if the point is classified correctly in classes \mathcal{A} or \mathcal{B} , respectively, as well as false positive (FP) if the point is misclassified in the class \mathcal{A} and as false negative (FN) if the point is misclassified in the class \mathcal{B} . Using this information, we calculate two classification metrics. The first one is accuracy (AC). It measures the proportion of correctly classified points and is given by

$$\text{AC} := \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \in [0, 1]. \quad (8)$$

Observe that for AC the greater the value, the better the classification. The second metric is Matthews correlation coefficient (MCC). It measures the correlation coefficient between the observed and predicted classifications and is computed by

$$\text{MCC} := \frac{\text{TP} \times \text{TN} - \text{FP} \times \text{FN}}{\sqrt{(\text{TP} + \text{FP})(\text{TP} + \text{FN})(\text{TN} + \text{FP})(\text{TN} + \text{FN})}} \in [-1, 1]. \quad (9)$$

As for accuracy, the higher the MCC, the better the classification. The main question is the following: For a specific instance, does S²OCT or OCT-H have a higher accuracy and MCC? Hence, we compute the instance-wise difference of the accuracy and MCC according to

$$\overline{\text{AC}} := \text{AC}_{\text{S}^2\text{OCT}} - \text{AC}_{\text{OCT-H}} \quad \overline{\text{MCC}} := \text{MCC}_{\text{S}^2\text{OCT}} - \text{MCC}_{\text{OCT-H}}, \quad (10)$$

where $\text{AC}_{\text{OCT-H}}$ and $\text{AC}_{\text{S}^2\text{OCT}}$ are computed as in (8), and $\text{MCC}_{\text{OCT-H}}$ and $\text{MCC}_{\text{S}^2\text{OCT}}$ as in (9). To keep the numerical results section concise, we report on precision and recall in Appendix A.

4.4. Numerical Results.

TABLE 2. Different quantile values for the number of continuous and binary variables

	Continuous		Binary	
	OCT-H	S ² OCT	OCT-H	S ² OCT
min	31	43	42	151
25 %	61	195	123	846
50 %	97	366	226	1843
75 %	319	3028	1451	16 469
max	14 039	121 910	54 373	695 835

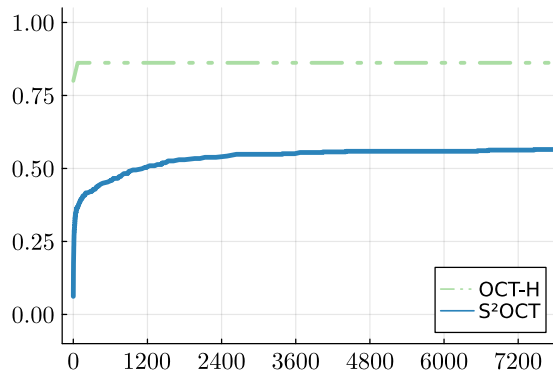


FIGURE 2. ECDFs for run times (in seconds).

4.4.1. *Run Time.* The number of continuous and binary variables is an important property for comparing different approaches. For this purpose, we computed the number of these variables for all instances presented in Section 4.1. Table 2 provides a quantile analysis of these quantities.

Observe that S²OCT has more variables than OCT-H. Therefore, it can be expected that OCT-H solves more instances than S²OCT within the time limit. However, note that the two approaches are designed for different purposes. Our approach considers labeled and unlabeled points together with the respective cardinality constraint, while OCT-H only deals with labeled points. Figure 2 shows ECDFs of run times of OCT-H and S²OCT. OCT-H solved 86 % of the instances within the time limit, while S²OCT does so for 58 %. As expected, OCT-H has significantly shorter run times.

4.4.2. *Accuracy and MCC.* Note that for both metrics, \overline{AC} and \overline{MCC} , a value greater than zero indicates that S²OCT had a better result than OCT-H and lower than zero indicates that OCT-H had a better result than S²OCT. Moreover, the box in the boxplot depicts the range of the medium 50 % of the values; 25 % of the values are below and 25 % are above the box. As can be seen in Figure 3, the \overline{AC} values are greater than zero in 75 % of the results (rows 1 and 2). Therefore, our proposed method takes advantage of the additional information on the total number of cases for the classes and has a better accuracy than OCT-H. When comparing all instances (column 1), the negative outliers indicate worse accuracy for S²OCT than OCT-H in some cases. This happens because in some instances our method does not terminate within the time limit while OCT-H does. Since for those instances that

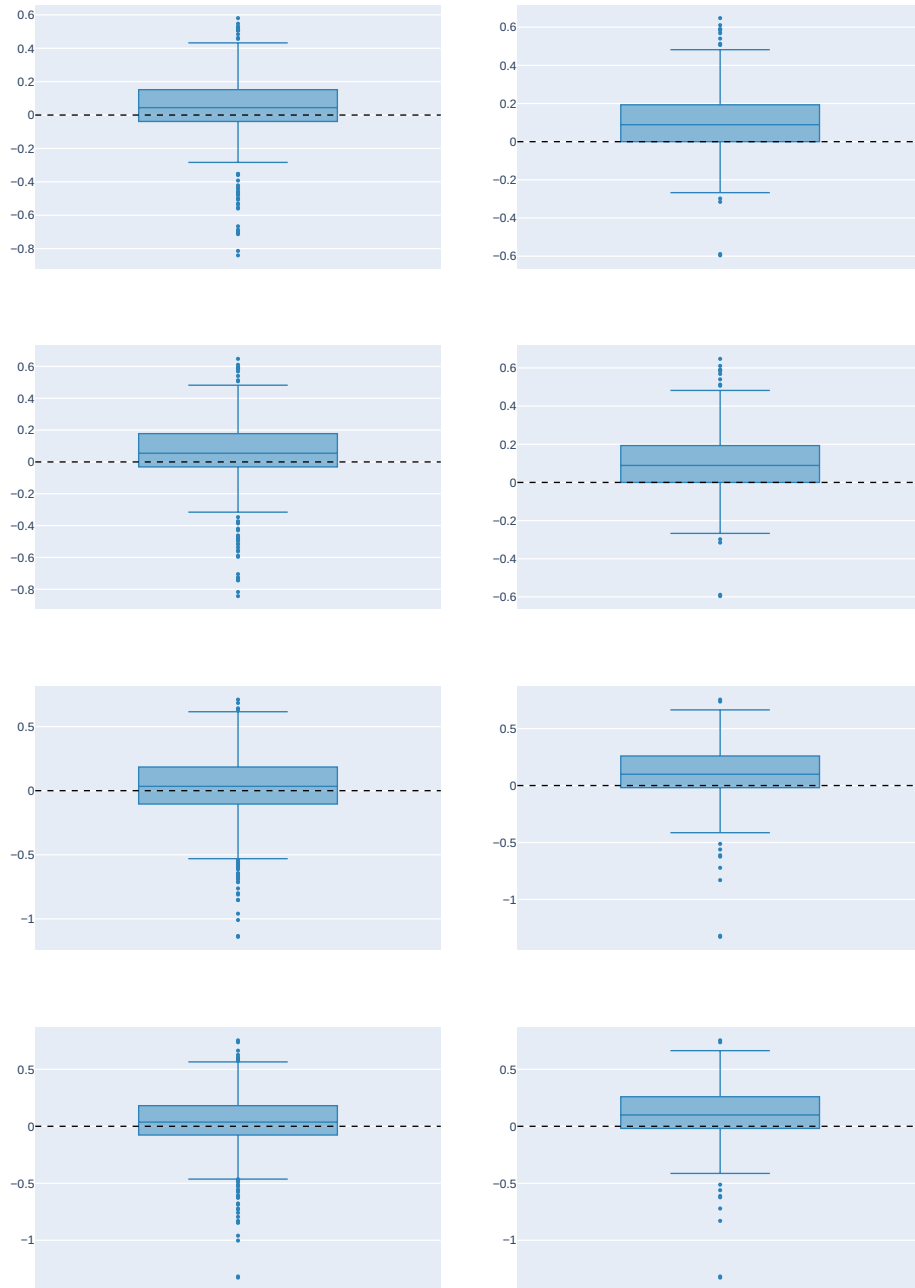


FIGURE 3. First row: Comparison of accuracy \overline{AC} as described in (10) for the entire data set. Second row: Comparison of \overline{AC} for unlabeled data. Third row: Comparison of precision \overline{MCC} as described in (10) for the entire data set. Last row: Comparison of \overline{MCC} for unlabeled data. Left: Comparison for all instances. Right: Comparison only for those instances for which both approaches terminate within the time limit.

terminate within the time limit (column 2), we have few outliers in accuracy (rows 1 and 2), we expect that the number of instances with lower precision will decrease if we would increase the time limit. Figure 3 also shows that the $\overline{\text{MCC}}$ values are greater than zero in most cases (rows 3 and 4), especially when comparing only the instances that terminate in the time limit (column 2). This means that our method has a better MCC than OCT-H. The consequences of the results so far are that using the unlabeled points as well as the cardinality constraint allows to correctly classify the points with higher accuracy and better MCC than with the optimal decision tree approach OCT-H. Moreover, further numerical tests revealed that if the percentage of labeled points is decreased, OCT-H tends to decrease in accuracy and MCC, while the deterioration for $S^2\text{OCT}$ is much less pronounced. This is especially relevant as in typical social surveys the sample proportion is seldomly over 1% of the population.

5. CONCLUSION

In many classification problems, acquiring labels for the entire population of interest can be expensive. Fortunately, external sources oftentimes can provide aggregated information on how many points are in each class. For this context, we proposed an MILP model for semi-supervised multivariate OCTs that considers the setting of labeled and unlabeled data points as well as additional aggregated information for the unlabeled data for a binary classification.

Under the condition of simple random sampling, our proposed approach has a slightly better accuracy and a better MCC than the conventional optimal classification tree. In many applications, however, the available data is coming from non-probability samples, where the data collection mechanism is largely unknown. Assuming simple random sampling in this setting is at least optimistic. Consequently, there is the risk of obtaining biased samples. Our numerical results show that our model has better accuracy, MCC, and precision than the existing approach from the literature, even with a small number of labeled points and biased samples. As expected, the drawback of introducing the cardinality constraint is that we get larger computational costs.

For future work, we will adapt our approach to a multiclass OCT. Furthermore, more research is needed to further reduce the computational burden.

ACKNOWLEDGEMENTS

The authors thank the DFG for their support within RTG 2126 “Algorithmic Optimization”.

REFERENCES

- Altıncay, H. (2007). “Decision trees using model ensemble-based nodes.” In: *Pattern Recognition* 40, pp. 3540–3551. DOI: [10.1016/j.patcog.2007.03.023](https://doi.org/10.1016/j.patcog.2007.03.023).
- Amini, M.-R. and P. Gallinari (2002). “Semi-Supervised Logistic Regression.” In: *Proceedings of the 15th European Conference on Artificial Intelligence*. ECAI’02. Lyon, France: IOS Press, pp. 390–394.
- Beale, E. and J. Tomlin (1969). “Special facilities in a general mathematical programming system for nonconvex problems using ordered sets of variables.” In: *Operational Research* 69, pp. 447–454.
- Bennett, K. P. and J. A. Blue (1996). “Optimal Decision Trees.” In: *Rensselaer Polytechnic Institute Math Report* 214.
- Bertsimas, D. and J. Dunn (2017). “Optimal classification trees.” In: *Machine Learning* 106.7, pp. 1039–1082. DOI: [10.1007/s10994-017-5633-9](https://doi.org/10.1007/s10994-017-5633-9).

