Programa de Doctorado en Matemáticas

PHD DISSERTATION

ENHANCING CLASSIFICATION AND REGRESSION TREE-BASED MODELS BY MEANS OF MATHEMATICAL OPTIMIZATION

Author

Mª Cristina Molero del Río

Supervisors

Prof. Dr. *Rafael Blanquero Bravo*

Prof. Dr. *Emilio Carrizosa Priego*

Prof. Dr. *Dolores Romero Morales*

October 2022

This research has been partly financed by research projects:

* Optimización (FQM-329). Junta de Andalucía, Spain.
* Mathematical Optimization for Data Visualization and Decision Making (MTM2015-65915-R). Ministerio de Economía y Competitividad, Spain.
* Cost-sensitive classification. A mathematical optimization approach. COSECLA (FUNDBBVA/016/005). Fundación BBVA, Spain.
* Data Science: from mathematical modelling to mathematical optimization (P18-FR-2369). Junta de Andalucía, Spain.
* Mathematical Optimization for Data-Driven Decision-Making (PID2019-110886RB-I00). MCIN/AEI/10.13039/501100011033, Spain.
* Mathematical Optimization in Data Science: Towards Interpretability in Classification, Regression and Dimensionality Reduction (US-1381178). Junta de Andalucía, Spain.
* Research and Innovation Staff Exchange Network of European Data Scientists - NeEDS (H2020-EU RISE 822214). European Commission.

This support is gratefully acknowledged.

Resumen

Esta tesis combina las disciplinas de Investigación Operativa y Aprendizaje Automático a través del desarrollo de formulaciones de Optimización Matemática y algoritmos de resolución numérica para construir modelos basados en árboles de clasificación y regresión.

A diferencia de los árboles de clasificación y regresión clásicos, generados de manera heurística y voraz, construir un árbol a través de un problema de optimización nos permite incluir fácilmente propiedades estructurales globales deseables. En esta tesis, ilustramos esta flexibilidad para modelar los siguientes aspectos: sparsity, como sinónimo de interpretabilidad, controlando el número de coeficientes no nulos, el número de variables predictoras y, si son funcionales, la proporción de dominio usado en la predicción; un criterio social importante, la equidad del modelo, evitando predicciones que discriminen a algunos individuos por su etnia u otras características sensibles; y la sensibilidad al coste de grupos de riesgo, asegurando un rendimiento aceptable para ellos. Además, con este enfoque se obtiene de manera natural el impacto que las variables predictoras continuas tienen en la predicción de cada individuo, mejorando así la explicabilidad local de los modelos de clasificación y regresión basados en árboles.

Todos los enfoques propuestos en esta tesis se formulan a través de problemas de Optimización Continua que son escalables con respecto al tamaño de la muestra de entrenamiento, se estudian desde el punto de vista teórico, se evalúan en conjuntos de datos reales y son competitivos frente a los procedimientos habituales. Esto, junto a las buenas propiedades resumidas en el párrafo anterior, se ilustra a lo largo de los diferentes capítulos de esta tesis.

La tesis se estructura de la siguiente manera. El estado del arte sobre árboles de decisión (óptimos) se discute ampliamente en el Capítulo 1, mientras que los cuatro capítulos siguientes exponen nuestra metodología.

El Capítulo 2 introduce de forma detallada el marco general que hila los capítulos de esta tesis: un árbol aleatorizado con cortes oblicuos. En particular, presentamos nuestra propuesta para tratar problemas de clasificación, la cual construye la probabilidad de pertenencia a cada clase ajustada a cada individuo, a diferencia de las técnicas más populares existentes, en las que a todos los individuos en el mismo nodo hoja se les asigna la misma probabilidad. Se tratan con éxito preferencias en las tasas de clasificación en clases críticas mediante restricciones de sensibilidad al coste.

El Capítulo 3 extiende la metodología de clasificación del Capítulo 2 para tratar adicionalmente sparsity. Esto se modela mediante regularizaciones con normas poliédricas que se añaden a la función objetivo. Se estudian propiedades teóricas del árbol más sparse, y se demuestra nuestra habilidad para sacrificar un poco de precisión en la clasificación por una ganancia en sparsity.

En el Capítulo 4, los resultados obtenidos en los Capítulos 2 y 3 se adaptan para construir árboles sparse para regresión. Se exploran propiedades teóricas de las soluciones. Los experimentos numéricos demuestran la escalabilidad de nuestro enfoque con respecto al tamaño de la muestra de entrenamiento, y se ilustra cómo se generan las explicaciones locales en las variables predictoras continuas. Además, mostramos cómo esta metodología puede reducir la discriminación de grupos sensibles a través de las denominadas restricciones de justicia.

El Capítulo 5 extiende la metodología de regresión del Capítulo 4 para considerar variables predictoras funcionales. De manera simultánea, la detección de un número reducido de intervalos que son críticos para la predicción es abordada. La sparsity en la proporción de dominio de las variables predictoras funcionales a usar se modela también a través de un término de regularización añadido a la función objetivo. De esta forma, se ilustra el equilibrio obtenido entre la precisión de predicción y la sparsity en este marco.

Por último, el Capítulo 6 cierra la tesis con conclusiones generales y líneas futuras de investigación.

Abstract

This PhD dissertation bridges the disciplines of Operations Research and Machine Learning by developing novel Mathematical Optimization formulations and numerical solution approaches to build classification and regression tree-based models.

Contrary to classic classification and regression trees, built in a greedy heuristic manner, formulating the design of the tree model as an optimization problem allows us to easily include, either as hard or soft constraints, desirable global structural properties. In this PhD dissertation, we illustrate this flexibility to model: sparsity, as a proxy for interpretability, by controlling the number of non-zero coefficients, the number of predictor variables and, in the case of functional ones, the proportion of the domain used for prediction; an important social criterion, the fairness of the model, which aims to avoid predictions that discriminate against race, or other sensitive features; and the cost-sensitivity for groups at risk, by ensuring an acceptable accuracy performance for them. Moreover, we provide in a natural way the impact that continuous predictor variables have on each individual prediction, thus enhancing the local explainability of tree models.

All the approaches proposed in this thesis are formulated through Continuous Optimization problems that are scalable with respect to the size of the training sample, are studied theoretically, are tested in real data sets and are competitive in terms of prediction accuracy against benchmarks. This, together with the good properties summarized above, is illustrated through the different chapters of this thesis.

This PhD dissertation is organized as follows. The state of the art in the field of (optimal) decision trees is fully discussed in Chapter 1, while the next four chapters state our methodology.

Chapter 2 introduces in detail the general framework that threads the chapters in this thesis: a randomized tree with oblique cuts. Particularly, we present our proposal to deal with classification problems, which naturally provides probabilistic output on class membership tailored to each individual, in contrast to the most popular existing approaches, where all individuals in the same leaf node are assigned the same probability. Preferences on classification rates in critical classes are successfully handled through cost-sensitive constraints.

Chapter 3 extends the methodology for classification in Chapter 2 to additionally handle sparsity. This is modeled by means of regularizations with polyhedral norms added to the

objective function. The sparsest tree case is theoretically studied. Our ability to easily trade in some of our classification accuracy for a gain in sparsity is shown.

In Chapter 4, the findings obtained in Chapters 2 and 3 are adapted to construct sparse trees for regression. Theoretical properties of the solutions are explored. The scalability of our approach with respect to the size of the training sample, as well as local explanations on the continuous predictor variables, are illustrated. Moreover, we show how this methodology can avoid the discrimination of sensitive groups through fairness constraints.

Chapter 5 extends the methodology for regression in Chapter 4 to consider functional predictor variables instead. Simultaneously, the detection of a reduced number of intervals that are critical for prediction is performed. The sparsity in the proportion of the domain of the functional predictor variables to be used is also modeled through a regularization term added to the objective function. The obtained trade off between accuracy and sparsity is illustrated.

Finally, Chapter 6 closes the thesis with general conclusions and future lines of research.

Contents

Chapter 1

Introduction

Extracting knowledge from data is a crucial task in Statistics and Machine Learning, and is at the core of many fields, such as Biomedicine [Jakaitiene et al., 2016; Leng and Müller, 2005; Pardalos et al., 2007], Business Analytics [Baesens et al., 2003; Martens et al., 2007; Van Vlasselaer et al., 2017], Computational Optimization [Alvarez et al., 2017; Khalil et al., 2016; Kruber et al., 2017; Lodi and Zarpellon, 2017], Criminal Justice [Ridgeway, 2013; Zeng et al., 2017], Cybersecurity [Martínez Torres et al., 2019], Health Care [Benítez-Peña et al., 2021; Bertsimas et al., 2016; Chaovalitwongse et al., 2008; Souillard-Mandar et al., 2016; Strzalkowska-Kominiak and Romo, 2021; Ustun and Rudin, 2016], Policy Making [Athey, 2018; Athey and Imbens, 2015; Athey et al., 2019; Kleinberg et al., 2018; Wager and Athey, 2018] and Regulatory Benchmarking [Benítez-Peña et al., 2020; Esteve et al., 2020; Lee and Cai, 2020]. Mathematical Optimization plays an important role in building such models and interpreting their output [Bertsimas and Shioda, 2007; Brooks, 2011; Carrizosa et al., 2018b, 2020, 2022a,b, 2017; Dash et al., 2018; Fang et al., 2013; Fountoulakis and Gondzio, 2016; Rudin and Ertekin, 2018], see Bottou et al. [2018]; Carrizosa et al. [2021a]; Carrizosa and Romero Morales [2013]; Duarte Silva [2017]; Gambella et al. [2021]; Goodfellow et al. [2016]; Liberti [2020]; Olafsson et al. [2008]; Palagi [2019]; Piccialli and Sciandrone [2018] for surveys.

The aim of this PhD dissertation is the development of novel Mathematical Optimization formulations for building classification and regression tree-based models, as well as numerical solution approaches to solve them. Our models, formulated through Continuous Optimization, enhance the flexibility of standard tree-based models. They are able to easily incorporate important desirable properties such as cost-sensitivity, explainability, fairness and sparsity. Moreover, they are able to deal with complex data, such as functional data, see Cuevas [2014]; Wang et al. [2016]; Goia and Vieu [2016] for recent reviews and a special issue on the topic, respectively.

The chapter is organized as follows. In Section 1.1, we introduce the role of Mathematical Optimization when building a classification and regression tree. In Section 1.2, we briefly go through the simplest (greedy heuristic) approaches to construct classification and regression trees, as well as extensions such as Random Forests, to understand how the one-shot optimization of the decisions across the whole tree is overcome. In Section 1.3, we review the Mixed Integer Linear Optimization paradigms, as well as our Continuous Optimization proposal, to build optimal decision trees, and how they compare against each other. Finally, Section 1.4 describes the main contributions of this thesis.

1.1 The role of Mathematical Optimization in Classification and Regression Trees

Classification and regression trees [Loh, 2014] are state-of-the-art methods based on recursive partitioning [Hastie et al., 2009]. They are conceptually simple, show excellent learning per-

formance, are computationally cheap and routines and packages to train them are available in popular languages such as Python and R, and are also appealing in terms of interpretability [Freitas, 2014; Goodman and Flaxman, 2017; Hu et al., 2019; Lin et al., 2020; Meinshausen, 2010] because of their rule-based nature. This makes them popular in many applications, including, for instance, a credit scoring exercise for granting a loan, described in what follows for illustration purposes. There, we have a data set of individuals characterized by demographic and financial predictor variables, among others, and, with this information, the model predicts whether customers will be *good* or *bad* payers. In Figure 1.1, we have a stylized credit scoring tree that will help us visualize some of the concepts reviewed below.

To construct a tree model, one has at hand a training sample defined by a set of individuals on which several predictor variables, either numerical or functional, as well as a response variable are measured. Note that w.l.g. we assume that categorical variables have been modeled through dummy ones, and thus replaced by a set of binary variables indicating the presence/absence of each category.

The main goal of a classification and regression tree is to predict, as accurately as possible, the response variable using the predictor variables. On the top of this primary goal, other important characteristics may need to be considered, such as, e.g., cost-sensitivity constraints to protect risk groups [Kao and Tang, 2014; Turney, 1995]; fairness of the method avoiding the discrimination of groups that share sensitive features such as gender and race [Aghaei et al., 2019; Iosifidis and Ntoutsi, 2019; Miron et al., 2020; Obermeyer et al., 2019; Romei and Ruggieri, 2014; Zafar et al., 2017]; and explainability [Ghorbani and Zou, 2020; Gunning and Aha, 2019; Holter et al., 2018; Miller, 2019] properties, e.g., sparsity of the tree model [Bertsimas and Digalakis, 2022; Cohen et al., 2007; Guyon and Elisseeff, 2003; Hastie et al., 2015; Weston et al., 2003] and local interpretability of the model [Lundberg et al., 2020; Lundberg and Lee, 2017; Molnar et al., 2020; Ribeiro et al., 2016], in order to ensure that the knowledge gained is actionable [Aouad et al., 2019; Bertsimas et al., 2019; Cui et al., 2015; Höppner et al., 2020; Mišić, 2020], according to the so-called right-to-explanation in algorithmic decision-making, imposed by the European Union since 2018 [European Commission, 2020; Goodman and Flaxman, 2017]. See Rudin et al. [2022] for a review on this topic.