- Blanco, V., A. Japón, and J. Puerto (2022a). “Robust optimal classification trees under noisy labels.” In: *Advances in Data Analysis and Classification* 16.1, pp. 155–179. DOI: [10.1007/s11634-021-00467-2](https://doi.org/10.1007/s11634-021-00467-2).
- Blanco, V., A. Japón, and J. Puerto (2022b). “A mathematical programming approach to SVM-based classification with label noise.” In: *Computers & Industrial Engineering* 172, p. 108611. DOI: [10.1016/j.cie.2022.108611](https://doi.org/10.1016/j.cie.2022.108611).
- Blanquero, R., E. Carrizosa, C. Molero-Río, and D. Romero Morales (2019). “Sparsity in optimal randomized classification trees.” In: *European Journal of Operational Research* 284.1, pp. 255–272. DOI: [10.1016/j.ejor.2019.12.002](https://doi.org/10.1016/j.ejor.2019.12.002).
- (2021). “Optimal randomized classification trees.” In: *Computers & Operations Research* 132, p. 105281. DOI: [10.1016/j.cor.2021.105281](https://doi.org/10.1016/j.cor.2021.105281).
- Breiman, L., J. H. Friedman, R. A. Olshen, and C. J. Stone (1984). *Classification and Regression Trees*. Monterey, CA: Wadsworth and Brooks.
- Burgard, J. P., J. Krause, and S. Schmaus (2021). “Estimation of regional transition probabilities for spatial dynamic microsimulations from survey data lacking in regional detail.” In: *Computational Statistics & Data Analysis* 154, p. 107048. DOI: [10.1016/j.csda.2020.107048](https://doi.org/10.1016/j.csda.2020.107048).
- Burgard, J. P., M. E. Pinheiro, and M. Schmidt (2023). *Mixed-Integer Quadratic Optimization and Iterative Clustering Techniques for Semi-Supervised Support Vector Machines*. arXiv: [2303.12532v2](https://arxiv.org/abs/2303.12532v2) [[math.OC](https://arxiv.org/abs/2303.12532v2)].
- Bzdok, D., M. Eickenberg, O. Grisel, B. Thirion, and G. Varoquaux (2015). “Semi-Supervised Factored Logistic Regression for High-Dimensional Neuroimaging Data.” In: *Advances in Neural Information Processing Systems*. Ed. by C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett. Vol. 28. Curran Associates, Inc.
- Carrizosa, E., C. Molero-Río, and D. R. Morales (2021). “Mathematical optimization in classification and regression trees.” In: *TOP: An Official Journal of the Spanish Society of Statistics and Operations Research* 29.1, pp. 5–33. DOI: [10.1007/s11750-021-00594-1](https://doi.org/10.1007/s11750-021-00594-1).
- Chapelle, O., M. Chi, and A. Zien (2006). “A Continuation Method for Semi-Supervised SVMs.” In: *Proceedings of the 23rd International Conference on Machine Learning*. ICML ’06. New York, NY, USA: Association for Computing Machinery, pp. 185–192. DOI: [10.1145/1143844.1143868](https://doi.org/10.1145/1143844.1143868).
- Cortes, C. and V. Vapnik (1995). “Support Vector Networks.” In: *Machine Learning* 20, pp. 273–297. DOI: [10.1007/BF00994018](https://doi.org/10.1007/BF00994018).
- D’Onofrio, F., G. Grani, M. Monaci, and L. Palagi (2023). *Margin Optimal Classification Trees*. arXiv: [2210.10567](https://arxiv.org/abs/2210.10567) [[math.OC](https://arxiv.org/abs/2210.10567)].
- Dunning, I., J. Huchette, and M. Lubin (2017). “JuMP: A Modeling Language for Mathematical Optimization.” In: *SIAM Review* 59.2, pp. 295–320. DOI: [10.1137/15M1020575](https://doi.org/10.1137/15M1020575).
- Gambella, C., B. Ghaddar, and J. Naoum-Sawaya (2021). “Optimization problems for machine learning: A survey.” In: *European Journal of Operational Research* 290.3, pp. 807–828. DOI: [10.1016/j.ejor.2020.08.045](https://doi.org/10.1016/j.ejor.2020.08.045).
- Kim, K. (2016). “A hybrid classification algorithm by subspace partitioning through semi-supervised decision tree.” In: *Pattern Recognition* 60, pp. 157–163. DOI: [10.1016/j.patcog.2016.04.016](https://doi.org/10.1016/j.patcog.2016.04.016).
- Kocev, M. C. D., J. Levatić, and S. Džeroski (2017). “Semi-supervised classification trees.” In: *Journal of Intelligent Information Systems* 49, pp. 461–486. DOI: [10.1007/s10844-017-0457-4](https://doi.org/10.1007/s10844-017-0457-4).
- Kotsiantis, S. (2014). “A hybrid decision tree classifier.” In: *Journal of Intelligent & Fuzzy Systems: Applications in Engineering and Technology* 26, pp. 327–336. DOI: [10.3233/IFS-120741](https://doi.org/10.3233/IFS-120741).

- Lee, D.-H. (2013). “Pseudo-Label : The Simple and Efficient Semi-Supervised Learning Method for Deep Neural Networks.” In: *ICML 2013 Workshop : Challenges in Representation Learning (WREPL)*.
- McCormick, G. P. (1976). “Computability of Global Solutions to Factorable Non-convex Programs: Part I – Convex Underestimating Problems.” In: *Mathematical Programming* 10.1, pp. 147–175. DOI: [10.1007/BF01580665](https://doi.org/10.1007/BF01580665).
- Melacci, S. and M. Belkin (2009). “Laplacian Support Vector Machines Trained in the Primal.” In: *Journal of Machine Learning Research* 12. DOI: [10.48550/ARXIV.0909.5422](https://doi.org/10.48550/ARXIV.0909.5422).
- Nguyen, T. N. N., B. Veeravalli, and X. Fong (2023). “A Semi-Supervised Learning Method for Spiking Neural Networks Based on Pseudo-Labeling.” In: *2023 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–7. DOI: [10.1109/IJCNN54540.2023.10191317](https://doi.org/10.1109/IJCNN54540.2023.10191317).
- Oliver, A., A. Odena, C. A. Raffel, E. D. Cubuk, and I. Goodfellow (2018). “Realistic Evaluation of Deep Semi-Supervised Learning Algorithms.” In: *Advances in Neural Information Processing Systems*. Vol. 31. Curran Associates, Inc. DOI: [10.48550/arXiv.1804.09170](https://doi.org/10.48550/arXiv.1804.09170).
- Olson, R. S., W. La Cava, P. Orzechowski, R. J. Urbanowicz, and J. H. Moore (2017). “PMLB: a large benchmark suite for machine learning evaluation and comparison.” In: *BioData Mining* 10.36, pp. 1–13. DOI: [10.1186/s13040-017-0154-4](https://doi.org/10.1186/s13040-017-0154-4).
- Orsenigo, C. and C. Vercellis (2003). “Multivariate classification trees based on minimum features discrete support vector machines.” In: *IMA Journal of Management Mathematics* 14.3, pp. 221–234. DOI: [10.1093/imaman/14.3.221](https://doi.org/10.1093/imaman/14.3.221).
- Quinlan, J. R. (1986). “Induction of Decision Trees.” In: *Machine Learning* 1, pp. 81–106. DOI: doi.org/10.1007/BF00116251.
- Santhiappan, S. and B. Ravindran (2021). “A Semi-Supervised Approach to Growing Classification Trees.” In: *Proceedings of the 3rd ACM India Joint International Conference on Data Science & Management of Data (8th ACM IKDD CODS & 26th COMAD)*. CODS-COMAD '21. Bangalore, India: Association for Computing Machinery, pp. 29–37. DOI: [10.1145/3430984.3431009](https://doi.org/10.1145/3430984.3431009).
- Skinner, C. J. and D’arrigo (2011). “Inverse probability weighting for clustered nonresponse.” In: *Biometrika* 98.4, pp. 953–966. DOI: [10.1093/biomet/asr058](https://doi.org/10.1093/biomet/asr058).
- Tanha, J., M. van Someren, and H. Afsarmanesh (2017). “Semi-supervised self-training for decision tree classifiers.” In: *International Journal of Machine Learning and Cybernetics* 8, pp. 355–370. DOI: [10.1007/s13042-015-0328-7](https://doi.org/10.1007/s13042-015-0328-7).
- Verwer, S. and Y. Zhang (2019). “Learning Optimal Classification Trees Using a Binary Linear Program Formulation.” In: vol. 33. AAAI Press, pp. 1625–1632. DOI: [10.1609/aaai.v33i01.33011624](https://doi.org/10.1609/aaai.v33i01.33011624).
- Yildiz, O. and O. Dikmen (2007). “Parallel Univariate Decision Trees.” In: *Pattern Recognition Letters* 28, pp. 825–832. DOI: [10.1016/j.patrec.2006.11.009](https://doi.org/10.1016/j.patrec.2006.11.009).
- Zharmagambetov, A. and M. A. Carreira-Perpinan (2022). “Semi-Supervised Learning with Decision Trees: Graph Laplacian Tree Alternating Optimization.” In: *Advances in Neural Information Processing Systems*. Vol. 35, pp. 2392–2405.
- Zhu, X. and A. B. Goldberg (2009). *Introduction to Semi-Supervised Learning*. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool Publishers. DOI: [10.2200/S00196ED1V01Y200906AIM006](https://doi.org/10.2200/S00196ED1V01Y200906AIM006).

APPENDIX A. FURTHER NUMERICAL RESULTS

Besides the measures of accuracy and MCC, we compare two further measures that, depending on the application, can be highly relevant. The first metric is precision (PR). It measures the proportion of correctly classified points among all

positively classified points and is thus defined as

$$\text{PR} := \frac{\text{TP}}{\text{TP} + \text{FP}} \in [0, 1]. \quad (11)$$

This quantity is important in some application such as for fraud detection systems, where identifying legitimate transactions as fraudulent is better than identify fraudulent transactions as legitimate. Moreover, precision can be higher when there are more positives in the dataset.

Second, we consider recall (RE), which quantifies the proportion of positive instances that are correctly classified as positive. It is formally given by

$$\text{RE} := \frac{\text{TP}}{\text{TP} + \text{FN}} \in [0, 1]. \quad (12)$$

This quantity is important in some applications such as in cancer diagnosis, where evaluating recall is relevant as it is more significant to identify potential cancer cases than to do not. Different from precision, recall can be higher when there are more negatives in the dataset.

As accuracy and MCC, the main question is how much better precision and recall of S²OCT are compared to the one of the OCT-H. Hence, we compute the difference of the precision and recall according to

$$\overline{\text{PR}} := \text{PR}_{\text{S}^2\text{OCT}} - \text{PR}_{\text{OCT-H}} \quad \overline{\text{RE}} := \text{RE}_{\text{S}^2\text{OCT}} - \text{RE}_{\text{OCT-H}}, \quad (13)$$

where $\text{PR}_{\text{OCT-H}}$ and $\text{PR}_{\text{S}^2\text{OCT}}$ are computed as in (11) for the OCT-H and S²OCT, respectively. In the same way, $\text{RE}_{\text{OCT-H}}$ and $\text{RE}_{\text{S}^2\text{OCT}}$ are computed as in (12) for the OCT-H and S²OCT. Note that as in Section 4, for both $\overline{\text{PR}}$ and $\overline{\text{RE}}$, a value greater than zero indicates that S²OCT has a better result than OCT-H and lower than zero indicates that S²OCT has a worse result than OCT-H. As can be seen in Figure 4, the $\overline{\text{PR}}$ values are greater than zero in more than 75 % of the results (rows 1 and 2). This means that S²OCT classifies the points with higher precision than OCT-H. Hence, OCT-H has more false-positive results. The negative outliers most likely are due to the same reason as those for the respective $\overline{\text{AC}}$ and $\overline{\text{MCC}}$ values.

On the other hand, Figure 4 also show that $\overline{\text{RE}}$ is, in general, lower than 0. This means that OCT-H has better recall than our method. The results of precision and recall can be justified by the fact that the biased sample is more likely to have labeled data in class \mathcal{A} and having no information about the unlabeled data, the OCT-H ends up classifying points on the positive side.

APPENDIX B. NUMERICAL RESULTS FOR SIMPLE RANDOM SAMPLES

Our computational study in Section 4 focuses on the analysis of non-representative biased samples. The typical baseline scenario for evaluating the performance of estimators is to apply them on simple random samples. Therefore, to complement our numerical results, we also present the results under simple random sampling. In a simple random sampling, each unit in the data set has the same probability $\pi_i = n/N$ to be included in the sample of labeled data of size n . The instances are the same as described in Section 4.1. The computational setup follows the description in Section 4.2. As before, the used evaluation criteria are $\overline{\text{AC}}$ and $\overline{\text{MCC}}$ as in (10) and $\overline{\text{PR}}$ and $\overline{\text{RE}}$ as in (13).

Figure 5 shows that for all the instances (column 1), $\overline{\text{AC}}$ (rows 1 and 2) and $\overline{\text{MCC}}$ (rows 3 and 4) have value greater than 0 and lower than 0 in 50 % of the cases. This means both approaches have similar accuracy and MCC. However, when comparing only those instances that terminate within the time limit (column 2), it can be seen that S²OCT has slightly better accuracy and MCC, but not as much as for biased samples; see Section 4.4.2. This is expected because the sample is not

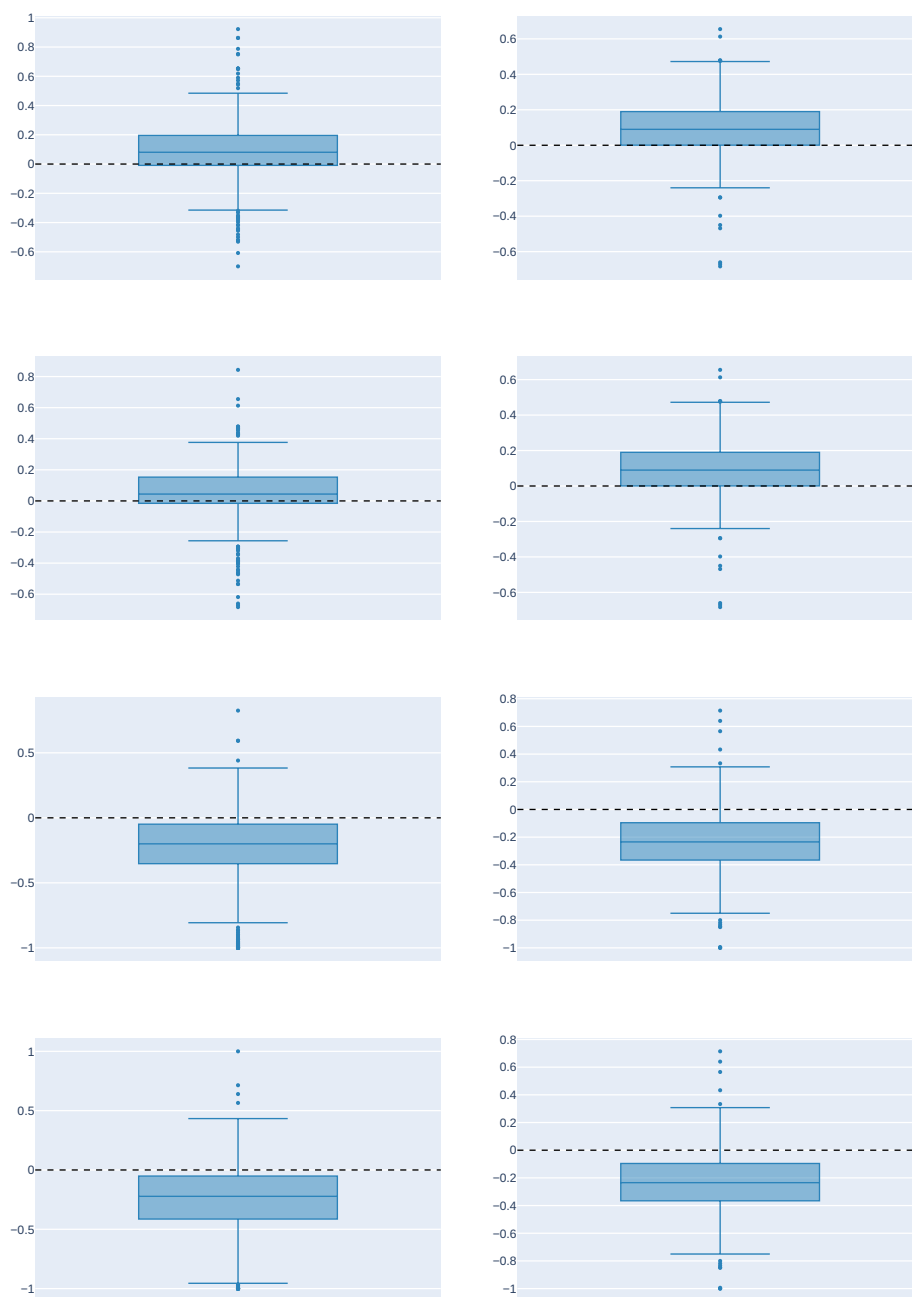


FIGURE 4. First row: Comparison of precision \overline{PR} as described in (13) for the entire data set. Second row: Comparison of \overline{PR} for unlabeled data. Third row: Comparison of recall \overline{RE} as described in (13) for the entire data set. Last row: Comparison of \overline{RE} for unlabeled data. Left: Comparison for all instances. Right: Comparison only for those instances for which both approaches terminate within the time limit.

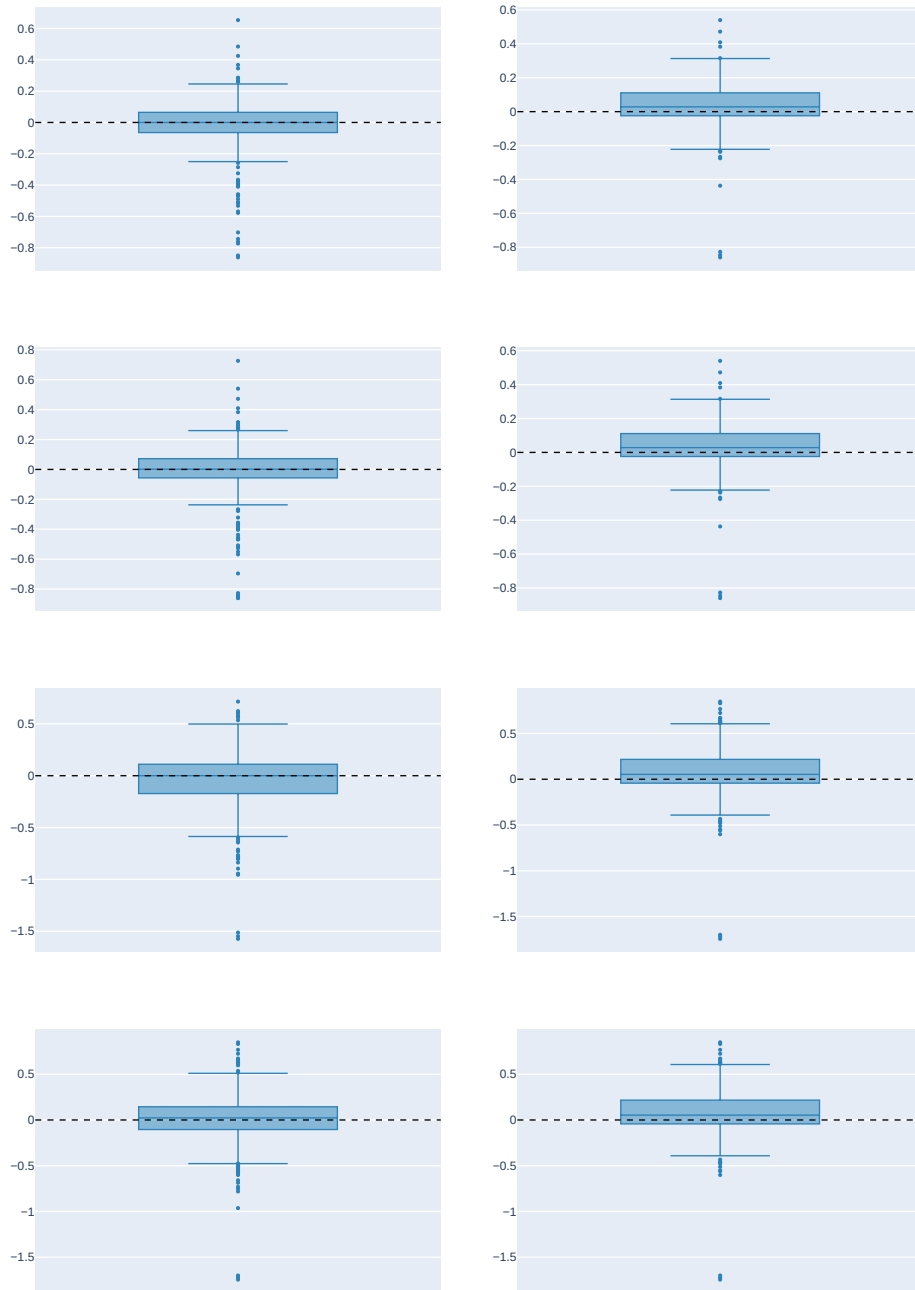


FIGURE 5. First row: Comparison of accuracy \overline{AC} as described in (10) for the entire data set. Second row: Comparison of \overline{AC} for unlabeled data. Third row: Comparison of precision \overline{MCC} as described in (10) for the entire data set. Last row: Comparison of \overline{MCC} for unlabeled data. Left: Comparison for all instances. Right: Comparison only for those instances for which both approaches terminate within the time limit.

biased. Consequently, the cardinality constraint, which aims to balance the class distribution, does not introduce additional meaningful information to the problem. As can be seen in Figure 6, precision and recall are similar for both approaches. Therefore, for the simple random samples, our approach has almost the same results as OCT-H, with slight improvements in accuracy and MCC.