A tree model consists of a tree decision structure and a prediction structure. The tree decision structure is defined by two elements, namely, the topology of the tree, i.e., the branch nodes and the leaf nodes, as well as the arcs between them, and the splitting rules applied at the branch nodes. The prediction structure is defined by the (statistical) prediction models attached to the leaf nodes. To illustrate these concepts, consider the topology of the tree model in Figure 1.1, which consists of two branch nodes, Node 1 and Node 2, and three leaf nodes, Node 3, Node 4 and Node 5. This is a binary tree, since each branch node has two children. The root node is where all individuals of the training sample start. These individuals move along the tree according to the queries asked at the branch nodes. In this way, and after partitioning the training sample successively, each individual ends up reaching exactly one leaf node. In terms

of splitting rules, the query asked in this example at the root node is whether predictor variable `age` is below 50, while at Node 2 we ask whether `salary` is below 30. The purpose of this splitting process is to ensure that individuals in the same leaf node follow the same pattern (i.e., they are from the same class or their response variable can be accurately predicted by a unique model, such as, for instance, a linear or a logistic model) and said pattern is expected to be also present for new individuals falling inside this leaf node. The prediction in a given leaf node is chosen fitting a model to the subsample fallen into it. In Figure 1.1, we can see that Node 4 predicts individuals as *good* payers, since this is the most frequent class (in bold font) in Node 4, while, following a similar argument, the other two leaf nodes predict as *bad* payers.

Once the tree model is built, the prediction of future data is done in a deterministic way. Given a new observation, starting from the root node, and applying the queries at the branch nodes, it will end up in a leaf node. The prediction is that associated with such leaf node. In our example, a new individual of `age` 43 and `salary` 28 would end up in Node 3, and therefore it would be predicted as a *bad* payer.

Mathematical Optimization is present within the three elements that define a tree model, namely, topology design of the tree, branching and prediction. First, we face the problem of designing the topology of the tree. This network design problem is often avoided, by, e.g., choosing a binary tree of a given depth. To make this decision more data dependent, a larger tree is built and pruned afterwards, collapsing existing leaf nodes into new ones containing more individuals. See, e.g., Sherali et al. [2009] for structural properties of the optimization problem associated with the pruning step. In this way, one obtains a more parsimonious tree, which is expected to perform better for future individuals. Second, we have to decide the splitting rules at each branch node. It is common to see trees implementing splitting rules that correspond to so-called orthogonal cuts, i.e., queries involving a single predictor variable, as in Figure 1.1. The choice of the predictor variable and the threshold value can be modeled with 0-1 decision variables. However, it is common to see enumerative procedures being applied independently to each of the branch nodes, thus solving the problem locally and not globally, as reviewed in Section 1.2. Although orthogonal cuts are popular (easy to build and to interpret), higher efficiency can be achieved with more sophisticated splitting rules that combine several predictor variables, such as, e.g., linear oblique cuts. See, e.g., Street [2005] for the optimization of oblique cuts. Third, and last, we need to decide how predictions are made at each leaf node. This boils down to solving an optimization problem for each one, the shape of which depends upon the nature of the response variable. For instance, in a regression tree, predictions can be made with a linear model obtained through an Ordinary Least Squares model. See, e.g., Demirović and Stuckey [2021] for the optimization of other criteria to measure the quality of prediction.

Because of the availability of more powerful hardware and the dramatic advances in optimization solvers over the last decades, there has been an increased interest by the Mathematical Optimization community to develop novel approaches to build classification and regression

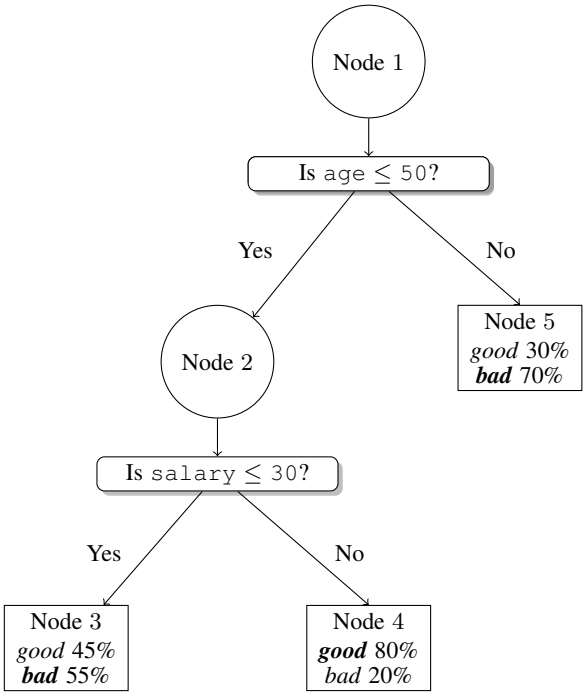


Figure 1.1: A tree model $\mathcal{T}$ to predict the *good* payers class vs the *bad* payers class, with $\tau_B = \{\text{Node } 1, \text{Node } 2\}$ and $\tau_L = \{\text{Node } 3, \text{Node } 4, \text{Node } 5\}$; orthogonal cuts $\texttt{age} \leq 50$ and $\texttt{salary} \leq 30$; and prediction *good* for Node 4 and *bad* for Node 3 and Node 5.

trees, as reviewed in Section 1.3.

1.2 Greedy Classification and Regression Trees

In this section, we review the basic steps of greedy heuristics for classification and regression trees as well as typical enhancements to reduce overfitting, or in other words, that these methods are too much focused on the training data and may fail to accurately predict incoming data. Throughout this section we discuss global optimization models that focus on the design of specific components of the tree model [Aglin et al., 2020; Barros et al., 2011; Bennett, 1992; Bennett and Blue, 1996; Fu et al., 2003; Grubinger et al., 2014; Hu et al., 2019; Lin et al., 2020; Nijssen and Fromont, 2010; Pangilinan and Janssens, 2011; Savickỳ et al., 2000].

Since constructing optimal binary classification and regression trees is known to be an NP-complete problem [Hyafil and Rivest, 1976], early research traditionally focused on the design of greedy heuristic procedures [Yang et al., 2017] that require a low computational effort to build tree models with just orthogonal cuts. These are recursive partitioning methods that build the tree model in a forward stepwise search implementing orthogonal cuts, yielding binary trees, e.g., CART [Breiman et al., 1984] and QUEST [Loh and Shih, 1997], or nonbinary trees, a.k.a. multi-way trees [Kim and Loh, 2001], e.g., CHAID [Kass, 1980] and C4.5 [Quinlan,

1993]. Figure 1.2 depicts the tree model for `carevaluations`, a real-world data set from the UCI Machine Learning repository [Blake and Merz, 1998], with 1728 car evaluations divided into 4 classes. This is a data set with 15 predictor variables, used to predict whether the car evaluation is *unacceptable*, *acceptable*, *good* or *very good*.

In these greedy heuristic approaches, a criterion is needed to guide the branching at each branch node. In our credit scoring example, at each branch node, one aims to leave (most of) the *good* payers at one branch and (most of) the *bad* payers at the other one. This has been done by optimizing some measure of the purity of a node with respect to the class split in a classification task, e.g., Gini index or entropy, or its homogeneity with respect to the response variable in a regression task, such as, e.g., mean squared error or mean absolute error. Purity, although popular, only measures classification accuracy in an indirect way, and therefore may not yield good generalization results [Fayyad and Irani, 1992], and other criteria, such as Maximum Likelihood [Su et al., 2004], have been proposed. At any branch node, one searches for the splitting rule that yields the larger gain in purity or homogeneity of the children versus the parent. For orthogonal cuts, this implies examining the gain for all the predictor variables and all possible values of threshold, as many as individuals in the parent node. Although in principle there may be an infinite amount of values for this threshold, just taking the midpoints between consecutive observed values of the predictor variable in the training sample suffices. We refer the reader to Liu et al. [2002] for a comprehensive review on enhancing classification and regression tree methods through the discretization of continuous predictor variables [Dougherty et al., 1995]. Functional predictor variables can follow the same splitting strategy proposed for numerical ones by losing their functional nature once discretized and converted to vectors. To overcome this issue, splitting rules tailored to functional predictor variables based, e.g., on the search of representative curves [Balakrishnan and Madigan, 2006] or on weighted integral features from the functional predictor variables, such as the mean or the variance, [Belli and Vantini, 2021], have been proposed. The process of partitioning finishes when a stopping criterion is satisfied, for instance, when the requirement on the minimum number of individuals at leaf nodes would be violated. Then, a prediction is chosen in each of the leaf nodes. Commonly, for classification, a leaf node is labeled with the most frequent class in the set of individuals that have fallen into the node, while for regression, the prediction equals to the average of the response variable on those individuals, which is the prediction given by a linear model with just an intercept and no predictor variables.

Trees built in this way may still overfit and therefore a post-pruning step is performed to remove any unnecessary splits. Pruning is usually performed in a greedy fashion in which leaf nodes are sequentially removed. While in the forward phase a purity criterion was considered, now a criterion combining accuracy and tree complexity is used. The removal of leaf nodes continues while the value of the criterion improves. To enhance their performance, the greedy procedures were extended in different directions, and we will elaborate on two important ones below.

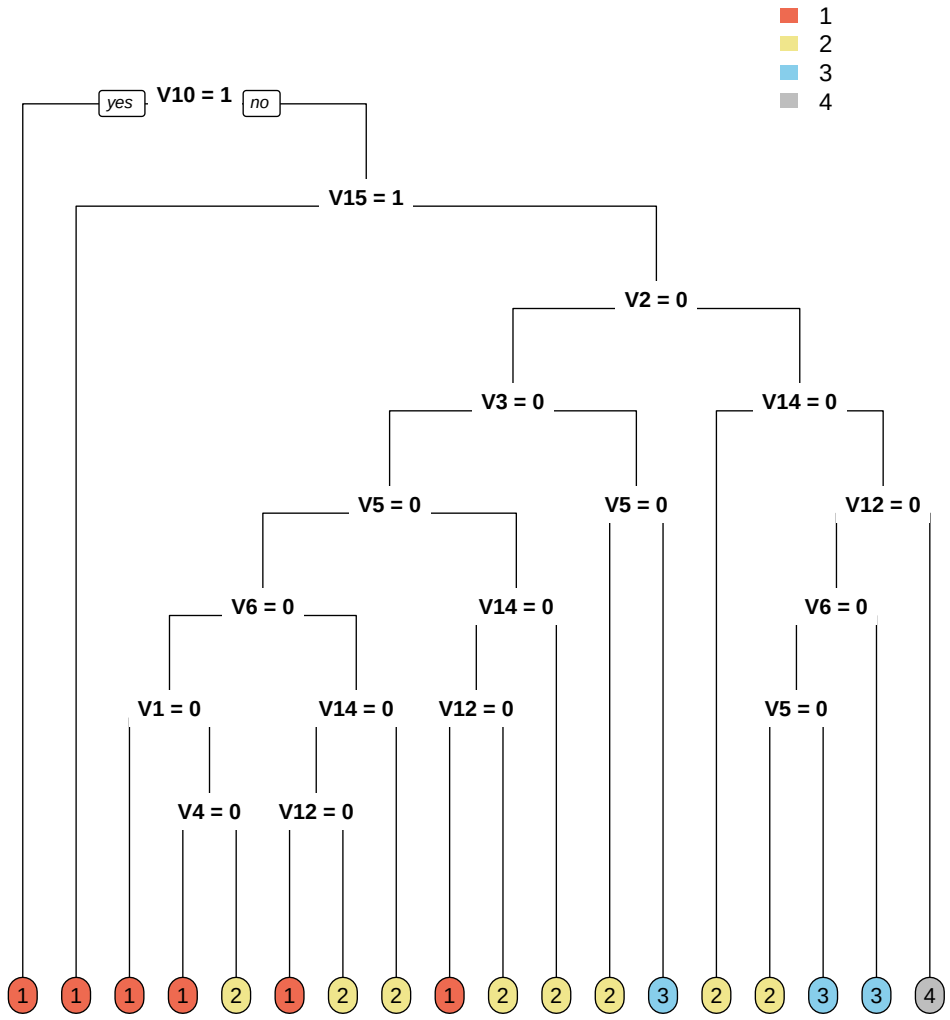


Figure 1.2: Illustration of CART for `carevaluations` obtained with the R package `rpart` Therneau et al. [2015]. There are 16 leaf nodes, predicting one of the four classes, namely *unacceptable* (1), *acceptable* (2), *good* (3) or *very good* (4). The classification accuracy provided by this model is 88.1%, while 71.3% of the predictor variables are used across the tree.

The first enhancement relates to extending orthogonal splits to oblique, a.k.a. multivariate, splits, with implementations such as OC1 [Murthy et al., 1994], oblique.tree [Truong, 2009] and HHCART [Wickramarachchi et al., 2016]. Trees implementing oblique cuts are more versatile and tend to generate smaller trees with better performance [Brodley and Utgoff, 1995; Li et al., 2003]. This improvement in accuracy comes with increasing computational times, since the enumerative procedure no longer applies and instead some form of optimization problem has to be solved at each branch node. In addition, model interpretability may also be harmed. There have been some proposals to build oblique cuts using a baseline classification method at each branch node, such as Support Vector Machines [Orsenigo and Vercellis, 2003] or Logistic Regression [Truong, 2009], such that the predictions obtained in this way split the parent node into children. Nevertheless, tackling the optimization of oblique cuts is already found in the seminal papers of Bennett [Bennett, 1992; Bennett and Blue, 1996]. For binary classification, it is adjusted to the tree context the use of Linear Programming (LP) to build separating hyperplanes [Bennett and Mangasarian, 1992]. In Bennett [1992], the hyperplane that minimizes the average distance from the misclassified individuals to the hyperplane is modeled as an LP problem, while in Bennett and Blue [1996], for a fixed topology of the tree and fixed predictions at the leaf nodes, the problem of finding the optimal oblique cuts for all branch nodes is written as a set of disjunctive linear inequalities yielding a nonlinear problem. Since these approaches apply only to two-class problems, in Street [2005], multi-class problems are addressed. In Norouzi et al. [2015], and given the challenge of optimizing the empirical loss of the tree model, a convex-concave upper bound is optimized instead, using Stochastic Gradient Descent.

The second enhancement relates to building models that combine the outputs given by a collection of trees, as opposed to a single one, using bagging or boosting techniques, such as Random Forests [Biau and Scornet, 2016; Breiman, 2001; Cousins and Riondato, 2019; Fawagreh et al., 2014; Genuer et al., 2017; Möller et al., 2016; Pospisil and Lee, 2019; Rahman et al., 2019] or Gradient Boosting Machines [Friedman, 2001, 2002; Pande et al., 2017], respectively. Random Forests (RFs) bag (unpruned) orthogonal trees, and more recently oblique ones [Katuwal et al., 2020; Menze et al., 2011]. The trees in the RFs are built on bootstrapped samples of individuals, where the variable selection at each branch node is performed using a random subset of predictor variables. Hence, the trees built differ because different samples of individuals and predictor variables are used. Once the trees are built following the greedy approach described above, RF predicts by combining the predictions of the single trees.

There are other techniques to combine tree models, such as boosting, where weak learners in the form of trees of small depth are combined through weights. These weights need to be optimized [Demiriz et al., 2002; Friedman, 2001; Pfetsch and Pokutta, 2020], and techniques such as column generation are used for this. RFs, as well as other methods combining tree models, give, in general, better accuracies than single greedy trees [Fernández-Delgado et al., 2014]. However, this is at the expense of losing interpretability and increasing running times.

Indeed, these models have a highly complex decision function, being thus less appealing to novel users. The way this lack of interpretability is often addressed is by giving a measure of variable importance, which are often based on permutations of the sample values [Altmann et al., 2010; Louppe et al., 2013; Strobl et al., 2008] or on Game Theory concepts from cooperative games, such as the Shapley value [Casalicchio et al., 2019; Molnar et al., 2018]. Recently, there have also been some contributions to enhance model interpretability by replacing the complex model with a simpler surrogate, say a tree model, such that the output of both models is as close as possible. This approach is suggested in Vidal and Schiffer [2020], where the closest tree is extracted using dynamic programming. Alternatively, one can extract a collection of rules, and techniques such as column generation may be used as in Akyüz and Birbil [2021]; Birbil et al. [2020] or heuristics as in Bénard et al. [2019, 2021].

To end this section, we note that classic classification and regression trees, as well as the extensions mentioned above, cannot easily include desirable global structural properties, such as model sparsity and cost-sensitivity, due to their greedy nature. Nevertheless, some attempts have been made to address this shortcoming. To enhance model interpretability, one wishes to perform feature selection to control the number of predictor variables used across the tree [Ruggieri, 2019]. The regularization framework in Deng and Runger [2012] adds to the criterion optimized in each branch node a penalty term for predictor variables that have not appeared yet in the tree, so that the process is reluctant to use too many predictor variables, yielding a sparse tree model. This approach is refined in Deng and Runger [2013], by also including the importance scores of the predictor variables [Louppe et al., 2013; Strobl et al., 2008], obtained in a preprocessing step running a preliminary RF. In the next section, we model sparsity explicitly, which can then be optimized, as we achieve with the learning performance of the tree model. Similarly, the control on the performance of the tree model in critical/risk groups is done through cost parameters, such as penalizing with a higher cost the errors in the critical groups, as opposed to modeling the corresponding constraints explicitly as we do.

1.3 Optimal Classification and Regression Trees

In recent times, many papers on building optimal (in some sense) classification and regression trees have appeared. In this section, we review the literature on Mixed Integer Linear Optimization approaches, and move on to the Continuous Optimization paradigm, which is the focus of this thesis. The reader is referred to, e.g., Verhaeghe et al. [2020] for a constraint programming paradigm, a SAT (Boolean satisfiability) one in Narodytska et al. [2018]; Yu et al. [2020], and a dynamic programming one in Demirović et al. [2022].

1.3.1 Mixed-Integer Linear Optimization

In this section, we review Mixed-Integer Linear Optimization (MILO) approaches to build Optimal Classification and Regression Trees. The key issue in this paradigm is that one controls

the path each individual takes and thus calls for the modeling of (many) binary decision variables. We start with the approach in Bertsimas and Dunn [2017]; Dunn [2018], and continue by reviewing other relevant literature, which involves different decision variables and/or more sophisticated solution approaches.

In Bertsimas and Dunn [2017], the aim is to build a deterministic optimal binary tree of a given depth guided by two objectives, namely, the misclassification error and the complexity of the tree, where the latter is measured as the summation across all branch nodes of the number of predictor variables used in the splitting rules. The MILO formulation in Bertsimas and Dunn [2017] requires many decision variables, and a great part of them are binary. Also, the size of some of the binary decision variables linearly depends on the size of the training sample. Therefore, as noted in Dunn [2018], this approach is only feasible for moderate training sample sizes. This formulation has a nonlinear objective function, but one can define additional variables and additional big-M constraints to linearize it. This formulation can be given to any MILO solver. The computational experiments in Bertsimas and Dunn [2017] illustrate that good accuracies can be achieved with small depths, but at a considerable computational cost for small and medium problem instances. To reduce this computational burden, a local search approach is proposed in Dunn [2018], where the MILO formulation is solved for the subproblems associated with branch nodes, thus yielding smaller formulations that are solved repeatedly. With this local search procedure, it is possible to deal with deeper trees, up to depth 10, more efficiently. However, it is harder to directly control, for instance, the number of predictor variables to be used across the tree, a crucial issue if, on top of having a procedure yielding high accuracies, identification of the relevant predictor variables is sought. Moreover, we cannot perform a proper sensitivity analysis to explain how small perturbations on a given feature affect the prediction. This means that it is not easy to identify the relevant variables for a given individual, while this is obtained as a byproduct for randomized trees, as seen in Section 1.3.2.

There are other approaches in the literature within the MILO paradigm. In Aghaei et al. [2020], a flow-based MILO formulation is proposed for binary predictor variables. A sink node is added to the tree, yielding a directed acyclic graph [Ahuja et al., 1993]. Only individuals ending up in the sink are correctly classified, while flow conservation constraints are imposed at the other nodes of the tree. In Firat et al. [2019], an alternative formulation is proposed with new decision variables associated with the paths from the root node to the leaf nodes and their splits, which is solved with a column generation-based heuristic. In Günlük et al. [2021], a MILO formulation for combinatorial splitting rules for categorical variables is proposed, i.e., rules defined by a subset of categories that move individuals to the left child node if the rule is satisfied and to the right child node, otherwise, yielding a binary representation [Carrizosa et al., 2021b].

With the MILO approach, we face the curse of dimensionality since the number of binary decision variables grows linearly with the size of the training sample. Recent attempts to

address this can be found in the literature. An alternative formulation is proposed in Verwer and Zhang [2017]; Verwer et al. [2017, 2019] with a more compact feasible region that aggregates some of the constraints described above. In Zhu et al. [2020] a subset of the training sample is selected in a preprocessing step using an LP problem, while in Zantedeschi et al. [2020] a continuous relaxation is developed.

1.3.2 Continuous Optimization

In this section, we describe the Nonlinear Continuous Optimization approach proposed in this thesis to build Optimal Randomized Classification and Regression Trees, where functional predictor variables are also addressed. We then discuss how it compares to the MILO formulations in the previous section.

In Optimal Randomized Classification and Regression Trees, the splitting rule at each branch node is probabilistic [Irsoy et al., 2012; Suárez and Lutsko, 1999; Yang et al., 2018], i.e., individuals move with a certain probability to the left child and with the complementary probability to the right one. This type of rule is modeled evaluating the smooth cumulative density function (CDF) of a univariate continuous random variable at splitting rules. See in Figure 1.3 the probabilistic splitting rules at Nodes 1 and 2, using the CDF of a logistic random variable. With a probabilistic splitting rule, each individual moves along all paths in the tree, by defining a probability distribution across the leaf nodes. With the probabilities associated to the individuals and the predictions at the leaf nodes, one can evaluate the total expected error of the randomized tree model. The goal of Optimal Randomized Classification and Regression Trees is to minimize the expected error as well as maximize different kinds of sparsity of the tree. These sparsity terms are modeled with LASSO terms and controlled with their corresponding parameters. Figure 1.4 plots the Optimal Randomized Classification Tree for `carevaluations`.

Once the tree model is built, the prediction of future data is done in a probabilistic way. The predictions made at leaf nodes are weighted according to the probability distribution defined across them, and added up. Note that the prediction function of randomized trees is smooth in the continuous predictor variables, since the CDF is assumed to be a smooth function.

Apart from being continuous, this formulation requires fewer decision variables than the MILO paradigm. The number of nodes in the tree, and thus the number of decision variables in both paradigms, grows exponentially with the depth of the tree. Hence, solving this problem may be time-demanding for large or even moderate depths. Fortunately, as in Section 1.3.1, the computational experiments in Blanquero et al. [2020b, 2021a, 2022a,b] illustrate that good accuracies can be achieved with small depths, namely, up to 4, with a low computational effort. There are other important remarks on this problem that are noteworthy. First, the feasible region speaks favorably towards the scalability with respect to the size of the training sample. Indeed, when the number of individuals grows, the feasible region remains of the same size, since there are no decision variables directly relating to the individuals. Hence, although the evaluation of

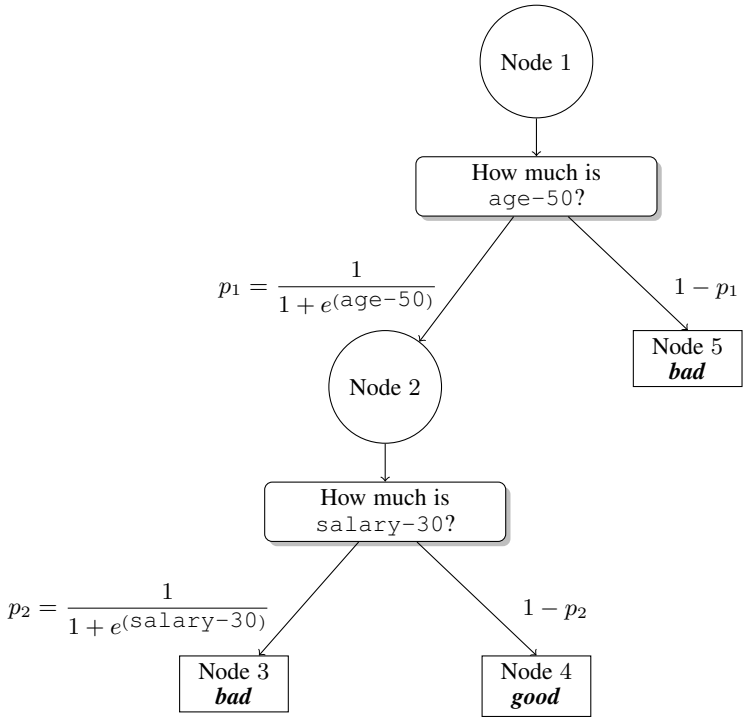


Figure 1.3: A randomized tree model $\mathcal{T}$ to predict the *good* payers class vs the *bad* payers class, using the CDF of a logistic random variable.

the objective function becomes more time demanding with larger sizes, the dimensionality of the problem to be solved remains the same. Second, there are two regularization terms that can help with feature selection, i.e., identify a subset of predictor variables with a good tradeoff between accuracy and sparsity. Third, we can perform with standard techniques a full sensitivity analysis to study the impact that predictor variables have on the prediction for each individual. Recall that the prediction function is smooth in the continuous predictor variables. Therefore, we have that small changes in the continuous predictor variables in a given individual lead to small changes in prediction since the prediction function can be approximated by its first order Taylor expansion.

1.4 Contributions of this thesis

This PhD dissertation is devoted to enhancing Classification and Regression Tree-Based Models by means of Mathematical Optimization. The reader is referred to the paper Carrizosa et al. [2021a] for a comprehensive review on this area. In this section, we briefly introduce and motivate the problems addressed.

Chapter 2, based on the paper Blanquero et al. [2021a], introduces ORCT, a novel continuous optimization approach to build classification trees with oblique cuts that scales well with the size of the training sample. ORCT naturally provides probabilistic output on class

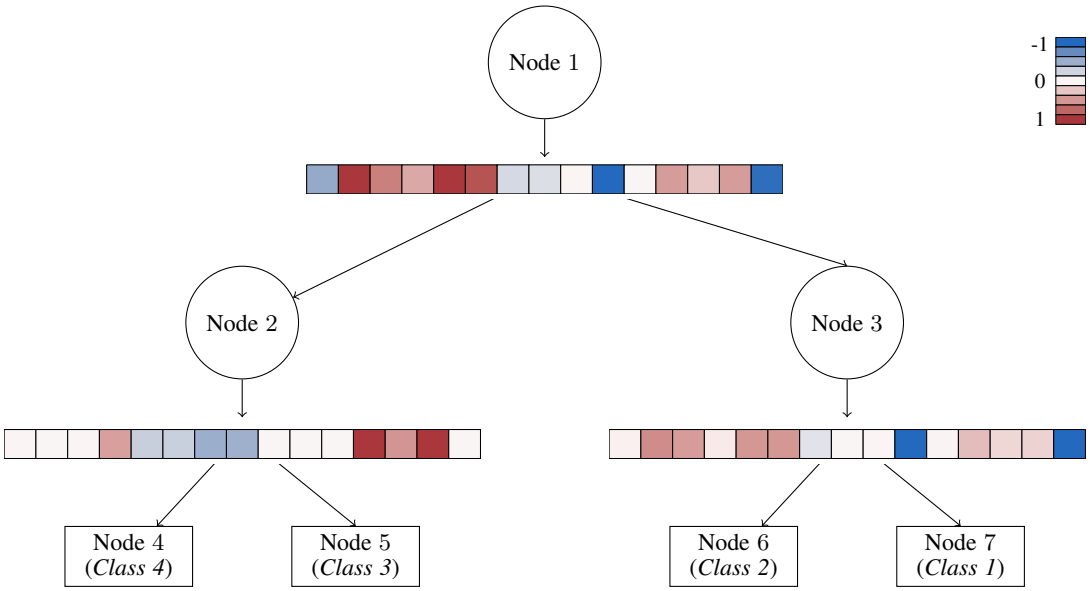


Figure 1.4: Illustration of Optimal Randomized Classification Tree for `carevaluations`. The classification accuracy of this model is 92.7%, while 100% of the predictor variables are used across the tree as well as in each of the three branch nodes. The magnitude of the coefficients of the splitting rule in each branch node is visualized with a heatmap. The heatmap transitions from blue for negative coefficients, to red for positive ones, while white is chosen for values close to 0.

membership tailored to each individual, in contrast to the most popular existing approaches, where all individuals in the same leaf node are assigned the same probability. The computational experience reported illustrates the outperformance of our approach in terms of accuracy against CART and OC1 as well as recent proposals based on integer programming as those in Bertsimas and Dunn [2017]. Moreover, we are comparable to *oblique.tree* and manage to get close to improved methods like RFs and the local-search heuristic in Dunn [2018]. Preferences on classification rates in critical classes are successfully handled through cost-sensitive constraints, unlike heuristic approaches like CART, OC1, *oblique.tree* or RF, which can not address this issue explicitly.