(J. P. Burgard) TRIER UNIVERSITY, DEPARTMENT OF ECONOMIC AND SOCIAL STATISTICS,
UNIVERSITÄTSRING 15, 54296 TRIER, GERMANY
Email address: `burgardj@uni-trier.de`

(M. E. Pinheiro, M. Schmidt) TRIER UNIVERSITY, DEPARTMENT OF MATHEMATICS, UNIVER-
SITÄTSRING 15, 54296 TRIER, GERMANY
Email address: `pinheiro@uni-trier.de`
Email address: `martin.schmidt@uni-trier.de`

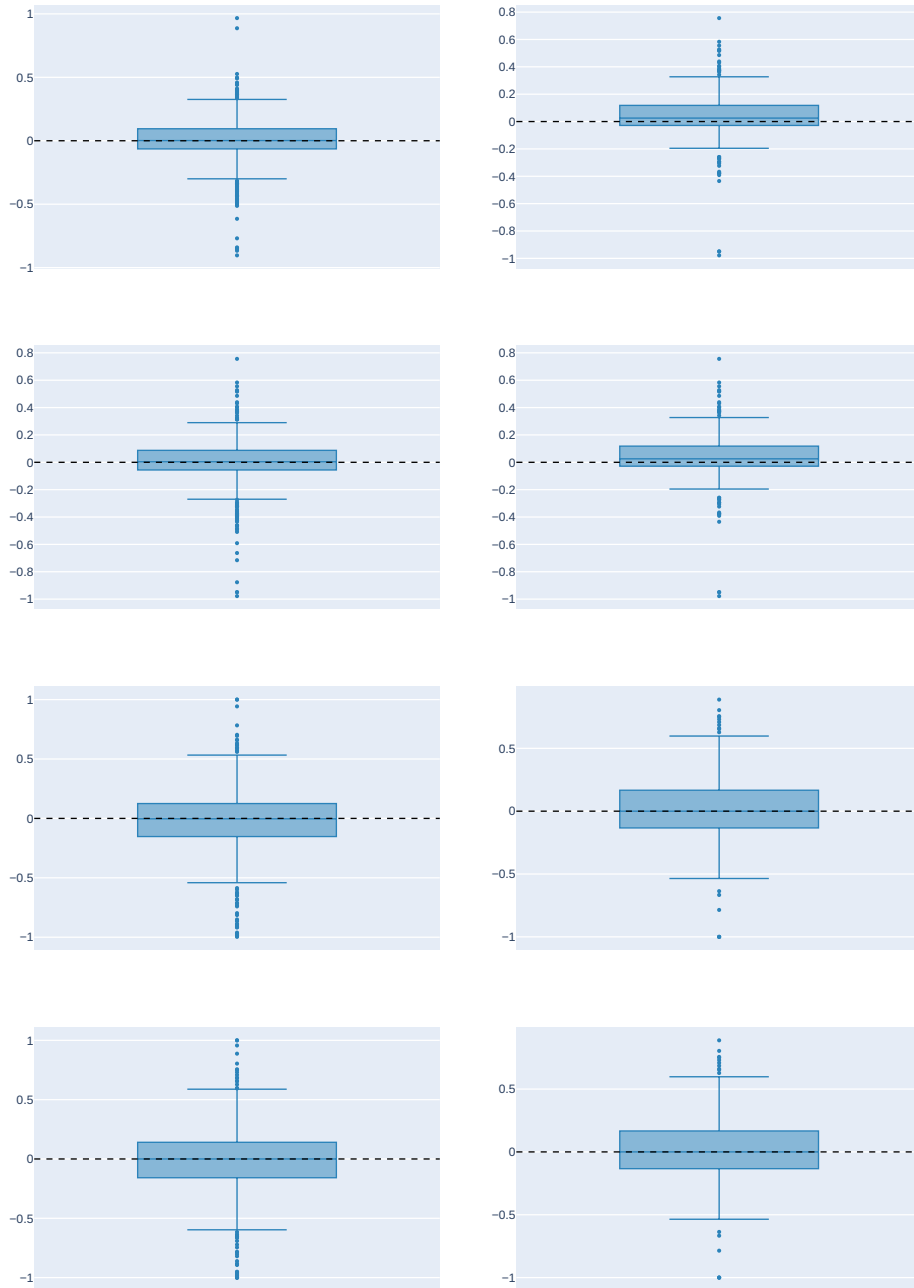


FIGURE 6. First row: Comparison of precision \overline{PR} as described in (13) for the entire data set. Second row: Comparison of \overline{PR} for unlabeled data. Third row: Comparison of recall \overline{RE} as described in (13) for the entire data set. Last row: Comparison of \overline{RE} for unlabeled data. Left: Comparison for all instances. Right: Comparison only for those instances for which both approaches terminate within the time limit.

Paper 3

Mixed-Integer Linear Optimization for Cardinality-Constrained Random Forests

Jan Pablo Burgard, Maria Eduarda Pinheiro, Martin Schmidt

Preprint under review

<https://arxiv.org/abs/2405.09832>

MIXED-INTEGER LINEAR OPTIMIZATION FOR CARDINALITY-CONSTRAINED RANDOM FORESTS

JAN PABLO BURGARD, MARIA EDUARDA PINHEIRO, MARTIN SCHMIDT

ABSTRACT. Random forests are among the most famous algorithms for solving classification problems, in particular for large-scale data sets. Considering a set of labeled points and several decision trees, the method takes the majority vote to classify a new given point. In some scenarios, however, labels are only accessible for a proper subset of the given points. Moreover, this subset can be non-representative, e.g., due to collection bias. Semi-supervised learning considers the setting of labeled and unlabeled data and often improves the reliability of the results. In addition, it can be possible to obtain additional information about class sizes from undisclosed sources. We propose a mixed-integer linear optimization model for computing a semi-supervised random forest that covers the setting of labeled and unlabeled data points as well as the overall number of points in each class for a binary classification. Since the solution time rapidly grows as the number of variables increases, we present some problem-tailored preprocessing techniques and an intuitive branching rule. Our numerical results show that our approach leads to a better accuracy and a better Matthews correlation coefficient for biased samples compared to random forests by majority vote, even if only few labeled points are available.

1. INTRODUCTION

Random forests are one of the most famous approaches in supervised learning (Breiman 2001). It has been applied to various fields such as the prediction of diseases (Gupta et al. 2021; Pal and Parija 2021), 3D object recognition (Shotton et al. 2011) and Fraude and accident detection (Dogru and Subasi 2018; Xuan et al. 2018). The main reasons why random forests are popular are that they prevent over-fitting (Hastie et al. 2009), that they have only a few parameters to tune, and that they can be used directly for high-dimensional problems (Biau and Scornet 2016; Cutler et al. 2012). The core idea is, given labeled data, to combine the prediction of different trees, in general, using the majority vote to classify new points.

Nevertheless, acquiring labels for every unit of interest can be costly—in particular when classic surveys are used to obtain the labels. In this situation, it would be beneficial to train the random forest with only partly labeled data. This yields a semi-supervised learning setting (Zhu and Goldberg 2009). Algorithms for semi-supervised learning have already been proposed for neural networks (Lee 2013; Nguyen et al. 2023; Oliver et al. 2018), logistic regression (Amini and Gallinari 2002; Bzdok et al. 2015), support vector machines (Chapelle et al. 2006; Melacci and Belkin 2009), and decision trees (Kim 2016; Kocev et al. 2017; Zharmagambe-tov and Carreira-Perpinan 2022).

In the case of random forests, Leistner et al. (2009) propose an iterative and deterministic annealing-like training algorithm that maximizes the multi-class margin

Date: May 16, 2024.

2020 Mathematics Subject Classification. 90C11,90C90,90-08,68T99.

Key words and phrases. Random forests, Semi-supervised learning, Mixed-integer linear optimization, Preprocessing, Cardinality constraints.

of labeled and unlabeled samples. Furthermore, Li and Zhou (2007) extend the co-training paradigm to random forests, determining how certain the model is about its predictions for unlabeled data. Moreover, Zhang et al. (2019) combine active learning and semi-supervised learning to improve the final classification performance of random forests by utilizing supervised clustering to categorize the unlabeled data.

However, in many applications, it is possible to know the total amount of elements in each class within a population, e.g., when external sources provide this information. For instance, a company might only know the overall number of successful transactions, but might not be able to identify which specific customer’s transactions were successful. An intuitive example is an online retailer that may track the total number of good customer reviews but does not have access to individual ratings due to anonymity practices. Another example is from healthcare, where it is possible to know how many patients have a disease but, due to data privacy reasons, one does not know which specific person is affected or not. Burgard et al. (2021) propose aggregating this extra information for logistic regression. They develop a cardinality-constrained multinomial logit model. For support vector machines, Burgard et al. (2024b) present a mixed-integer quadratic optimization model and iterative clustering techniques to tackle cardinality constraints for each class. Moreover, for the case of decision trees, Burgard et al. (2024a) propose a mixed-integer linear optimization model for computing semi-supervised optimal classification trees that serve the same purpose.

Our contribution here is to propose a random forest model that imposes a cardinality constraint on the classification of the unlabeled data. We develop a big- M -based mixed-integer linear programming (MILP) model to solve the cardinality-constrained random forest (C^2RF) problem that includes the cardinality constraint for the unlabeled data. The cardinality constraint helps to account for biased samples since the number of predictions in each class on the population is bounded by the constraint. In particular, our numerical results show that our approach leads to a better accuracy and a better Matthews correlation coefficient for biased samples compared to random forests by majority vote, even if only few labeled points are available. The computation time for this MILP grows with the number of variables—especially for an increasing number of integer variables. To account for this, we present theoretical results that lead to preprocessing techniques that significantly reduce the computation time.

This paper is organized as follows. In Section 2 we present the optimization model and prove the correctness of the used big- M parameter. Afterward, the preprocessing techniques are discussed in Section 3 and an intuitive branching rule is presented in Section 4. There, we also present our algorithm that combines the mentioned techniques and the MILP formulation. In Section 5 we report and discuss numerical results. Finally, we conclude in Section 6.

2. AN MILP FORMULATION FOR CARDINALITY-CONSTRAINED RANDOM FORESTS

Let $X \in \mathbb{R}^{p \times N} = [X_u, X_l]$ be the data matrix with unlabeled data $X_u = [x^1, \dots, x^m]$ and labeled data $X_l = [x^{m+1}, \dots, x^N]$. Hence, we are given points $x^i \in \mathbb{R}^p$ for all $i \in [1, N] := \{1, \dots, N\}$. We set $n := N - m$ and $y \in \{-1, 1\}^n$ as the vector of class labels for the labeled data. Let t be the number of given decision trees and let $A^j \in \mathbb{R}^{p \times d}$ be a subset of the labeled data with size d for $j \in [1, t]$. For each $j \in [1, t]$, based on each column of A^j and its label, the j th tree generates a vector $r^j \in \{-1, 1\}^m$ that classifies the unlabeled data X_u . Thus, for each unlabeled point x^i we observe a vector of classification $r_i = [r_i^1, \dots, r_i^t] \in \{-1, 1\}^t$. Hence, $R = [r_1, \dots, r_m] \in \{-1, 1\}^{t \times m}$ and r_i^j is the classification of $x^i \in X_u$ given

by the tree j . In a random forest, the prediction for a point $x^i \in X_u$ is the dominant class chosen by the individual t trees, i.e., the majority vote.