Chapter 3, based on the paper Blanquero et al. [2020b], introduces S-ORCT, a continuous optimization approach that extends ORCT to additionally handle sparsity, with the aim of using fewer predictor variables in the cuts as well as along the whole tree, and thus enhancing interpretability. Both types of sparsity, namely local and global, are modeled by means of regularizations with polyhedral norms. Theoretical results on the range of the sparsity parameters are shown. The computational experience reported supports the usefulness of our methodology. In all our data sets, local and global sparsity can be improved without harming classification accuracy. Unlike greedy approaches like CART, our ability to easily trade in some of our classification accuracy for a gain in global sparsity is shown.

Chapter 4, based on the paper Blanquero et al. [2022b], introduces S-ORRT, a continuous optimization approach that extends S-ORCT to consider regression trees instead. Similar theoretical properties to those for classification are explored in this framework. Apart from sparsity, and unlike CART and RF, S-ORRT can easily accommodate other important desirable properties for the regression task, such as cost-sensitivity and fairness. Thanks to the smoothness of the predictions, local explanations on the continuous predictor variables are derived. The computational experience reported shows the outperformance of S-ORRT in terms of prediction accuracy against standard benchmark regression methods such as CART, OLS and LASSO. Moreover, the scalability of our approach with respect to the size of the training sample is illustrated.

Chapter 5, based on the work Blanquero et al. [2022a], introduces S-ORRT-FD, a continuous optimization approach that extends S-ORRT to consider functional predictor variables instead. Whilst fitting S-ORRT-FD, the detection of a reduced number of intervals that are critical for prediction is performed, yielding a more interpretable output. An additional kind of sparsity comes into play in this framework, which is defined as the proportion of the domain of the functional predictor variables to be used, namely, length sparsity, also modeled through a regularization term. The computational experience reported shows that S-ORRT-FD is competitive in terms of prediction accuracy to benchmark methods, including RF. Moreover, S-ORRT-FD illustrates its ability to easily trade in some of our prediction accuracy for a gain in length sparsity.

Finally, some general conclusions and possible lines of future research are provided in

Chapter 6.

Chapter 2

Optimal Randomized Classification Trees

In this chapter, a novel continuous-based approach for building optimal classification trees with oblique cuts is provided, called hereafter an Optimal Randomized Classification Tree (ORCT). The randomization of our approach as well as the inclusion of oblique cuts allow the removal of the integer decision variables present in other recent proposals in the literature. Moreover, the feasible region in our formulation is independent of the size of the training sample, which makes our approach scalable. An ORCT naturally provides probabilistic output on class membership tailored to each individual, in contrast to existing approaches, where all individuals in the same leaf node are assigned the same probability. Preferences on classification rates in critical classes are successfully handled, unlike heuristic approaches such as CART, OC1, *oblique.tree* or RF, which can not address this issue explicitly. Our numerical results illustrate the outperformance of our approach in terms of accuracy against CART and OC1 as well as recent proposals based on integer programming as those in Bertsimas and Dunn [2017]. Moreover, we are comparable to *oblique.tree* and manage to get close to improved methods like RFs and the local-search heuristic in Dunn [2018].

The chapter is organized as follows. In Section 2.1, we introduce the ORCT and its formulation, as well as a variant that includes the possibility of controlling the classification performance in critical classes. In Section 2.2, we show some theoretical properties that belong to ORCT. In Section 2.3, computational experiments with ORCTs are reported. The results obtained are compared with CART, OC1, *oblique.tree*, the integer programming-based approach in Bertsimas and Dunn [2017] and its related local-search heuristic in Dunn [2018], and RFs. Finally, conclusions and possible extensions are provided in Section 2.4.

2.1 Optimal Randomized Classification Trees

2.1.1 The basic idea

Suppose we are given a training sample of N individuals, $I = \{(\boldsymbol{x}_i, y_i)\}_{1 \leq i \leq N}$, on which p numerical predictor variables are measured, $\boldsymbol{x}_i \in \mathbb{R}^p$, and one class label is associated with each one, $y_i \in \{1, \ldots, K\}$. Let $I_k \subset I$ denote the set of individuals in I in class k, and $|I_k|$ its cardinality. For simplicity, numerical predictor variables are considered. Without loss of generality, we assume that $\boldsymbol{x}_i \in [0, 1]^p$.

Classic decision tree methods consist of sequentially, and greedily, partitioning the predictor space $[0, 1]^p$ into disjoint sets or nodes by imposing certain conditions on the predictor variables. The usual splitting criterion is to take the split that makes descendant nodes purer, i.e., nodes with observations more and more homogenous in terms of class membership. The process of partitioning finishes when a stopping criterion is satisfied. Then, leaf nodes are labeled with a class label, $1, \ldots, K$. Commonly, a leaf node is labeled with the most frequent class in the individuals that have fallen into the node. Once the tree is built, the prediction of future unlabeled data is done in a deterministic way. Given a new observation $\boldsymbol{x}$ starting from the root node, it will end up in a leaf node depending on the values the predictor variables take,

and its predicted class will be the class label of that leaf node. Alternatively, the prediction of future unlabeled data can be done in a probabilistic way, using the relative frequencies of belonging to each class of the corresponding leaf node.

As introduced in Chapter 1, the approach proposed here is different: prediction is randomized. At each node of an Optimal Randomized Classification Tree (ORCT), a random variable will be generated to indicate by which branch one has to continue. Since we are building binary trees, the Bernoulli distribution is appropriate, whose probability of success will be determined by the value of a cumulative density function (CDF), evaluated over the vector of predictor variables. In this way, each leaf node will not contain a subset of individuals but all the individuals in the training sample, for which the probability of falling into such leaf node is known. Finally, the class label of each leaf node will be a decision variable, which will be found by minimizing the expected misclassification cost over the whole training sample.

The distinctive element in our approach is the fact that the yes/no rule in decision tree methods is replaced by a soft rule, induced by a continuous CDF. In this way, a smoother and therefore more stable rule is obtained. Indeed, suppose that the first cut of a classic classification tree forces individuals with $X \leq b$ to go down the tree by the left branch. Then, for an incoming individual with $X = b + \varepsilon$, $\varepsilon > 0$ sufficiently small, classic decision tree methods would give them probability of going down the branch $X > b$ equal to 1 and 0, otherwise, as in the orange line in Figure 2.1. To avoid the discontinuities present in tree-based estimators like CART and RF, we propose to smooth the probabilities of going left or right in a neighbourhood of b through a CDF, see the green line in Figure 2.1.

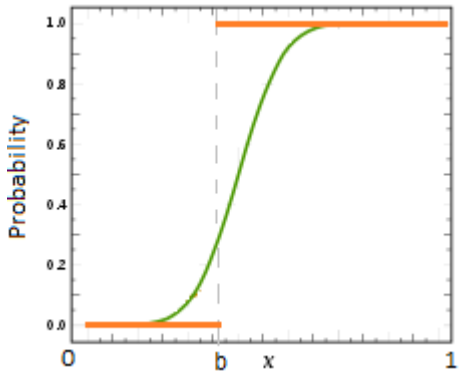


Figure 2.1: The probability of an individual going down by the right branch is depicted for both types of trees: the classic approach (orange line) and the proposed ORCT (green line).

2.1.2 The model

After having introduced how ORCTs work, we will next formulate the problem.

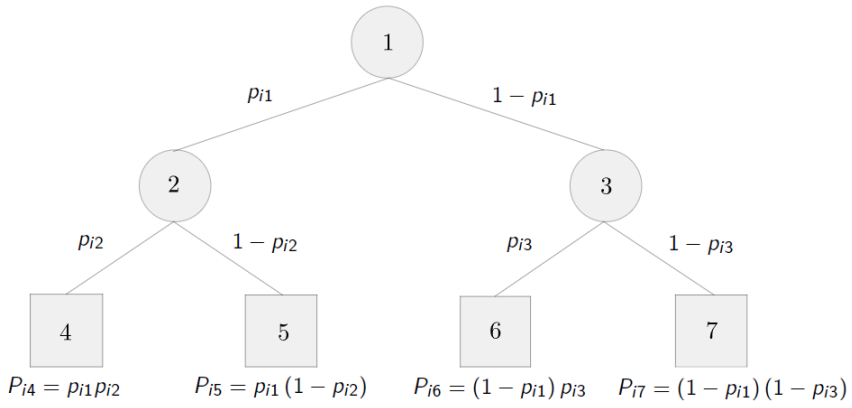


Figure 2.2: Optimal Randomized Classification Tree for depth $D = 2$.

Our approach starts from the maximal binary tree of depth D, i.e., a binary tree in which each branch node has two children and terminal nodes all have the same depth, namely, D. For instance, Figure 2.2 shows the maximal binary tree of depth $D = 2$. Given D, the total number of nodes is known in advance, $T = 2^{(D+1)} - 1$. The sets of branch and leaf nodes are known and numbered as follows:

Branch nodes: nodes $t \in \tau_B = \{1, \ldots, \lfloor T/2 \rfloor\}$.

Leaf nodes: nodes $t \in \tau_L = \{\lfloor T/2 \rfloor + 1, \ldots, T\}$.

Oblique splits are modeled through linear combinations of the predictor variables. To do that, we need to define, for each $j = 1, \ldots, p$ and each $t \in \tau_B$, the decision variables $a_{jt} \in [-1, 1]$ to indicate the value of the coefficient of predictor variable j in the oblique cut over branch node $t \in \tau_B$. The $p \times |\tau_B|$ matrix of these coefficients will be denoted by $\boldsymbol{a} = (a_{jt})_{j=1,\ldots,p, t \in \tau_B}$, and the expressions $\boldsymbol{a}_{j\cdot}$ and $\boldsymbol{a}_{\cdot t}$ will stand for the j-th row and the t-th column of $\boldsymbol{a}$, respectively. The intercepts of the linear combinations correspond to decision variables $\mu_t \in [-1, 1]$, seen as the location parameter at every branch node $t \in \tau_B$. Let $\boldsymbol{\mu}$ be the vector that comprises every μ_t, i.e., $\boldsymbol{\mu} = (\mu_t)_{t \in \tau_B}$.

Now, the smooth CDF of a univariate continuous random variable $F(\cdot)$ centered at zero is assumed. Then, for each individual $i = 1, \ldots, N$ at each branch node $t \in \tau_B$, the parameter of their corresponding Bernoulli distribution is obtained as follows:

$$p_{it}(\boldsymbol{a}_{\cdot t}, \mu_t) = F\left(\frac{1}{p}\sum_{j=1}^{p} a_{jt} x_{ij} - \mu_t\right), \; i = 1, \ldots, N, \; t \in \tau_B. \tag{2.1}$$

Note that this probability is a continuous function in the predictor variables $\boldsymbol{x}_i$, since the CDF F is a continuous function.

The value $p_{it}(\boldsymbol{a}_{\cdot t}, \mu_t)$ will be used in the corresponding left branch and $1 - p_{it}(\boldsymbol{a}_{\cdot t}, \mu_t)$ in the right one, as seen in Figure 2.2. We denote as $N_L(t)$ the set of ancestor nodes of node t whose left branch takes part in the path from the root node to node t. Respectively, $N_R(t)$ is

the set of ancestor nodes of node t whose right branch takes part in the path from the root node to t. If $N(t)$ denotes the set of ancestors of node t, we have that $N(t) = N_L(t) \cup N_R(t)$. For leaf node $t = 5$ in Figure 2.2: $N_L(5) = \{1\}$, $N_R(5) = \{2\}$ and $N(5) = \{1, 2\}$.

Once these sets are defined, we can obtain the probability of an individual falling into a given leaf node:

$$P_{it}\left(\boldsymbol{a}, \boldsymbol{\mu}\right) \equiv \mathbb{P}\left(\boldsymbol{x}_i \in t\right) = \prod_{t_l \in N_L(t)} p_{it_l}\left(\boldsymbol{a}_{\cdot t_l}, \mu_{t_l}\right) \prod_{t_r \in N_R(t)} \left(1 - p_{it_r}\left(\boldsymbol{a}_{\cdot t_r}, \mu_{t_r}\right)\right), \ i = 1, \ldots, N, \ t \in \tau_L. \tag{2.2}$$

As a consequence of (2.1), this probability is a continuous function in the predictor variables $\boldsymbol{x}_i$.

Now, it is necessary to define, for each leaf node $t \in \tau_L$, the binary decision variables $\boldsymbol{C} = (C_{kt})_{k=1,\ldots,K, t \in \tau_L}$ that model the class label assigned to each of them, where

$$C_{kt} = \begin{cases} 1, & \text{if node } t \text{ is labeled with class } k \\ 0, & \text{otherwise} \end{cases}, k = 1, \ldots, K, \ t \in \tau_L.$$

We must add the following set of constraints for making a single class prediction at each leaf node:

$$\sum_{k=1}^{K} C_{kt} = 1, \ t \in \tau_L.$$

As a natural strengthening, we force each class $k = 1, \ldots, K$ to be identified by, at least, one terminal node, by adding the set of constraints below:

$$\sum_{t \in \tau_L} C_{kt} \geq 1, \ k = 1, \ldots, K, \tag{2.3}$$

where it is implicitly assumed that $K \leq 2^D$ so that the previous constraints make sense. This set of constraints prevent the observations belonging to the minority classes from being fully misclassified. Nevertheless, they could be easily removed when desired.

For fixed $\boldsymbol{a}, \boldsymbol{\mu}, \boldsymbol{C}$, the probability of individual i being assigned to class k is equal to

$$\sum_{t \in \tau_L} P_{it}\left(\boldsymbol{a}, \boldsymbol{\mu}\right) C_{kt}. \tag{2.4}$$

Using the continuity of $P_{it}\left(\boldsymbol{a}, \boldsymbol{\mu}\right)$ in the predictor variables $\boldsymbol{x}_i$, we have that small changes in $\boldsymbol{x}_i$ lead to small changes in the values of the probabilities of class membership in (2.4). We illustrate this amenable property of our approach using the balanced two-class simulated data set in Figure 2.3. The data set consists of $N = 400$ individuals equally split into the two classes, characterized by $p = 2$ predictor variables. The predictor variables for individuals labeled as class k, $k = 1, 2$, have been generated following a bivariate normal distribution, $\mathcal{N}\left(\boldsymbol{\eta}_k, \boldsymbol{\Sigma}_k\right)$. We have chosen $\boldsymbol{\eta}_1 = (0.00, 1.25)^\top$, $\boldsymbol{\eta}_2 = (1.00, -0.25)^\top$ and $\boldsymbol{\Sigma}_1 = \boldsymbol{\Sigma}_2$ the

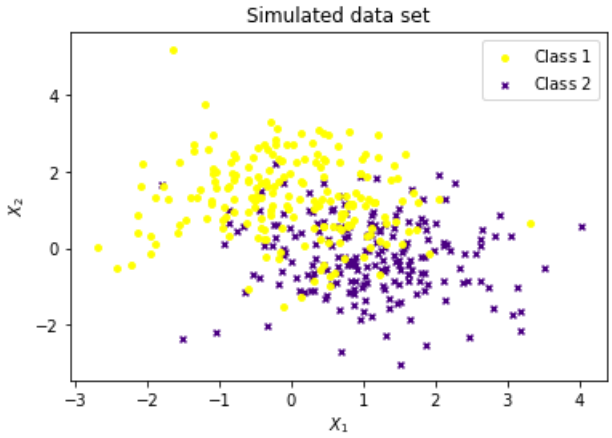


Figure 2.3: Simulated data set with $p = 2$, $K = 2$ and $N = 400$ to compare the probabilities of class membership derived from ORCT with those derived from CART (deterministic and probabilistic) and RF.

identity matrix of size 2. In Figure 2.4, we compare the probabilistic output of ORCT with the one derived from the two versions of CART described in Subsection 2.1.1 (a deterministic as well as a probabilistic one) and RF. The CART classifier is built with the `rpart` R package [Therneau et al., 2015] and RF with the `randomForest` R package [Liaw and Wiener, 2002] both with the default tuning parameters, while the ORCT classifier outlined below is built with depth $D = 1$. For each classifier, we derive the probability of belonging to class $k = 1$ and use a heatmap plot to visualize it. Clearly, our ORCT approach is able to produce smoother class membership probabilities. We illustrate these probabilities at four different points, namely, $A_1 = (-1.0, 2.0)$, $A_2 = (1.1, 1.0)$, $A_3 = (-2.1, -1.1)$ and $A_4 = (3.0, 2.3)$. On the yellow zone, the probability of belonging to class $k = 1$ for A_1 is equal 0.9998. If one is placed on the oblique cut, almost no-discriminatory probabilities are obtained. Indeed, the probability of belonging to class $k = 1$ for A_2 is equal to 0.5077. Above the cut, the probability of belonging to class $k = 1$ increases smoothly. See A_3 on the green zone, for which the probability of belonging to class $k = 1$ is equal to 0.8701. Likewise, below the cut, the probability of belonging to class $k = 1$ decreases instead. For A_4, this probability is equal to 0.2438.

Once the probabilities in (2.4) have been defined, we can now model the objective function of our model. As said before, the objective is to minimize the expected misclassification cost over the sample, so we need to introduce a misclassification cost for classifying an individual i, whose class is y_i, in class k:

$$W_{y_i k} \geq 0, \;\; k = 1, \ldots, K. \tag{2.5}$$

We define $W_{y_i k} = 0$ if $y_i = k, \;\; k = 1, \ldots, K$.

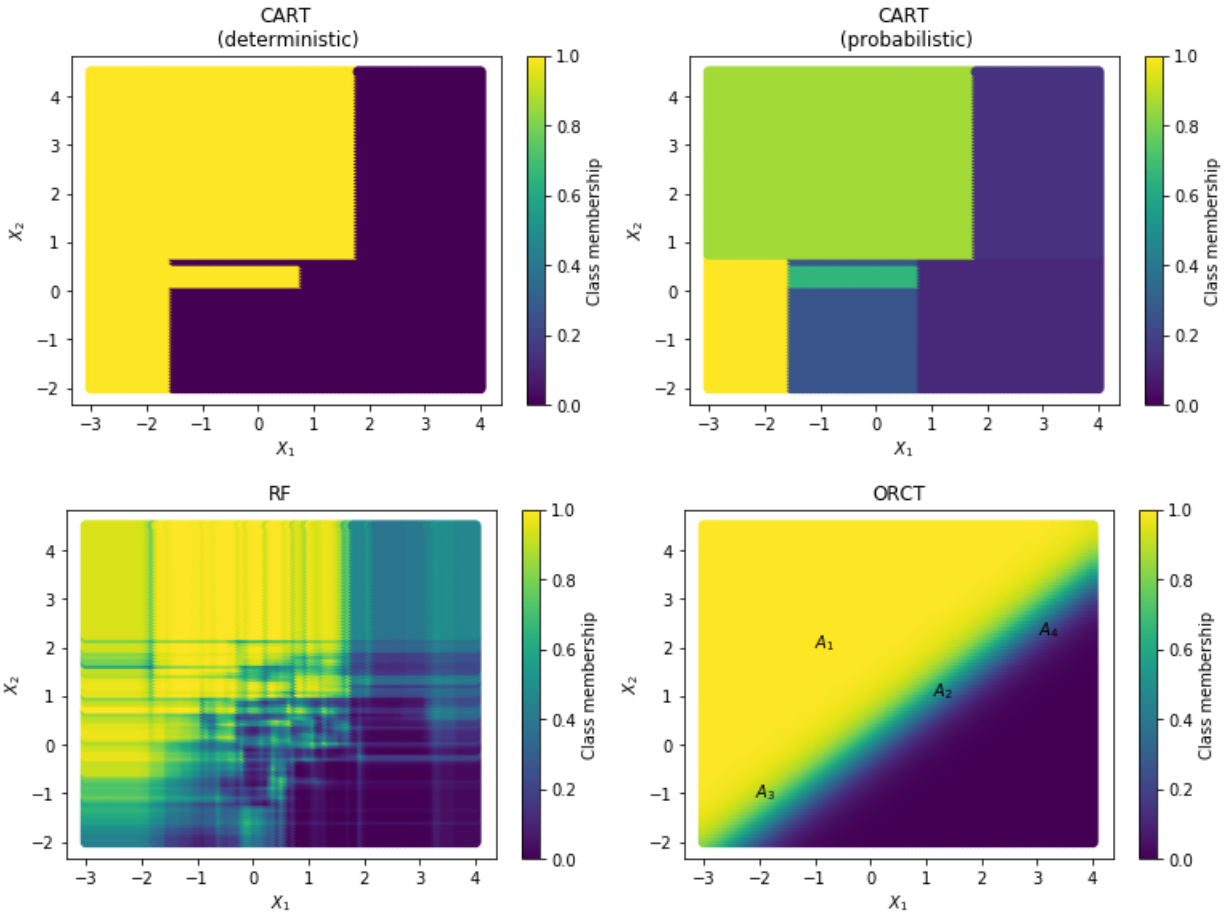


Figure 2.4: Heatmap of probabilities of class membership for deterministic CART, probabilistic CART, RF and ORCT on the simulated data set in Figure 2.3.

Thus, the objective function takes the following form:

$$\frac{1}{N}\sum_{i=1}^{N}\sum_{t\in\tau_L} P_{it}\left(\boldsymbol{a},\boldsymbol{\mu}\right)\sum_{k=1}^{K} W_{y_i k}C_{kt}. \tag{2.6}$$

Hence, given the training sample I split into K classes, the CDF F, the depth of the tree D and the misclassification costs $W_{y_i k}$, a mixed-integer non-linear optimization (MINLO) problem to build the proposed classification tree reads as follows:

$$\begin{aligned}
\min \quad & \frac{1}{N}\sum_{i=1}^{N}\sum_{t\in\tau_L} P_{it}\left(\boldsymbol{a},\boldsymbol{\mu}\right)\sum_{k=1}^{K} W_{y_i k}C_{kt} \\
\text{s.t.} \quad & \sum_{k=1}^{K} C_{kt} = 1,\ t\in\tau_L, \\
& \sum_{t\in\tau_L} C_{kt} \geq 1,\ k=1,\dots,K, \\
& a_{jt}\in[-1,1],\ j=1,\dots,p,\ t\in\tau_B, \\
& \mu_t \in [-1,1],\ t\in\tau_B, \\
& C_{kt}\in\{0,1\},\ k=1,\dots,K,\ t\in\tau_L,
\end{aligned} \tag{2.7}$$

where $P_{it}\left(\boldsymbol{a},\boldsymbol{\mu}\right)$ is defined as in (2.2).

The presence of binary decision variables in a framework where the objective function is highly complex non-convex could appear to be discouraging. Nevertheless, without loss of optimality, we can relax the binary decision variables $C_{kt},\ k=1,\dots,K,\ t\in\tau_L$, yielding to the continuous formulation we were looking for. Theorem 2.1 guarantees the equivalence of the resulting Non-Linear Continuous Optimization (NLCO) problem and the MINLO problem.

The NLCO problem, which will be referred henceforth as the Optimal Randomized Classification Tree (ORCT), reads as follows:

$$\begin{aligned}
\min \quad & \frac{1}{N}\sum_{i=1}^{N}\sum_{t\in\tau_L} P_{it}\left(\boldsymbol{a},\boldsymbol{\mu}\right)\sum_{k=1}^{K} W_{y_i k}C_{kt} \\
\text{s.t.} \quad & \sum_{k=1}^{K} C_{kt} = 1,\ t\in\tau_L, \\
& \sum_{t\in\tau_L} C_{kt} \geq 1,\ k=1,\dots,K, \\
& a_{jt}\in[-1,1],\ j=1,\dots,p,\ t\in\tau_B, \\
& \mu_t \in [-1,1],\ t\in\tau_B, \\
& C_{kt}\geq 0,\ k=1,\dots,K,\ t\in\tau_L,
\end{aligned} \tag{2.8}$$

where $C_{kt},\ k=1,\dots,K,\ t\in\tau_L$, can be seen as the probability that the class label k is

assigned to leaf node t.

Theorem 2.1. *There exists an optimal solution to* (2.8) *such that* $C_{kt} \in \{0,1\},\ k = 1,\dots,K,\ t \in \tau_L$.

Proof.

The continuity of the objective function, defined in (2.8) over a compact set, ensures the existence of an optimal solution of the optimization problem, by the Weierstrass Theorem. Let $(\boldsymbol{a}^*, \boldsymbol{\mu}^*, \boldsymbol{C}^*)$ be an optimal solution to (2.8). Fixing $(\boldsymbol{a}^*, \boldsymbol{\mu}^*)$, we have the following problem on the decision variables $C_{kt},\ k = 1,\dots,K,\ t \in \tau_L$:

$$
\begin{aligned}
\min \quad & \frac{1}{N}\sum_{i=1}^{N}\sum_{t\in\tau_L} P_{it}\left(\boldsymbol{a}^*, \boldsymbol{\mu}^*\right)\sum_{k=1}^{K} W_{y_i k} C_{kt} \\
\text{s.t.} \quad & \sum_{k=1}^{K} C_{kt} = 1,\ t \in \tau_L, \\
& \sum_{t\in\tau_L} C_{kt} \geq 1,\ k = 1,\dots,K, \\
& C_{kt} \geq 0,\ k = 1,\dots,K,\ t \in \tau_L,
\end{aligned}
$$

a transportation problem for which the existence of an integer optimal solution is well-known to hold, i.e., there is $\overline{\boldsymbol{C}} = (\overline{C}_{kt})$, with $\overline{C}_{kt} \in \{0,1\},\ k = 1,\dots,K,\ t \in \tau_L$, such that $(\boldsymbol{a}^*, \boldsymbol{\mu}^*, \overline{\boldsymbol{C}})$ is also optimal for (2.8). □

The prediction of future unlabeled data with predictor variables $\boldsymbol{x}$ that ORCT makes is probabilistic by construction, namely, the probabilities in (2.4) are returned where $\boldsymbol{x}_i$ is replaced by $\boldsymbol{x}$. In our computational experience, this prediction is made in a deterministic fashion by choosing the class for which the class membership probability is the highest.