In many applications, aggregated information on the labels is available, e.g., from census data. For what follows, we assume to know the total number $\lambda \in \mathbb{N}$ of unlabeled points that belong to the positive class and propose a model such that we can use a linear combination of the tree classifications as well as λ as an additional information. Our goal is to find optimal parameters $\alpha^* \in \mathbb{R}^t$, $\eta^* \in \mathbb{R}$, and $z^* \in \{0, 1\}^m$ that solve the optimization problem

$$\min_{\alpha, \eta, z} \eta \quad (\text{P1a})$$

$$\text{s.t. } \alpha^\top r_i \leq -1 + z_i M, \quad i \in [1, m], \quad (\text{P1b})$$

$$\alpha^\top r_i \geq 1 - (1 - z_i)M, \quad i \in [1, m], \quad (\text{P1c})$$

$$\lambda - \eta \leq \sum_{i=1}^m z_i \leq \lambda + \eta, \quad (\text{P1d})$$

$$\ell \leq \alpha_j \leq u, \quad j \in [1, t], \quad (\text{P1e})$$

$$0 \leq \eta \leq \bar{\eta}, \quad (\text{P1f})$$

$$z_i \in \{0, 1\}, \quad i \in [1, m], \quad (\text{P1g})$$

where M needs to be chosen sufficiently large, $u > \ell > 0$ holds, and we set

$$\bar{\eta} := \max \{ \lambda, m - \lambda \}. \quad (1)$$

Note that the objective function in (P1a) minimizes the classification error for the unlabeled data. As z_i is binary, Constraints (P1b) and (P1c) lead to

$$\alpha^\top r_i \geq 1 \implies z_i = 1, \quad i \in [1, m],$$

$$\alpha^\top r_i \leq -1 \implies z_i = 0, \quad i \in [1, m].$$

Constraint (P1d) ensures that the number of unlabeled data points classified as positive is as close to λ as possible. Constraint (P1e) bounds the weight of each tree's decision for the final classification. This means that for $j \in [1, t]$, as α_j gets closer to u , the j th tree gets greater influence on the final classification, and as α_j gets closer to ℓ , the j th tree has less influence on the final classification. Observe that since $\alpha_j \geq \ell > 0$ holds for all $j \in [1, t]$, all trees contribute to the final classification. Moreover, if α_j has the same value for all $j \in [1, t]$, all trees contribute equally to the final classification and we are in the standard random forest setup with majority vote. Note that the upper bound u is not necessary for the correctness of the model but will serve as a big- M -type parameter as can be seen in Proposition 1 below. The upper bound $\bar{\eta}$ in Constraint (P1f) is also not necessary for the correctness of Model (P1). Nevertheless, as can be seen in Proposition 2, this upper bound does not cut off any solution. Hence, we include it in our implementation because we expect that the solution process benefits from tight bounds. Problem (P1) is an MILP. We refer to this problem as C²RF (Cardinality-Constrained Random Forest). As usual for big- M formulations, the choice of M is crucial. If M is too small, the problem can become infeasible or optimal solutions could be cut off. If M is chosen too large, the respective continuous relaxations usually lead to bad lower bounds and solvers may encounter numerical troubles. The choice of M is discussed in the following proposition.

Proposition 1. *A valid big- M for Problem (P1) is given by $M = ut + 1$, i.e., M is linear in the number of trees in the forest.*

Proof. For all $i \in [1, m]$ we have $r_i \in \{-1, 1\}^t$. Moreover, Constraint (P1e) ensures that $\alpha_j \leq u$ holds for all $j \in [1, t]$. Hence,

$$\alpha^\top r_i \leq \sum_{j=1}^t \alpha_j \leq ut$$

and

$$\alpha^\top r_i \geq -\sum_{j=1}^t \alpha_j \geq -ut$$

hold for all $i \in [1, m]$ and $M = ut + 1$ does not cut any feasible solution. \square

The following proposition makes a statement about the upper bound $\bar{\eta}$ in Constraint (P1f).

Proposition 2. *Consider Problem (P1) in which Constraint (P1f) is replaced by $\eta \geq 0$. Then, for every η^* as being part of an optimal solution, it holds $\eta^* \leq \bar{\eta}$ for $\bar{\eta}$ as defined in (1).*

Proof. Observe that since $z_i \in \{0, 1\}$ for all $i \in [1, m]$,

$$0 \leq \sum_{i=1}^m z_i \leq m$$

holds. Moreover, the maximum value occurs if all points are classified as positive. If this happens, from Constraint (P1d) we obtain

$$\eta \geq \sum_{i=1}^m z_i - \lambda = m - \lambda.$$

On the other hand, the minimum value of $\sum_{i=1}^m z_i$ occurs if all points are classified as negative. If this happens, from Constraint (P1d) we obtain

$$\eta \geq \sum_{i=1}^m z_i + \lambda = \lambda.$$

Since Problem (P1) is a minimization Problem, $\eta \leq \bar{\eta}$ holds. Thus, the upper bound $\bar{\eta}$ in Constraint (P1f) does not cut off any optimal point. \square

3. PREPROCESSING

In this section, we present different preprocessing techniques for Problem (P1) that can be used to reduce the number of binary as well as the number of continuous variables.

The first insight is that, if all trees have the same classification for some unlabeled points, these points must have the same final classification and, therefore, the respective binary variables always have the same values.

Proposition 3. *Let $k \in [1, m]$ and consider $\mathcal{K} := \{i \in [1, m] : r_i = r_k\}$. Then, (α, η, z) is a feasible point of Problem (P1) if and only if there exists a vector $\bar{z} \in \{0, 1\}^{m+1-|\mathcal{K}|}$ such that (α, η, \bar{z}) is a feasible point of the problem*

$$\min_{\alpha, \eta, z} \eta \quad (\text{P2a})$$

$$s.t. \quad \alpha^\top r_i \leq -1 + z_i M, \quad i \in \{k\} \cup [1, m] \setminus \mathcal{K}, \quad (\text{P2b})$$

$$\alpha^\top r_i \geq 1 - (1 - z_i)M, \quad i \in \{k\} \cup [1, m] \setminus \mathcal{K}, \quad (\text{P2c})$$

$$\lambda - \eta \leq \sum_{i \in [1, m] \setminus \mathcal{K}} z_i + |\mathcal{K}| z_k \leq \lambda + \eta, \quad (\text{P2d})$$

$$(\text{P1e}), (\text{P1f}) \quad (\text{P2e})$$

$$z_i \in \{0, 1\}, \quad i \in \{k\} \cup [1, m] \setminus \mathcal{K}. \quad (\text{P2f})$$

Proof. Consider (α, η, z) a feasible point of (P1) and

$$\bar{z}_i = z_i, \quad i \in \{k\} \cup [1, m] \setminus \mathcal{K}.$$

We now prove that (α, η, \bar{z}) is a feasible point of (P2). Because (P1b), (P1c), and (P1g) hold, (P2b), (P2c), and (P2f) are satisfied. Moreover, since for all $i \in \mathcal{K}$ it holds $r_i = r_k$, we obtain that $\alpha^\top r_k = \alpha^\top r_i$ holds for all $i \in \mathcal{K}$. Hence, by Constraint (P1b) and (P1c), we obtain that $z_i = z_k$ also holds for all $i \in \mathcal{K}$. This together with $\bar{z}_k = z_k$ implies that $|\mathcal{K}| \bar{z}_k = \sum_{i \in \mathcal{K}} z_i$ is satisfied. Hence,

$$\sum_{i \in [1, m] \setminus \mathcal{K}} \bar{z}_i + |\mathcal{K}| \bar{z}_k = \sum_{i \in [1, m] \setminus \mathcal{K}} z_i + \sum_{i \in \mathcal{K}} z_i = \sum_{i=1}^m z_i \quad (2)$$

is also satisfied, and, by Constraint (P1d), we obtain that Constraint (P2d) holds. Therefore, (α, η, \bar{z}) is a feasible point of Problem (P2).

On the other hand, let (α, η, \bar{z}) be a feasible point of Problem (P2) and set

$$z_i = \begin{cases} \bar{z}_i, & \text{if } i \notin \mathcal{K}, \\ \bar{z}_k, & \text{otherwise.} \end{cases} \quad (3)$$

Since (P2b), (P2c) and (P2f) are satisfied, (P1b) and (P1c) hold for each $i \notin \mathcal{K}$ and (P1g) holds for all $i \in [1, m]$. Further, because each $i \in \mathcal{K}$ satisfies $r_i = r_k$, $\alpha^\top r_i = \alpha^\top r_k$ holds for all $i \in \mathcal{K}$. Hence, by Constraints (P2b) and (P2c) we obtain that

$$1 - (1 - z_i)M = 1 - (1 - \bar{z}_k)M \leq \alpha^\top r_i \leq -1 + \bar{z}_k M = -1 + z_i M$$

is satisfied for all $i \in \mathcal{K}$. Besides that, Expression (3) implies that $|\mathcal{K}| \bar{z}_k = \sum_{i \in \mathcal{K}} z_i$ and, therefore, Expression (2) also holds. Hence, by Constraint (P2d), we obtain that Constraint (P1d) is satisfied. Therefore, (α, η, z) is a feasible point of Problem (P1). \square

The following proposition shows that if one or more trees classify all points exactly as another tree, some continuous variables of Problem (P1) can be eliminated.

Proposition 4. *Given $g \in [1, t]$, consider $\mathcal{G} := \{j \in [1, t]: r^g = r^j\}$. Then, (α^*, η^*, z^*) is a solution to Problem (P1) if and only if there exists a vector $\bar{\alpha} \in \mathbb{R}^{t+1-|\mathcal{G}|}$ such that $(\bar{\alpha}, \eta^*, z^*)$ is a solution to the problem*

$$\min_{\alpha, \eta, z} \eta \quad (\text{P3a})$$

$$s.t. \quad \sum_{j \in [1, t] \setminus \mathcal{G}} \alpha_j r_i^j + |\mathcal{G}| \alpha_g r_i^g \leq -1 + z_i M, \quad i \in [1, m], \quad (\text{P3b})$$

$$\sum_{j \in [1, t] \setminus \mathcal{G}} \alpha_j r_i^j + |\mathcal{G}| \alpha_g r_i^g \geq 1 - (1 - z_i) M, \quad i \in [1, m], \quad (\text{P3c})$$

$$(\text{P1d}), \quad (\text{P3d})$$

$$\ell \leq \alpha_j \leq u, \quad j \in \{g\} \cup [1, t] \setminus \mathcal{G}, \quad (\text{P3e})$$

$$(\text{P1f}), (\text{P1g}). \quad (\text{P3f})$$

Proof. Let (α^*, η^*, z^*) be a solution to Problem (P1) and

$$\bar{\alpha}_j = \begin{cases} \alpha_j^*, & \text{if } j \notin \mathcal{G}, \\ \sum_{j \in \mathcal{G}} \alpha_j^* / |\mathcal{G}|, & \text{otherwise.} \end{cases}$$

Since $\ell \leq \alpha_j^* \leq u$ holds for all $j \in [1, t]$, we obtain

$$\ell = \frac{\ell}{|\mathcal{G}|} (|\mathcal{G}|) \leq \bar{\alpha}_g \leq \frac{u}{|\mathcal{G}|} (|\mathcal{G}|) = u.$$

Moreover, because $r^g = r^j$ is satisfied for all $j \in \mathcal{G}$,

$$\sum_{j \in \mathcal{G}} \alpha_j^* r_i^j = r_i^g \sum_{j \in \mathcal{G}} \alpha_j^* = |\mathcal{G}| \bar{\alpha}_g r_i^g \quad (4)$$

holds for all $i \in [1, m]$. Hence, for all $i \in [1, m]$,

$$\sum_{j \in [1, t] \setminus \mathcal{G}} \bar{\alpha}_j r_i^j + |\mathcal{G}| \bar{\alpha}_g r_i^g = \sum_{j \in [1, t] \setminus \mathcal{G}} \alpha_j^* r_i^j + \sum_{j \in \mathcal{G}} \alpha_j^* r_i^j = (\alpha^*)^\top r_i \quad (5)$$

is satisfied and, consequently,

$$1 - (1 - z_i^*) M \leq \sum_{j \in [1, t] \setminus \mathcal{G}} \bar{\alpha}_j r_i^j + |\mathcal{G}| \bar{\alpha}_g r_i^g \leq -1 + z_i^* M$$

holds for $i \in [1, m]$. Therefore, $(\bar{\alpha}, \eta^*, z^*)$ is a solution to Problem (P3).