2.1.3 ORCT with constraints on expected performance

Although classifiers seek a rule yielding a good overall classification rate, there are many cases in which misclassification has different consequences for different classes. It is then more appealing not to focus on the overall classification, but to obtain an acceptable overall performance while ensuring a certain level of performance in some classes. Our ORCT has two ways of achieving this. First, one could modify the misclassification costs (2.5) in the objective function (2.6), for different sets of values of $W_{y_i k}$. However, in this way, one has no direct control on the misclassification rates for critical classes. Second, ORCT is flexible enough to allow the incorporation of constraints on expected performance over the different classes explicitly. Indeed, define the random variable O_i,

$$
O_i = \begin{cases} 1, & \text{if individual } i \text{ is correctly classified} \\ 0, & \text{otherwise.} \end{cases}
$$

Given $k = 1, \ldots, K$, a Correct Classification Rate (CCR) over the $k-$th class, namely, ρ_k, is desired:

$$\frac{1}{|I_k|} \sum_{i \in I_k} O_i \geq \rho_k.$$

The expectation of achieving this performance can be written as:

$$\mathbb{E}\left[\frac{1}{|I_k|} \sum_{i \in I_k} O_i\right] = \frac{1}{|I_k|} \sum_{i \in I_k} \mathbb{E}\left[O_i\right] = \frac{1}{|I_k|} \sum_{i \in I_k} \sum_{t \in \tau_L} P_{it}\left(\boldsymbol{a}, \boldsymbol{\mu}\right) C_{kt} \geq \rho_k.$$

Hence, given the class $k = 1, \ldots, K$ to be controlled and its desired performance ρ_k, the following constraint would need to be added to the model:

$$\sum_{i \in I_k} \sum_{t \in \tau_L} P_{it}\left(\boldsymbol{a}, \boldsymbol{\mu}\right) C_{kt} \geq \rho_k |I_k|. \tag{2.9}$$

2.2 Theoretical properties

In this section, we explore some theoretical properties of our approach. First, we show the relationship between ORCT, which uses a randomized decision rule in each branch node, and other optimization-based approaches, which are deterministic, as CARTs are. We will prove that when the level of randomization decreases to zero, our ORCT converges to an Optimal Deterministic Classification Tree (hereafter, ODCT), i.e., those reviewed in Section 1.3.1. Second, we prove asymptotic results for the optimization problem attached to our ORCT when the training sample size grows to infinity.

We start by investigating the relationship between ORCT and the so-called ODCTs. Recall that ORCT uses the CDF F to make, at each branch node, the decision to move to the left or to the right child node, while ODCT makes this decision in a deterministic fashion. In order to show the convergence of ORCT to ODCT, we define a family of CDFs F_γ, parametrized by $\gamma > 0$, such that

$$\lim_{\gamma \to \infty} F_\gamma\left(\cdot\right) = \begin{cases} 1, & \text{if } (\cdot) \geq 0 \\ 0, & \text{otherwise} \end{cases}. \tag{2.10}$$

An example of this family can be defined using the logistic CDF as follows

$$F_\gamma\left(\cdot\right) = \frac{1}{1 + \exp\left(-\left(\cdot\right)\gamma\right)}, \ \gamma > 0, \tag{2.11}$$

provided that the argument is different from zero.

For each value of the parameter γ we have an ORCT, say ORCT(γ). The larger the value of γ, the closer the decision rule defined by F_γ is to a deterministic rule. In the limit case, when γ is equal to ∞, the decision rule is deterministic, using (2.10), and therefore the corresponding optimal classification tree is an ODCT. Thus, it is now easy to show the following property.

Proposition 2.1. *One has* $\lim_{\gamma\to\infty} ORCT(\gamma) = ODCT$.

In our numerical section, we will illustrate that with the logistic CDF family, and for large values of γ, ORCT yields better results than the ODCT reported in Bertsimas and Dunn [2017]. This means that, by just allowing a small level of randomization, corresponding to an almost deterministic cut, ORCT is preferable.

We now prove limit results for ORCT when the sample $I = \{(\boldsymbol{x}_i, y_i)\}_{1\leq i\leq N}$ is independent and identically distributed (i.i.d.) and its size grows to infinity. Unlike other optimization-based tree classifiers, the feasible region in (2.8) does not depend on the training sample I, and the objective function is continuous and separable on I. Thus, Problem (2.8) can be reformulated as the Sample Average Approximation (SAA) problem of some theoretical or true stochastic problem, which makes it possible to show consistency of the estimators of the optimal value and the set of optimal solutions to their true counterparts, as stated in Shapiro et al. [2009].

We will start by rewriting (2.8) into a more compact formulation. The decision variables $a_{jt},\ j = 1,\ldots,p,\ t \in \tau_B$, $\mu_t,\ t \in \tau_B$ and $C_{kt},\ k = 1,\ldots,K,\ t \in \tau_L$, are grouped into the n-dimensional decision vector $\boldsymbol{z} = (\boldsymbol{a}, \boldsymbol{\mu}, \boldsymbol{C})^T$, where $n = (p+1)\,|\tau_B| + K|\tau_L|$. Note that $\boldsymbol{z}$ comprises information on the cuts in the branch nodes of the tree as well as the assignments in the leaf nodes. The feasible region will be denoted by Z, where

$$Z = \Big\{\boldsymbol{z} = (\boldsymbol{a}, \boldsymbol{\mu}, \boldsymbol{C})^T \in \mathbb{R}^n : \boldsymbol{a} \in [-1,1]^{p\times|\tau_B|}, \boldsymbol{\mu} \in [-1,1]^{|\tau_B|}, \sum_{k=1}^{K} C_{kt} = 1, t \in \tau_L,$$
$$\sum_{t\in\tau_L} C_{kt} \geq 1, k = 1,\ldots,K, C_{kt} \geq 0,\ k = 1,\ldots,K,\ t \in \tau_L\Big\}$$

which is a non-empty compact subset of $\mathbb{R}^n$. Then, we have a sample of N i.i.d. realizations of a random vector $\boldsymbol{\xi} = (\boldsymbol{X}, Y)$ whose probability distribution E is supported on a set $\Xi \subset \mathbb{R}^{p+1}$, i.e., we have $I = \{\boldsymbol{\xi}_i = (\boldsymbol{x}_i, y_i)\}_{1\leq i\leq N}$. Thus, Problem (2.8) can be written as:

$$\min_{\boldsymbol{z}\in Z} \left\{\hat{g}_N(\boldsymbol{z}) := \frac{1}{N}\sum_{i=1}^{N} G(\boldsymbol{z}, \boldsymbol{\xi}_i)\right\}, \tag{2.12}$$

where

$$G(\boldsymbol{z}, \boldsymbol{\xi}_i) = \sum_{t\in\tau_L} P_{it}(\boldsymbol{a}, \boldsymbol{\mu}) \sum_{k=1}^{K} W_{y_i k} C_{kt}. \tag{2.13}$$

Problem (2.12) can be seen as the SAA problem associated with the true stochastic problem:

$$\min_{\boldsymbol{z}\in Z} \{g(\boldsymbol{z}) := \mathbb{E}[G(\boldsymbol{z}, \boldsymbol{\xi})]\}. \tag{2.14}$$

That is, for any $\boldsymbol{z} \in Z$, the estimator of the expected value $g(\boldsymbol{z})$, $\hat{g}_N(\boldsymbol{z})$, is obtained by averaging values $G(\boldsymbol{z}, \boldsymbol{\xi}_i),\ i = 1,\ldots,N$.

Let ϑ^* and S denote the optimal value and the set of optimal solutions of the true problem (2.14), respectively. Similarly, $\hat{\vartheta}_N$ and $\hat{S}_N$ will denote the optimal value and the set of optimal solutions of the SAA problem (2.12), respectively. Our goal is to prove the consistency of the SAA estimators to the true counterparts. The estimator $\hat{\vartheta}_N$ of the parameter ϑ is said to be consistent, in the sense of Shapiro et al. [2009], if $\hat{\vartheta}_N$ converges with probability 1 (w.p.1) to ϑ as $N \to \infty$. For the set of optimal solutions, Shapiro et al. [2009] establish consistency of the estimator $\hat{S}_N$ to S when the deviation of $\hat{S}_N$ from S, $\mathbb{D}(\hat{S}_N, S)$, converges w.p.1 to 0 as $N \to \infty$, where $\mathbb{D}(\hat{S}_N, S)$ actually represents the distance between both sets.

Theorem 2.2. *$\hat{\vartheta}_N$ and $\hat{S}_N$ are consistent estimators of ϑ^* and S.*

Proof.

The result is a direct consequence of Theorems 7.48 and 5.3 in Shapiro et al. [2009]. We will first show that the conditions of Theorem 7.48 hold. Indeed, we have that the sample is i.i.d. and that Z is a nonempty compact subset of $\mathbb{R}^n$. We also have that for any $\boldsymbol{z} \in Z$ the function $G(\cdot, \boldsymbol{\xi})$ is continuous at $\boldsymbol{z}$ for almost every $\boldsymbol{\xi} \in \Xi$, since the CDF F is a continuous function by assumption. Finally, $G(\boldsymbol{z}, \boldsymbol{\xi}),\ \boldsymbol{z} \in Z$, is dominated by the following constant function

$$|G(\boldsymbol{z}, \boldsymbol{\xi})| \leq \max_{k \neq y_i} \{W_{y_i k}\}.$$

This constant function is integrable since the support of the probability distribution E is contained in a compact set, namely $\Xi \subset [0,1]^p \times [1, K]$. From Theorem 7.48, we know that $g(\boldsymbol{z})$ is finite valued and continuous on Z and $\hat{g}_N(\boldsymbol{z})$ converges to $g(\boldsymbol{z})$ w.p.1, as $N \to \infty$, uniformly in $\boldsymbol{z} \in Z$. In addition, we have that $\hat{S}_N, S \subset Z$. This, together with the continuity of both $\hat{g}_N$ and g, implies that $\hat{S}_N, S \neq \emptyset$. Thus, the conditions of Theorem 5.3 hold, and the desired result follows. □

2.3 Computational experiments

The purpose of this section is to illustrate the performance of our ORCT, against the natural benchmarking tree-based methods.

We can draw the following conclusions from our computational experiments. First, in terms of classification accuracy, our approach outperforms CART, OC1 and a benchmark decision tree approach based on integer programming, is comparable to *oblique.tree*, and is close to the local-search heuristic approach in Dunn [2018] and RFs. Second, we show that our running times are low. Third, we illustrate the flexibility of ORCTs to produce variable importance metrics, and to handle cost-sensitive constraints, unlike heuristic approaches.

Several well-known data sets from the UCI Machine Learning Repository [Lichman, 2013] have been chosen for the computational experiments. Table 2.1 reports the size, the number

Table 2.1: Information about the data sets considered.

Data set	Abbreviation	N	p	K	Class distribution
Connectionist-bench-sonar	CBS	208	60	2	55% - 45%
Wisconsin	W	569	30	2	63% - 37%
Credit-approval	CA	653	37	2	55% - 45%
Pima-indians-diabetes	PID	768	8	2	65% - 35%
Statlog-project-German-credit	SPGC	1000	48	2	70% - 30%
Ozone-level-detection-one	OLDO	1848	72	2	97% - 3%
Spambase	SB	4601	57	2	61% - 39%
Magic-gamma-telescope	MGT	19020	10	2	65% - 35%
Iris	I	150	4	3	33.3%-33.3%-33.3%
Wine	Wi	178	13	3	40%-33%-27%
Seeds	S	210	7	3	33.3%-33.3%-33.3%
Thyroid-disease-ann-thyroid	TADT	3772	21	3	92.5%-5%-2.5%
Car-evaluation	CE	1728	15	4	70%-22%-4%-4%

of predictor variables, and the number of classes as well as the corresponding class distribution. This selection comprises data sets of different nature: from small-sized data sets, such as Iris, to larger data sets, such as Thyroid-disease-ann-thyroid, Spambase and Magic-gamma-telescope; the class distribution is also diverse, from Iris or Seeds, which are balanced data sets, to Ozone-level-detection-one, which is highly imbalanced; lastly, and in addition to numerical (continuous as well as integer) predictor variables, we have also considered data sets with categorical predictor variables, modeled, as usual, through dummies.

The NLCO model (2.8) has been implemented using Pyomo optimization modeling language [Hart et al., 2011, 2017] in Python 3.7 [Python Core Team, 2015]. As a solver, we have used IPOPT 3.11.1 [Wächter and Biegler, 2006]. Our experiments have been conducted on a PC, with an Intel® Core™ i7-7700 CPU 3.60GHz processor and 32 GB RAM. The operating system is 64 bits.

To train the ORCT, we solve the NCLP problem 20 times, starting from different random initial solutions.

The CDF chosen has been the logistic one, see Equation (2.11). In our computational experience, we illustrate that a small level of randomization is enough for obtaining good results. Thus, we have set $\gamma = 512$ for both constructing and testing our model.

Equal misclassification weights, $W_{y_i k} = 0.5,\ i = 1, \ldots, N,\ k = 1, \ldots, K,\ k \neq y_i$, have been used for the experiments.

Each data set has been split into two subsets: the training subset (75%) and the test subset (25%). The ORCT is built over the training subset and, then, its performance is evaluated by determining the out-of-sample accuracy over the test subset. This procedure has been repeated ten times, and average results are reported.

We will compare our ORCT of depths $D = 1, \ldots, 4$ with: the full CART, as implemented

Table 2.2: Results for $D = 1$ in terms of the out-of-sample accuracy.