On the other hand, let $(\bar{\alpha}, \eta^*, z^*)$ be a solution to Problem (P3) and set

$$\alpha_j^* = \begin{cases} \bar{\alpha}_j, & \text{if } j \notin \mathcal{G}, \\ \bar{\alpha}_g, & \text{otherwise.} \end{cases}$$

Since (P3e) holds, (P1e) is satisfied for all $j \in [1, t]$. Besides that, since $r^g = r^j$ is satisfied for all $j \in \mathcal{G}$, (4) and (5) also hold for all $i \in [1, m]$. Hence, for all $i \in [1, m]$, we have

$$1 - (1 - z_i^*) M \leq (\alpha^*)^\top r_i \leq -1 + z_i^* M$$

and (α^*, η^*, z^*) is a solution to Problem (P1). \square

Finally, the following proposition allows to fix some binary variables z_i of Problem (P1).

Proposition 5. *For each $i \in [1, m]$, consider $\mathcal{A}_i = \{j \in [1, t] : r_i^j = -1\}$ and $\mathcal{B}_i = \{j \in [1, t] : r_i^j = 1\}$. If for some $i \in [1, m]$,*

$$\varphi_i := -u|\mathcal{A}_i| + \ell|\mathcal{B}_i| \geq 1 \quad (6)$$

holds, then every feasible point (α, η, z) of Problem (P1) satisfies $z_i = 1$. If, on the other hand,

$$\phi_i := -\ell|\mathcal{A}_i| + u|\mathcal{B}_i| \leq -1 \quad (7)$$

is satisfied for some $i \in [1, m]$, then any feasible point (α, η, z) of Problem (P1) satisfies $z_i = 0$.

Proof. Since $\ell \leq \alpha_j \leq u$ is satisfied for all $j \in [1, \ell]$, if for some $i \in [1, m]$,

$$-u|\mathcal{A}_i| + \ell|\mathcal{B}_i| \geq 1$$

holds, we obtain

$$\alpha^\top r_i = - \sum_{j \in \mathcal{A}_i} \alpha_j + \sum_{j \in \mathcal{B}_i} \alpha_j \geq -u|\mathcal{A}_i| + \ell|\mathcal{B}_i| \geq 1,$$

and by Constraint (P1b), $z_i = 1$. On the other hand, if for some $i \in [1, m]$, it holds

$$-\ell|\mathcal{A}_i| + u|\mathcal{B}_i| \leq -1,$$

we get

$$\alpha^\top r_i = - \sum_{j \in \mathcal{A}_i} \alpha_j + \sum_{j \in \mathcal{B}_i} \alpha_j \leq -\ell|\mathcal{A}_i| + u|\mathcal{B}_i| \leq -1,$$

and by Constraint (P1c), $z_i = 0$. \square

Consider now

$$\mathcal{P} := \{i \in [1, m]: \varphi_i \geq 1\} \quad \text{and} \quad \mathcal{N} := \{i \in [1, m]: \phi_i \leq -1\}.$$

Then, $|\mathcal{P}| + |\mathcal{N}|$ binary variables can be fixed. Moreover, $|\mathcal{P}|$ points then are already classified as positive. If $|\mathcal{P}| \geq \lambda$, due to cardinality constraint, all remaining points $x^i \in X_u \setminus (\mathcal{P} \cup \mathcal{N})$ must be classified as negative, and λ can be set to 0. On the other hand, if $|\mathcal{P}| < \lambda$, only $\lambda - |\mathcal{P}|$ points in $X_u \setminus (\mathcal{P} \cup \mathcal{N})$ must be classified as positive. This update is present in Step 20 in Algorithm 1 below.

4. BRANCHING PRIORITIES

One aspect that can significantly affect the performance of MILP solvers is the applied branching rule. In this brief section, we present a problem-tailored rule for helping the MILP code to solve Problem (P1). To this end, let us consider binary variables $z_i, z_k \in \{0, 1\}$, $i, k \in [1, m]$, and positive integer values ξ_i and ξ_k so that $\xi_i > \xi_k$ implies that the solver should branch on z_i before z_k . In our context, a point for which the percentage of trees that classify the point as positive (or negative) is larger than for another point seems to be “easier” to classify. Hence, we want to branch on the respective binary variable first. Based on that, we establish a criterion for a branching strategy. We set $\theta_i = \lfloor \text{mean}(r_i) \rfloor$ for each $x^i \in X_u \setminus (\mathcal{P} \cup \mathcal{N})$. Observe that the higher the value of θ_i , the more trees classify the point x^i in one specific class. Hence, we consider ξ_i the position of θ_i in the vector of the increasingly sorted values of θ . Thus, the higher the value of θ_i , the higher the value of ξ_i , and hence, the higher the branching priority for the binary variable z_i .

Motivated by the preprocessing techniques presented in Section 3 and the branching priority discussed in this section, we obtain Algorithm 1 to solve Problem (P1).

5. NUMERICAL RESULTS

In this section, we present and discuss our computational results that demonstrate the impact of considering the total amount of points in each class and of using the preprocessing techniques as well as the branching rule to speed up the solution process.

We illustrate this on different test sets from the literature. The test sets are discussed in Section 5.1, while the computational setup is described in Section 5.2. The evaluation criteria are depicted in Section 5.3. Finally, the numerical results are discussed in Section 5.4 and 5.5.

Algorithm 1: p-C²RF: Preprocessing and Solving C²RF

Input : $R \in \{-1, 1\}^{t \times m}$, $u > \ell > 0$, $\lambda \in \mathbb{N}$, $\mathcal{K} = \emptyset$, $\beta = 0$, and $\gamma = 0$.

- 1 Compute $M = ut + 1$ and $\bar{R} = [\bar{r}_1, \dots, \bar{r}_h] \in \{-1, 1\}^{t \times h}$ being the set of all different $r_i \in R$.
- 2 **for** $k \in \{1, \dots, h\}$ **do**
- 3 Compute $w_k = |\{i \in [1, m] : r_i = \bar{r}_k\}|$, φ_k as described in (6), and ϕ_k as described in (7).
- 4 **if** $\varphi_k \geq 1$ **then**
- 5 Set $\mathcal{K} \leftarrow \mathcal{K} \cup \{k\}$ and $\beta \leftarrow \beta + w_k$.
- 6 **else if** $\phi_k \leq -1$ **then**
- 7 Set $\mathcal{K} \leftarrow \mathcal{K} \cup \{k\}$ and $\gamma \leftarrow \gamma + w_k$.
- 8 **end**
- 9 **end**
- 10 Compute $S = [s^1, \dots, s^q]^\top \in \{-1, 1\}^{q \times h}$ being the set of all different $\bar{r}^j \in \bar{R}$.
- 11 **for** $g \in \{1, \dots, q\}$ **do**
- 12 Compute $v_g = |\{j \in [1, t] : r^j = \bar{r}^g\}|$ and set $s^g \leftarrow v_g s^g$.
- 13 **end**
- 14 **for** $i \in \{1, \dots, h\} \setminus \mathcal{K}$ **do**
- 15 Compute $\theta_i = |\text{mean}(s_i)|$.
- 16 **end**
- 17 **for** $i \in \{1, \dots, h\} \setminus \mathcal{K}$ **do**
- 18 Compute ξ_i , i.e., the position of θ_i in the vector of the increasingly sorted values of θ .
- 19 **end**
- 20 Compute $\bar{\lambda} = \min\{0, \lambda - \beta\}$ and $\bar{\eta} = \max\{\bar{\lambda}, m - \beta - \gamma - \bar{\lambda}\}$ and solve

$$\begin{aligned}
& \min_{\alpha, \eta, z} \quad \eta \\
& \text{s.t.} \quad \alpha^\top s_i \leq -1 + z_i M, \quad i \in [1, h] \setminus \mathcal{K}, \\
& \quad \alpha^\top s_i \geq 1 - (1 - z_i)M, \quad i \in [1, h] \setminus \mathcal{K}, \\
& \quad \bar{\lambda} - \eta \leq \sum_{i \in [1, h] \setminus \mathcal{K}} w_i z_i \leq \bar{\lambda} + \eta, \\
& \quad \ell \leq \alpha_j \leq u, \quad j \in [1, q], \\
& \quad 0 \leq \eta \leq \bar{\eta}, \\
& \quad z_i \in \{0, 1\}, \quad i \in [1, h] \setminus \mathcal{K}.
\end{aligned}$$

with branching priorities ξ to compute α^*, η^*, z^* .

5.1. Test Sets. For the computational analysis of the proposed approaches, we consider the subset of instances presented by Olson et al. (2017) that are suitable for classification problems and that have at most three classes and at least 5000 points. Repeated instances are removed and instances with missing information are reduced to the observations without missing information. If three classes are given in an instance, we transform them into two classes such that the class with label 1 represents the positive class and the other two classes represent the negative class. This results in a final test set of 13 instances, as listed in Table 1. To avoid numerical instabilities, all data sets are scaled as follows. For each coordinate $j \in [1, p]$, we compute

$$l_j = \min_{i \in [1, N]} \{x_j^i\}, \quad u_j = \max_{i \in [1, N]} \{x_j^i\}, \quad m_j = 0.5(l_j + u_j)$$

TABLE 1. The entire test set with the number of points (N) and the dimension (p)

ID	Instance	N	p
1	phoneme	5349	5
2	magic	18 905	10
3*	adult	48 790	14
4*	churn	5000	20
5*	ring	7400	20
6	twonorm	7400	20
7	waveform_21	5000	21
8	ann_thyroid	7129	21
9	agaricus_lepiota	8124	22
10	waveform_40	5000	40
11	connect_4	67 557	42
12	coil2000	8380	85
13*	clean2	6598	168

and shift each coordinate j of all data points x^i via $\bar{x}_j^i = x_j^i - m_j$. Furthermore, if a coordinate j of the re-scaled points is still large, i.e., if $\bar{l}_j = l_j - m_j < -10^2$ or $\bar{u}_j = u_j - m_j > 10^2$ holds, it is re-scaled via

$$\tilde{x}_j^i = (\bar{v} - \underline{v}) \frac{\bar{x}_j^i - \bar{l}_j}{\bar{u}_j - \bar{l}_j} + \bar{v}$$

with $\bar{v} = 10^2$ and $\underline{v} = -10^2$. The corresponding 4 instances that we re-scale are marked with an asterisk in Table 1.