Data set	Out-of-sample accuracy							Average time (in secs)
	CART	OC1	*oblique.tree*	OCT-H MIO	OCT-H LS	RF	ORCT	
CBS	70.0(7)	70.8(5)	72.5(4)	70.4(6)	77.3(2)	83.1(1)	76.3(3)	8
W	92.0(7)	94.1(4)	93.7(5)	93.1(6)	94.8(3)	95.5(2)	96.4(1)	10
CA	85.7(4)	82.0(7)	83.5(6)	87.9(1)	86.0(3)	86.7(2)	83.7(5)	7
PID	74.2(4)	60.7(7)	76.0(2)	71.6(6)	73.1(5)	76.3(1)	76.0(2)	6
SPGC	72.1(3)	68.5(7)	73.8(2)	71.6(6)	72.1(3)	75.2(1)	72.1(3)	10
OLDO	95.6(5)	95.5(6)	92.9(7)	96.8(1)	96.2(4)	96.4(3)	96.7(2)	66
SB	89.2(6)	92.3(4)	92.7(3)	83.6(7)	94.2(2)	95.1(1)	89.8(5)	49
MGT	82.1(4)	78.8(6)	82.7(3)	-	86.9(2)	87.7(1)	79.9(5)	122
Average	82.6(5.0)	80.3(5.8)	83.5(4.0)	82.1(4.7)	85.1(3.0)	87.0(1.5)	83.9(3.3)	35

in the `rpart` R package [Therneau et al., 2015]; two other greedy approaches that implement oblique cuts: the full OC1 and the full *oblique.tree*, as implemented in Murthy et al. [1994] and Csárdi and Truong [2012], respectively, with the default tuning parameters; the OCT-H MIO, proposed in [Bertsimas and Dunn, 2017] that also constructs oblique cuts, at the same depth as ORCT; the OCT-H LS in Dunn [2018] that employs a local-search heuristic for building oblique trees at maximum depth $D = 10$; and Random Forests from `randomForest` R package [Liaw and Wiener, 2002] with the default tuning parameters.

2.3.1 Results for ORCT

Tables 2.2, 2.3, 2.4 and 2.5 present the comparison of ORCT at depth $D = 1, 2, 3$ and 4, respectively, against the benchmarks described above. Figures 2.5, 2.6, 2.7 and 2.8 depict these results, where ORCT is highlighted in striped grey. Note that for data sets with $K \geq 3$, the ORCT at depth $D = 1$ would become infeasible due to the set of constraints (2.3).

Each table and each figure displays, per data set, the average out-of-sample accuracy over the ten runs for CART, OC1, *oblique.tree*, RF and ORCT, as well as the average out-of-sample accuracy across all data sets. Results for OCT-H MIO and OCT-H LS are taken from [Bertsimas and Dunn, 2017] and [Dunn, 2018], respectively. Note that the Magic-gamma-telescope data set is not available in [Bertsimas and Dunn, 2017]. Tables also include information about the average execution time for ORCT. This time involves the data reading, the scaling of the training set, the random generation of initial solutions, the optimization time, the scaling of the test set using the scale parameters obtained in the training and the evaluation of performance. The average solving time of an instance of a specific optimization problem is also shown. Moreover, for each data set, we rank the methods by their accuracy. The rank is shown in parentheses. A rank of 1 indicates that the method is the best in terms of out-of-sample accuracy on a given data set and a rank of 7 indicates that the method performed the worst. The average accuracy and rank of each method across all data sets are found at the bottom of the table.

We start by discussing the results for $D = 1$ in Table 2.2 and Figure 2.5. We will say that two methods have a comparable accuracy if they differ in less than 1 percentage point. ORCT

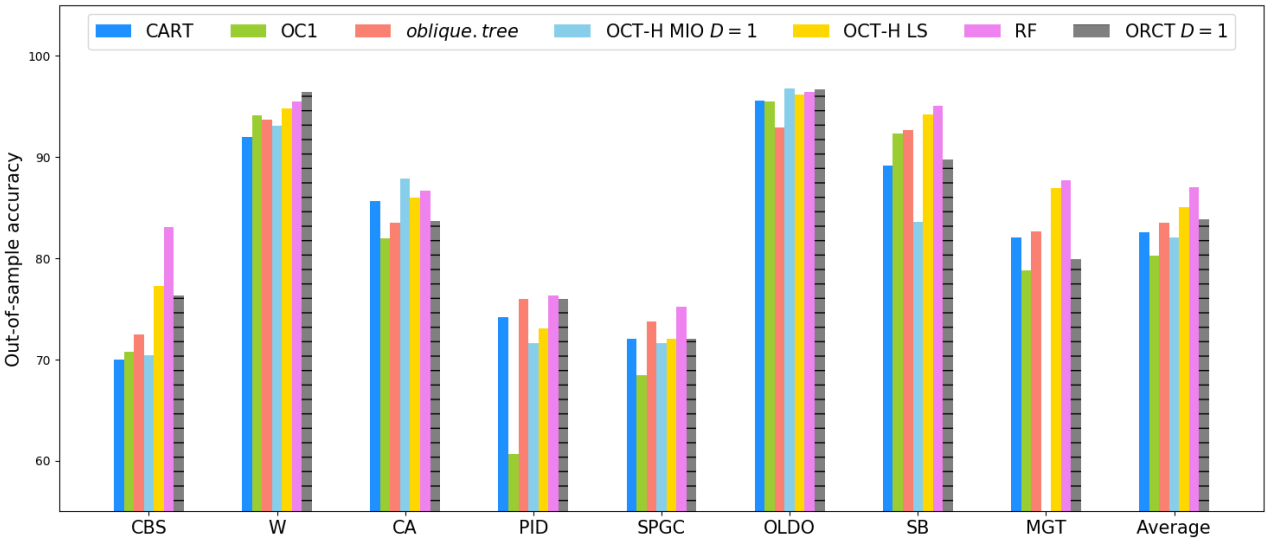


Figure 2.5: Comparison of ORCT at depth $D = 1$ and other tree-based methods in terms of the out-of-sample accuracy.

generally outperforms CART, even with depth $D = 1$, i.e., one single oblique cut. This is the case for all data sets except for CA and MGT. Regarding OC1, ORCT is generally better except for SB. With respect to *oblique.tree*, ORCT is comparable (CA, PID) or better (CBS, W, OLDO) in five out of eight datasets. ORCT is generally better than OCT-H MIO except for CA. With respect to OCT-H LS, ORCT outperforms or is comparable in four out of eight data sets (W, PID, SPGC, OLDO). Finally, RF tends to lead in performance among all one-single-tree-based methods we have tested. Nevertheless, ORCTs are comparable to RFs in some data sets (PID, OLDO) and even competitive in others (W). In terms of the average performance across all data sets, ORCT is superior to CART, OC1 and OCT-H MIO, and is comparable to *oblique.tree*. Although OCT-H LS has a higher average accuracy across all data sets than ORCT, they both have the same average rank. RF presents the best average accuracy as well as rank.

We continue by discussing the results for $D = 2$ in Table 2.3 and Figure 2.6. The out-of-sample accuracy has improved for both CA and MGT, in comparison with $D = 1$, but the same conclusions as in Table 2.2 can be drawn for two-class problems when comparing ORCT with CART and OCT-H MIO. The outperformance of ORCT against CART and OCT-H MIO is also clear for I, W, S and CE. This is not the case for TDAT, in which ORCT is comparable to OCT-H MIO although CART outperforms these two methods. Compared to OC1, ORCT is comparable (I) or outperforms (the rest except for SB, TDAT and CE) in ten out of the thirteen data sets. With respect to *oblique.tree*, ORCT is comparable (CA, PID, I, W, S) or outperforms (CBS, W, OLDO) in eight out of the thirteen data sets. Regarding to OCT-H LS, ORCT is competitive (CBS) or outperforms (W, PID, SPGC, OLDO, I, W, S) in eight out of the thirteen data sets. Finally, ORCT is again comparable to RF in some of the data sets (PID, OLDO) and

Table 2.3: Results for $D = 2$ in terms of the out-of-sample accuracy.

Data set	Out-of-sample accuracy							Average time (in secs)
	CART	OC1	*oblique.tree*	OCT-H MIO	OCT-H LS	RF	ORCT	
CBS	70.0(6)	70.8(5)	72.5(4)	70.0(6)	77.3(3)	83.1(1)	77.5(2)	27
W	92.0(7)	94.1(4)	93.7(5)	93.1(6)	94.8(3)	95.5(2)	96.2(1)	36
CA	85.7(4)	82.0(7)	83.5(6)	87.9(1)	86.0(3)	86.7(2)	84.2(5)	20
PID	74.2(4)	60.7(7)	76.0(2)	71.4(6)	73.1(5)	76.3(1)	76.0(2)	20
SPGC	72.1(4)	68.5(7)	73.8(2)	70.4(6)	72.1(4)	75.2(1)	72.8(3)	40
OLDO	95.6(5)	95.5(6)	92.9(7)	96.8(1)	96.2(4)	96.4(3)	96.7(2)	267
SB	89.2(6)	92.3(4)	92.7(3)	85.7(7)	94.2(2)	95.1(1)	89.8(5)	58
MGT	82.1(4)	78.8(6)	82.7(3)	-	86.9(2)	87.7(1)	80.8(5)	551
I	92.7(7)	95.4(3)	96.8(1)	95.1(5)	94.6(6)	95.4(3)	95.9(2)	5
Wi	88.6(7)	90.9(6)	96.1(3)	91.1(5)	95.1(4)	98.6(1)	96.6(2)	9
S	90.2(7)	90.4(6)	94.2(1)	90.6(5)	91.7(4)	92.5(3)	94.2(1)	7
TDAT	99.1(2)	97.9(4)	97.7(5)	92.5(6)	99.7(1)	99.1(2)	92.2(7)	111
CE	88.1(5)	94.7(2)	93.9(3)	87.5(7)	97.8(1)	88.0(6)	89.8(4)	42
Average	86.1(5.2)	85.5(5.1)	88.1(3.5)	86.0(5.1)	89.2(3.2)	90.0(2.1)	87.9(3.2)	92

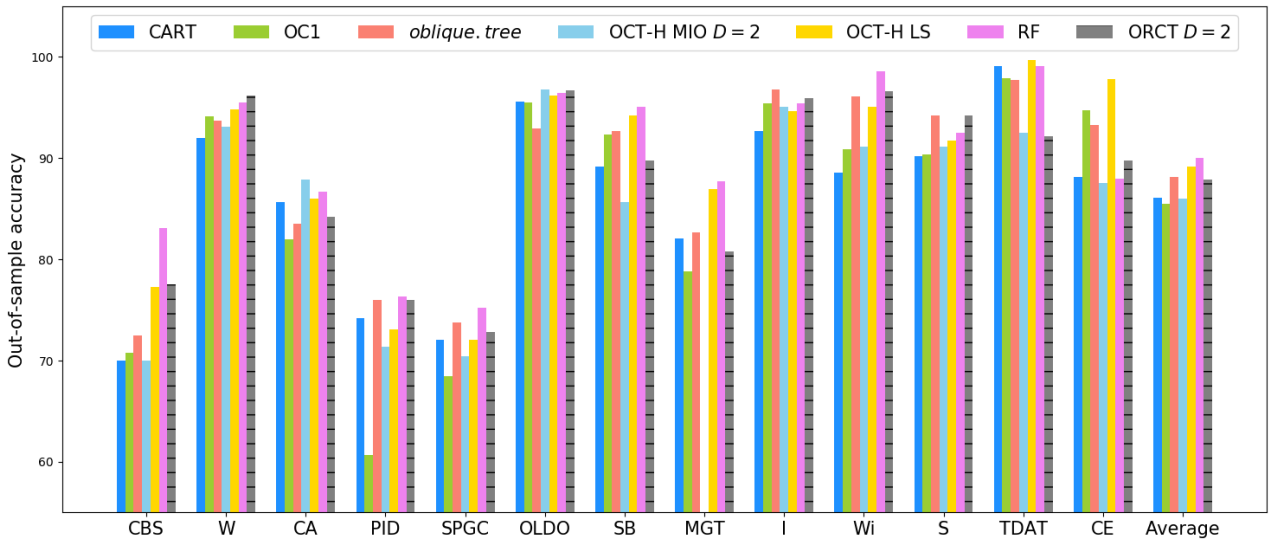


Figure 2.6: Comparison of ORCT at depth $D = 2$ and other tree-based methods in terms of the out-of-sample accuracy.

Table 2.4: Results for $D = 3$ in terms of the out-of-sample accuracy.

Data set	Out-of-sample accuracy							Average time (in secs)
	CART	OC1	*oblique.tree*	OCT-H MIO	OCT-H LS	RF	ORCT	
CBS	70.0(7)	70.8(5)	72.5(4)	70.8(5)	77.3(2)	83.1(1)	77.1(3)	84
W	92.0(7)	94.1(4)	93.7(6)	94.0(5)	94.8(3)	95.5(2)	96.4(1)	107
CA	85.7(4)	82.0(7)	83.5(6)	87.9(1)	86.0(3)	86.7(2)	84.4(5)	62
PID	74.2(4)	60.7(7)	76.0(2)	71.4(6)	73.1(5)	76.3(1)	75.2(3)	73
SPGC	72.1(4)	68.5(7)	73.8(2)	71.0(5)	72.1(4)	75.2(1)	73.4(3)	116
OLDO	95.6(5)	95.5(6)	92.9(7)	96.8(1)	96.2(4)	96.4(3)	96.7(2)	918
SB	89.2(6)	92.3(4)	92.7(3)	86.6(7)	94.2(2)	95.1(1)	89.8(5)	572
MGT	82.1(5)	78.8(6)	82.7(4)	-	86.9(2)	87.7(1)	82.9(3)	2018
I	92.7(7)	95.4(4)	96.8(1)	95.1(5)	94.6(6)	95.4(3)	95.7(2)	12
Wi	88.6(7)	90.9(6)	96.1(3)	92.9(5)	95.1(4)	98.6(1)	96.6(2)	26
S	90.2(7)	90.4(6)	94.2(1)	91.3(5)	91.7(4)	92.5(3)	94.0(2)	22
TDAT	99.1(2)	97.9(4)	97.7(5)	92.5(6)	99.7(1)	99.1(2)	92.5(7)	367
CE	88.1(5)	94.7(2)	93.9(3)	87.5(7)	97.8(1)	88.0(6)	91.7(4)	161
Average	86.1(5.4)	85.5(5.2)	88.1(3.6)	86.5(4.8)	89.2(3.1)	90.0(2.1)	88.2(3.2)	349

even competitive in others (W, I, Wi, S). On average across all data sets, RF is the best method and ORCT is superior to CART and OCT-H MIO again, and is comparable to *oblique.tree*. A slightly greater average rank is observed for ORCT at depth $D = 2$, although OCT-H LS still produces a higher average accuracy.