In our computational study, we focus on emphasizing the statistical importance of cardinality constraints—mainly in the case of non-representative biased samples. Biased samples are highly recurrent in non-probability surveys, which are surveys with an inclusion process that is not tracked and, hence, the inclusion probabilities are unknown. This means that correction methods such as inverse inclusion probability weighting cannot be applicable. For a primer on inverse inclusion probability weighting, we refer to Skinner and D’arrigo (2011) and the references therein.

To reproduce such a scenario, we create 5 biased samples with 1% of the data being labeled for each instance. Differently from a simple random sample, where each point has an equal probability of being chosen as labeled data, in these biased samples the labeled data is chosen with probability 85% for belonging to the positive class. Moreover, we consider $t = 20$ trees and for each $j \in [1, t]$, the size of the training subset A^j is 20% of the labeled data. For each training subset we use the Decision Tree package (Sadeghi et al. 2022) to generate r^j .

In addition, in Appendix A, we provide the results under simple random sampling, which produces unbiased samples. In this scenario, the results of the proposed methods are similar to the random forest. Hence, there is no downside to using the proposed method in case of an unknown sampling process.

5.2. Computational Setup. For each one of the 65 instances described in Section 5.1, we compare the following approaches.

- (a) RF: Random Forest by majority vote.
- (b) C²RF as given in Problem (P1) with $\bar{\eta}$ as defined in (1) and M from Proposition 1.
- (c) p-C²RF as described in Algorithm 1.

- (d) only PP: Algorithm 1 without the branching rule described in Step 18.
- (e) only BR: C²RF as given in Problem (P1) with the branching rule as described in Section 4 but without our problem-tailored preprocessing.

Our comparison has been implemented in Julia 1.8.5 and we use Gurobi 11.5 and JuMP (Dunning et al. 2017) to solve C²RF as well as the MILP in Algorithm 1. All computations were executed on the high-performance cluster “Elwetritsch”, which is part of the “Alliance of High-Performance Computing Rheinland-Pfalz” (AHRP). We use a single Intel XEON SP 6126 core with 2.6 GHz and 64 GB RAM as well as a time limit of 7200 s.

Based on our preliminary experiments, for C²RF and p-C²RF we set the bounds to $\ell = 1$ and $u = 100$. Moreover, we set the MIPFocus parameter of Gurobi to 3.

5.3. Evaluation Criteria. The first evaluation criterion is the run time of the different methods. To compare run times, we use empirical cumulative distribution functions (ECDFs). Specifically, for S being a set of solvers (or approaches as above) and for P being a set of problems, we denote by $t_{p,s} \geq 0$ the run time of the approach $s \in S$ applied to the problem $p \in P$ in seconds. If $t_{p,s} > 7200$, we consider problem p as not being solved by approach s . With these notations, the performance profile of approach s is the graph of the function $\gamma_s : [0, \infty) \rightarrow [0, 1]$ given by

$$\gamma_s(\sigma) = \frac{1}{|P|} |\{p \in P : t_{p,s} \leq \sigma\}|.$$

Moreover, knowing the true label of all points, we categorize them into four distinct categories: true positive (TP) or true negative (TN) if the point is classified correctly in the positive or negative class, respectively, as well as false positive (FP) if the point is misclassified in the positive class and as false negative (FN) if the point is misclassified in the negative class. Motivated by this, we compute two classification metrics, for which a higher value indicates a better classification. The first one is accuracy (AC). It measures the proportion of correctly classified points and is given by

$$\text{AC} := \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \in [0, 1]. \quad (8)$$

The second metric is Matthews correlation coefficient (MCC). It measures the correlation coefficient between the observed and predicted classifications and is computed by

$$\text{MCC} := \frac{\text{TP} \times \text{TN} - \text{FP} \times \text{FN}}{\sqrt{(\text{TP} + \text{FP})(\text{TP} + \text{FN})(\text{TN} + \text{FP})(\text{TN} + \text{FN})}} \in [-1, 1]. \quad (9)$$

The main statistical question is the following: For a specific instance, does using the cardinality constraint as additional information increase the accuracy and the MCC? Since C²RF p-C²RF, only PP and only BR solve the same problem, we only compare the difference of the accuracy and MCC according to

$$\overline{\text{AC}} := \text{AC}_{\text{p-C}^2\text{RF}} - \text{AC}_{\text{RF}}, \quad \overline{\text{MCC}} := \text{MCC}_{\text{p-C}^2\text{RF}} - \text{MCC}_{\text{RF}}, \quad (10)$$

where AC_{RF} and $\text{AC}_{\text{p-C}^2\text{RF}}$ are computed as in (8), and MCC_{RF} and $\text{MCC}_{\text{p-C}^2\text{RF}}$ as in (9).

5.4. Discussion of Run Times. Figure 1 shows the ECDFs for the measured run times. As expected, RF is the fastest algorithm because it does not include any binary variable related to the unlabeled points as the newly proposed models do. It can be seen that p-C²RF significantly outperforms C²RF. C²RF solves only 58 % of the instances within the time limit, while p-C²RF solves 94 %. This shows that the preprocessing techniques and the branching rule significantly decrease the run times. However, by comparing the two lines for “only PP” and “only BR”, we

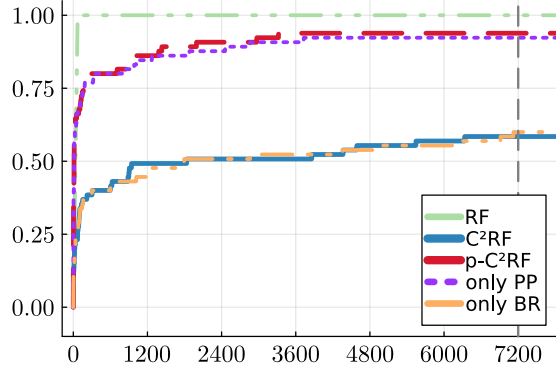


FIGURE 1. ECDFs for run times (in seconds)

TABLE 2. Median of run times (in seconds)

ID	RF	C ² RF	p-C ² RF	only PP	only BR
1	0.042	3859.72	2.939	3.000	1249.15
2	0.261	—	109.603	127.63	—
3	0.513	—	2.865	2.865	—
4	0.107	68.255	12.304	12.495	76.813
5	0.074	—	999.798	1018.56	—
6	0.069	—	—	—	—
7	0.172	1834.94	4.054	4.149	—
8	0.058	3.791	0.168	0.191	3.585
9	0.087	899.891	1.599	1.568	1018.80
10	0.066	883.18	144.127	182.252	—
11	1.129	—	204.337	147.684	—
12	0.194	70.102	14.443	12.619	75.937
13	0.229	0.986	0.275	0.367	1.124

see that most of the speed-up is obtained by the preprocessing techniques while the branching rule only helps to improve the performance for a small amount of instances. Since the branching rule is not harming and sometimes helps, we decide to include it in what follows.

In Table 2 we present the median run times of the 5 biased samples for each instance. A “—” means that the approach did not solve at least 3 of the samples of the instance within the time limit. One can see that RF almost always takes less than 1 s to solve the problem. When comparing the two novel approaches and only the instances that C²RF solves at least one sample, Table 2 shows that our techniques decrease the time computation by 89 % on average.

5.5. Discussion of Accuracy and MCC. Observe that for both metrics \overline{AC} and \overline{MCC} , a value greater than zero indicates that p-C²RF had a better result than RF. Besides that, the box in the boxplot depicts the range of the medium 50 % of the values; 25 % of the values are below and 25 % are above the box. Figure 2 shows that the \overline{AC} values are greater than zero in 75 % of the results (left plot). Hence, our proposed method has better accuracy than the conventional random forest. It can also be seen in Figure 2 that the \overline{MCC} values are greater than zero in most cases (right plot). Therefore, our method has a better MCC than RF.

FIGURE 2. Comparison of \overline{AC} (left) and \overline{MCC} (right); see (10)

TABLE 3. Median of AC and MCC (in percentage)

ID	Accuracy		MCC	
	RF	p-C ² RF	RF	p-C ² RF
1	62.16	72.51	69.28	69.20
2	65.14	75.03	70.35	73.13
3	76.28	77.78	55.66	67.16
4	61.98	79.64	55.17	57.09
5	50.80	60.20	54.20	61.45
6	58.90	66.93	65.76	67.05
7	75.88	78.59	78.55	76.47
8	98.58	98.74	84.03	84.73
9	81.30	87.59	83.95	87.57
10	61.35	71.13	71.14	70.00
11	24.79	56.69	52.19	55.98
12	89.10	88.72	54.61	53.57
13	99.43	100	98.89	100

When comparing each instance, Table 3 shows the median of AC and MCC of the 5 biased samples for RF and p-C²RF. It can be seen that, in the majority of instances, our approach has a greater value of accuracy and MCC than RF. Especially in terms of accuracy, we obtained a better median value in 12 of the 13 instances. Regarding MCC, our approach has a better median value than RF in 8 of the 13 instances. When RF has better MCC than p-C²RF, it is never better than 2.5 %.

Figure 2 and Table 3 show that using the cardinality constraint of each class as additional information allows to correctly classify the points with higher accuracy and better MCC than with the RF by majority vote.

6. CONCLUSION

For several classification problems, it can be expensive to acquire labels for the entire population of interest. Nevertheless, external sources can, in some cases, offer additional information on how many points are in each class. For the case of binary classification, we proposed a semi-supervised random forest that can be modeled using a big- M -based MILP formulation. We also presented problem-tailored pre-processing techniques and a branching rule to reduce the computational cost of solving the MILP model.

Under the condition of simple random sampling, our proposed semi-supervised method has very similar accuracy and MCC as a standard random forest. In many applications, however, the available data come from non-probability samples. In this case, the data collection mechanism is largely unknown and there is the risk of obtaining biased samples. Our numerical results show that our model has better accuracy and MCC than the conventional random forest even with a small number of labeled points and biased samples.

ACKNOWLEDGEMENTS

The authors thank the DFG for their support within RTG 2126 “Algorithmic Optimization”.

REFERENCES

- Amini, M.-R. and P. Gallinari (2002). “Semi-Supervised Logistic Regression.” In: *Proceedings of the 15th European Conference on Artificial Intelligence*. ECAI’02. Lyon, France: IOS Press, pp. 390–394.
- Biau, G. and E. Scornet (2016). “A random forest guided tour.” In: *TEST: An Official Journal of the Spanish Society of Statistics and Operations Research* 25.2, pp. 197–227. DOI: [0.1007/s11749-016-0481-7](https://doi.org/10.1007/s11749-016-0481-7).
- Breiman, L. (2001). “Random Forests.” In: *Machine Learning* 45.1, pp. 5–32. DOI: [10.1023/A:1010933404324](https://doi.org/10.1023/A:1010933404324).
- Burgard, J. P., J. Krause, and S. Schmaus (2021). “Estimation of regional transition probabilities for spatial dynamic microsimulations from survey data lacking in regional detail.” In: *Computational Statistics & Data Analysis* 154, p. 107048. DOI: [10.1016/j.csda.2020.107048](https://doi.org/10.1016/j.csda.2020.107048).
- Burgard, J. P., M. E. Pinheiro, and M. Schmidt (2024a). *Mixed-Integer Linear Optimization for Semi-Supervised Optimal Classification Trees*. arXiv: [2401.09848](https://arxiv.org/abs/2401.09848) [math.OC].
- (2024b). “Mixed-integer quadratic optimization and iterative clustering techniques for semi-supervised support vector machines.” In: *TOP*. To appear. DOI: [10.1007/s11750-024-00668-w](https://doi.org/10.1007/s11750-024-00668-w).
- Bzdok, D., M. Eickenberg, O. Grisel, B. Thirion, and G. Varoquaux (2015). “Semi-Supervised Factored Logistic Regression for High-Dimensional Neuroimaging Data.” In: *Advances in Neural Information Processing Systems*. Ed. by C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett. Vol. 28. Cambridge, MA, USA: MIT Press, 3348–3356.
- Chapelle, O., M. Chi, and A. Zien (2006). “A Continuation Method for Semi-Supervised SVMs.” In: *Proceedings of the 23rd International Conference on Machine Learning*. ICML ’06. New York, NY, USA: Association for Computing Machinery, pp. 185–192. DOI: [10.1145/1143844.1143868](https://doi.org/10.1145/1143844.1143868).
- Cutler, A., D. R. Cutler, and J. R. Stevens (2012). “Random Forests.” In: *Ensemble Machine Learning: Methods and Applications*. Ed. by C. Zhang and Y. Ma. New York, NY: Springer New York, pp. 157–175. DOI: [10.1007/978-1-4419-9326-7_5](https://doi.org/10.1007/978-1-4419-9326-7_5).
- Dogru, N. and A. Subasi (2018). “Traffic accident detection using random forest classifier.” In: *2018 15th learning and technology conference (L&T)*. IEEE, pp. 40–45. DOI: [10.1109/LT.2018.8368509](https://doi.org/10.1109/LT.2018.8368509).
- Dunning, I., J. Huchette, and M. Lubin (2017). “JuMP: A Modeling Language for Mathematical Optimization.” In: *SIAM Review* 59.2, pp. 295–320. DOI: [10.1137/15M1020575](https://doi.org/10.1137/15M1020575).

- Gupta, V. K., A. Gupta, D. Kumar, and A. Sardana (2021). “Prediction of COVID-19 confirmed, death, and cured cases in India using random forest model.” In: *Big Data Mining and Analytics* 4.2, pp. 116–123. DOI: [10.26599/BDMA.2020.9020016](https://doi.org/10.26599/BDMA.2020.9020016).
- Hastie, T., R. Tibshirani, and J. Friedman (2009). *The elements of statistical learning: data mining, inference and prediction*. 2nd ed. Springer. DOI: [10.1007/978-0-387-84858-7](https://doi.org/10.1007/978-0-387-84858-7).
- Kim, K. (2016). “A hybrid classification algorithm by subspace partitioning through semi-supervised decision tree.” In: *Pattern Recognition* 60, pp. 157–163. DOI: [10.1016/j.patcog.2016.04.016](https://doi.org/10.1016/j.patcog.2016.04.016).
- Kocev, M. C. D., J. Levatić, and S. Džeroski (2017). “Semi-supervised classification trees.” In: *Journal of Intelligent Information Systems* 49, pp. 461–486. DOI: [10.1007/s10844-017-0457-4](https://doi.org/10.1007/s10844-017-0457-4).
- Lee, D.-H. (2013). “Pseudo-Label : The Simple and Efficient Semi-Supervised Learning Method for Deep Neural Networks.” In: *ICML 2013 Workshop : Challenges in Representation Learning (WREPL)*.
- Leistner, C., A. Saffari, J. Santner, and H. Bischof (2009). “Semi-Supervised Random Forests.” In: *2009 IEEE 12th International Conference on Computer Vision*, pp. 506–513. DOI: [10.1109/ICCV.2009.5459198](https://doi.org/10.1109/ICCV.2009.5459198).
- Li, M. and Z.-H. Zhou (2007). “Improve Computer-Aided Diagnosis With Machine Learning Techniques Using Undiagnosed Samples.” In: *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans* 37.6, pp. 1088–1098. DOI: [10.1109/TSMCA.2007.904745](https://doi.org/10.1109/TSMCA.2007.904745).
- Melacchi, S. and M. Belkin (2009). “Laplacian Support Vector Machines Trained in the Primal.” In: *Journal of Machine Learning Research* 12. DOI: [10.48550/ARXIV.0909.5422](https://doi.org/10.48550/ARXIV.0909.5422).
- Nguyen, T. N. N., B. Veeravalli, and X. Fong (2023). “A Semi-Supervised Learning Method for Spiking Neural Networks Based on Pseudo-Labeling.” In: *2023 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–7. DOI: [10.1109/IJCNN54540.2023.10191317](https://doi.org/10.1109/IJCNN54540.2023.10191317).
- Oliver, A., A. Odena, C. A. Raffel, E. D. Cubuk, and I. Goodfellow (2018). “Realistic Evaluation of Deep Semi-Supervised Learning Algorithms.” In: *Advances in Neural Information Processing Systems*. Vol. 31. Curran Associates, Inc. DOI: [10.48550/arXiv.1804.09170](https://doi.org/10.48550/arXiv.1804.09170).
- Olson, R. S., W. La Cava, P. Orzechowski, R. J. Urbanowicz, and J. H. Moore (2017). “PMLB: a large benchmark suite for machine learning evaluation and comparison.” In: *BioData Mining* 10.36, pp. 1–13. DOI: [10.1186/s13040-017-0154-4](https://doi.org/10.1186/s13040-017-0154-4).
- Pal, M. and S. Parija (Mar. 2021). “Prediction of Heart Diseases using Random Forest.” In: *Journal of Physics: Conference Series* 1817.1, p. 012009. DOI: [10.1088/1742-6596/1817/1/012009](https://doi.org/10.1088/1742-6596/1817/1/012009).
- Sadeghi, B., P. Chiarawongse, K. Squire, D. C. Jones, A. Noack, C. St-Jean, R. Huijzer, R. Schätzle, I. Butterworth, Y.-F. Peng, and A. Blaom (Nov. 2022). *DecisionTree.jl - A Julia implementation of the CART Decision Tree and Random Forest algorithms*. Version 0.11.3. DOI: [10.5281/zenodo.7359268](https://doi.org/10.5281/zenodo.7359268).
- Shotton, J., A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake (2011). “Real-time human pose recognition in parts from single depth images.” In: *CVPR 2011*, pp. 1297–1304. DOI: [10.1109/CVPR.2011.5995316](https://doi.org/10.1109/CVPR.2011.5995316).
- Skinner, C. J. and D’arrigo (2011). “Inverse probability weighting for clustered nonresponse.” In: *Biometrika* 98.4, pp. 953–966. DOI: [10.1093/biomet/asr058](https://doi.org/10.1093/biomet/asr058).

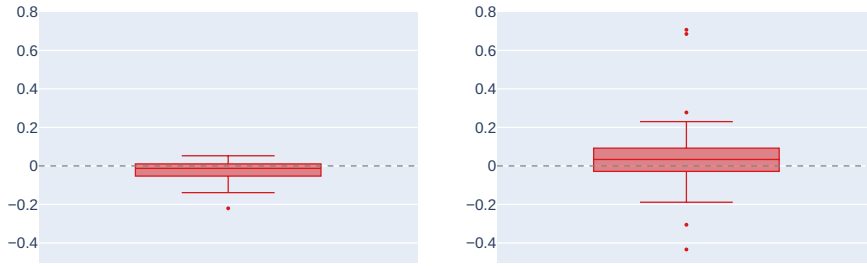


FIGURE 3. Comparison of \overline{AC} (left) and \overline{MCC} (right); see (10)

- Xuan, S., G. Liu, Z. Li, L. Zheng, S. Wang, and C. Jiang (2018). “Random forest for credit card fraud detection.” In: *2018 IEEE 15th International Conference on Networking, Sensing and Control (ICNSC)*, pp. 1–6. DOI: [10.1109/ICNSC.2018.8361343](https://doi.org/10.1109/ICNSC.2018.8361343).
- Zhang, Y., G. Cao, X. Li, B. Wang, and P. Fu (2019). “Active Semi-Supervised Random Forest for Hyperspectral Image Classification.” In: *Remote Sensing* 11.24. DOI: [10.3390/rs11242974](https://doi.org/10.3390/rs11242974).
- Zharmagambetov, A. and M. A. Carreira-Perpinan (2022). “Semi-Supervised Learning with Decision Trees: Graph Laplacian Tree Alternating Optimization.” In: *Advances in Neural Information Processing Systems*. Vol. 35, pp. 2392–2405.
- Zhu, X. and A. B. Goldberg (2009). *Introduction to Semi-Supervised Learning*. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool Publishers. DOI: [10.2200/S00196ED1V01Y200906AIM006](https://doi.org/10.2200/S00196ED1V01Y200906AIM006).

APPENDIX A. NUMERICAL RESULTS FOR SIMPLE RANDOM SAMPLES

In Section 5 we present a computational study on non-representative biased samples. To complement our numerical results, we also present the results for simple random sampling. For simple random sampling, each element in the data set has the same probability (n/N) to be included in the sample of labeled data of size n . The instances are the same as described in Section 5.1. The computational setup follows the description in Section 5.2. As before, the used evaluation criteria are \overline{AC} and \overline{MCC} as in (10).

It can be seen in Figure 3 that 75% of the values of \overline{AC} are between -0.05 and 0.05 (left plot). Figure 3 (right plot) also shows that \overline{MCC} has a value greater than 0 and lower than 0 in 50% of the cases.

Table 4 shows the median of AC and MCC for each instance for p-C²RF and RF. In the majority of instances, our approach has a better or a very similar accuracy and MCC compared to the conventional random forest. Especially in terms of MCC, this is the case for all 13 instances. From Figure 3 and Table 4 we can conclude that the accuracy and MCC of our proposed method p-C²RF and the standard random forest are very similar in the context of simple random sampling. This is expected because the cardinality constraint aims to balance the class distribution and since the sample is not biased, this constraint does not introduce additional meaningful information to the problem.

(J. P. Burgard) TRIER UNIVERSITY, DEPARTMENT OF ECONOMIC AND SOCIAL STATISTICS,
UNIVERSITÄTSRING 15, 54296 TRIER, GERMANY
Email address: burgardj@uni-trier.de

TABLE 4. Median of AC and MCC (in percentage)

ID	Accuracy		MCC	
	RF	p-C ² RF	RF	p-C ² RF
1	76.68	76.32	71.73	71.34
2	78.28	78.14	75.82	75.86
3	81.32	80.85	70.79	73.70
4	85.86	76.57	50.0	51.63
5	71.81	67.35	72.61	67.35
6	84.32	85.09	85.32	85.10
7	76.75	77.10	71.97	74.10
8	97.68	98.61	50.0	84.43
9	88.15	87.17	88.17	87.15
10	78.57	79.84	75.13	77.23
11	75.36	67.71	50.0	55.89
12	93.23	87.01	50.0	52.62
13	97.64	100	95.37	100

(M. E. Pinheiro, M. Schmidt) TRIER UNIVERSITY, DEPARTMENT OF MATHEMATICS, UNIVERSITÄTSRING 15, 54296 TRIER, GERMANY

Email address: pinheiro@uni-trier.de

Email address: martin.schmidt@uni-trier.de