Similar conclusions can be drawn for $D = 3$ and $D = 4$, see Tables 2.4 and 2.5, and Figures 2.7 and 2.8, respectively. The out-of-sample accuracies of ORCT at depths $D = 3$ and 4 have not significantly changed in most cases except for CE and MGT. The out-of-sample accuracies for both are already comparable to *oblique.tree* and superior to CART, respectively. The comparisons between ORCT and OC1, ORCT and OCT-H LS, and ORCT and RF remain the same. On average across all data sets, ORCT, while maintaing a slightly better average rank, tends to close its gap with OCT-H LS regarding the average out-of-sample accuracy across all data sets.

In summary, these numerical results illustrate that, in terms of accuracy, ORCT outperforms CART, OC1 and OCT-H MIO, while ORCT is comparable to *oblique.tree* and manages to get close to OCT-H LS and RFs. In contrast to these three latter heuristic approaches, ORCT has a direct and effective control on critical issues such as cost-sensitiveness, as will be seen in Section 2.3.3.

Regarding the computational time taken by ORCT, Tables 2.2, 2.3, 2.4 and 2.5 report low running times, compared to OCT-H MIO, where a CPU time limit from 5 to 15 minutes was imposed in Bertsimas and Dunn [2017]; excluding the computing time devoted to preprocessing that involves the tuning of a parameter for which time limits of 60 seconds were imposed to OCT-H MIO, which is not the case in ORCT. OCT-H LS also requires parameter tuning. Although limited information about computing times is found in Dunn [2018], a large gain in time is reported for a particular data set, compared to OCT-H MIO.

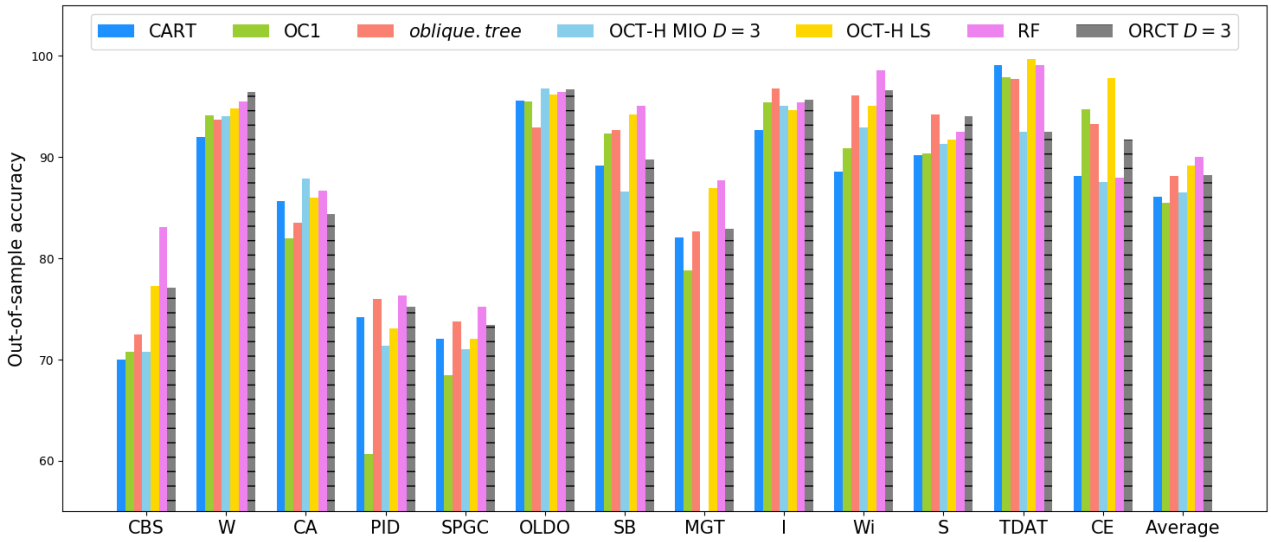


Figure 2.7: Comparison of ORCT at depth $D = 3$ and other tree-based methods in terms of the out-of-sample accuracy.

Table 2.5: Results for $D = 4$ in terms of the out-of-sample accuracy.

Data set	Out-of-sample accuracy							Average time (in secs)
	CART	OC1	*oblique.tree*	OCT-H MIO	OCT-H LS	RF	ORCT	
CBS	70.0(7)	70.8(6)	72.5(4)	71.5(5)	77.3(2)	83.1(1)	76.5(3)	210
W	92.0(7)	94.1(4)	93.7(6)	94.0(5)	94.8(3)	95.5(2)	96.2(1)	351
CA	85.7(4)	82.0(7)	83.5(6)	87.9(1)	86.0(3)	86.7(2)	84.6(5)	203
PID	74.2(4)	60.7(7)	76.0(3)	70.3(6)	73.1(5)	76.3(1)	76.1(2)	208
SPGC	72.1(4)	68.5(7)	73.8(2)	71.0(6)	72.1(4)	75.2(1)	72.8(3)	415
OLDO	95.6(5)	95.5(6)	92.9(7)	96.8(1)	96.2(4)	96.4(3)	96.7(2)	3360
SB	89.2(6)	92.3(4)	92.7(3)	86.6(7)	94.2(2)	95.1(1)	89.8(5)	1717
MGT	82.1(5)	78.8(6)	82.7(4)	-	86.9(2)	87.7(1)	84.3(3)	5603
I	92.7(7)	95.4(3)	96.8(1)	95.1(5)	94.6(6)	95.4(3)	96.2(2)	31
Wi	88.6(7)	90.9(6)	96.1(2)	91.6(5)	95.1(4)	98.6(1)	95.7(3)	69
S	90.2(7)	90.4(6)	94.2(1)	91.3(5)	91.7(4)	92.5(3)	93.1(2)	58
TDAT	99.1(2)	97.9(4)	97.7(5)	92.5(7)	99.7(1)	99.1(2)	93.1(6)	1051
CE	88.1(5)	94.7(2)	93.3(4)	87.5(7)	97.8(1)	88.0(6)	93.6(3)	468
Average	86.1(5.4)	85.5(5.2)	88.1(3.7)	86.3(5.0)	89.2(3.1)	90.0(2.1)	88.4(3.1)	1057

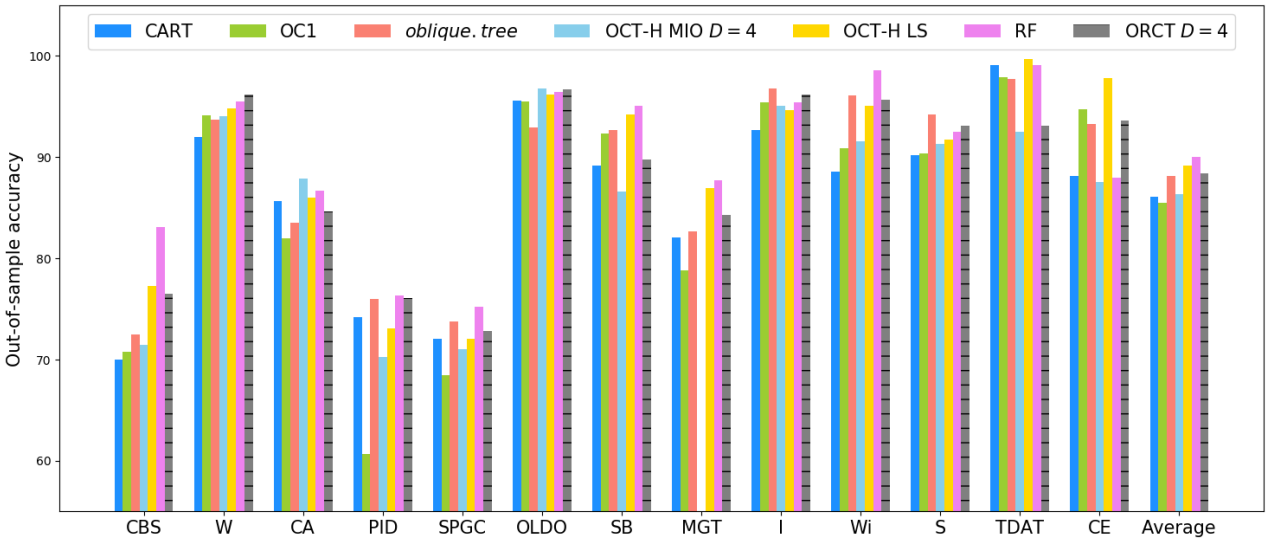


Figure 2.8: Comparison of ORCT at depth $D = 4$ and other tree-based methods in terms of the out-of-sample accuracy.

2.3.2 Results for variable importance measures via ORCTs

Measuring the importance of predictor variables in RFs has been thoroughly studied in the literature. In particular, the two most popular importance measures for forests are outlined in Biau and Scornet [2016]: the Mean Decrease Impurity (MDI) and the Mean Decrease Accuracy (MDA). The MDI takes advantage of the splitting criterion used for growing classic decision trees: given an impurity measure, the predictor variable that maximizes the decrease of impurity together with its corresponding splitting threshold are chosen. Thus, the MDI of predictor variable j is the average over every tree built of the decrease in impurity of splits along that variable, weighted with the fraction of individuals falling in the corresponding branch node. The MDA is based on the following notion: if a predictor variable is not influential, permuting the values it takes should not affect the prediction accuracy of the forest. Thus, the MDA of predictor variable j is the average over every tree built of the difference in accuracy before and after the permutation.

In our case we have two straightforward and inexpensive ways of measuring the importance of predictor variables by analyzing the distribution of the absolute values of coefficients $\boldsymbol{a_{jt}}$. The first metric to measure the importance of predictor variable j, $j = 1, \ldots, p$, is to sum the absolute values of the coefficients a_{jt} for all branch nodes $t \in \tau_B$, called hereafter the Sum Importance Measure (SIM):

$$\text{SIM}_j = \sum_{t \in \tau_B} |a_{jt}|.$$

The second metric is the so-called Maximum Importance Measure (MIM), which takes instead

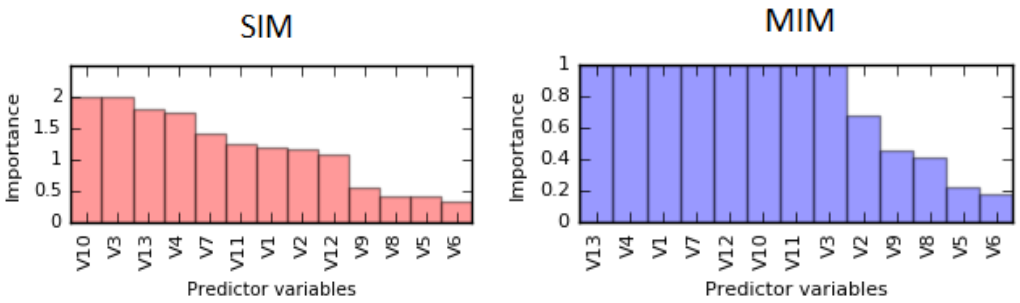


Figure 2.9: ORCT variable importance measures for the Wine data set.

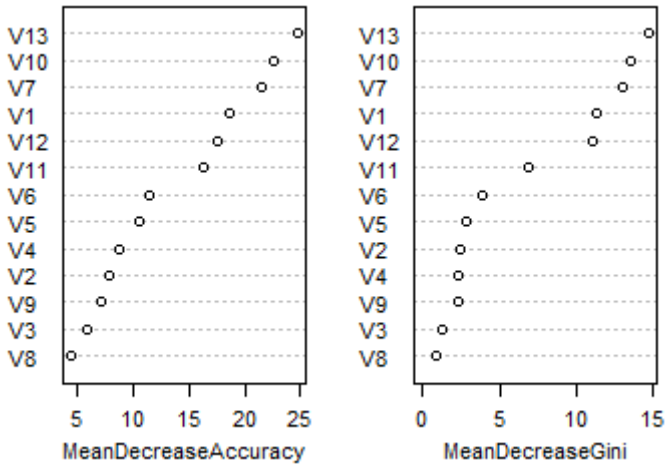


Figure 2.10: RF variable importance measures for the Wine data set.

the maximum among all the branch nodes $t \in \tau_B$:

$$\text{MIM}_j = \max_{t \in \tau_B} |a_{jt}|.$$

For illustrative purposes, these variable importance measures have been evaluated for the Wine and Car-evaluation data sets in Table 2.1, see Figures 2.9 and 2.11. Both measures have been evaluated over the resultant ORCT of the tenth run. The variable importance measures for RFs, the MDA and the MDI using the Gini index as the impurity measure, are also depicted for the aforementioned data sets. These can be found in Figures 2.10 and 2.12 and have been obtained with the `randomForest` R package. The message conveyed by these plots is, in general, in agreement. For instance, V8 in the Wine data set is not as important as V10. The same conclusion can be drawn for the Car-evaluation data set.

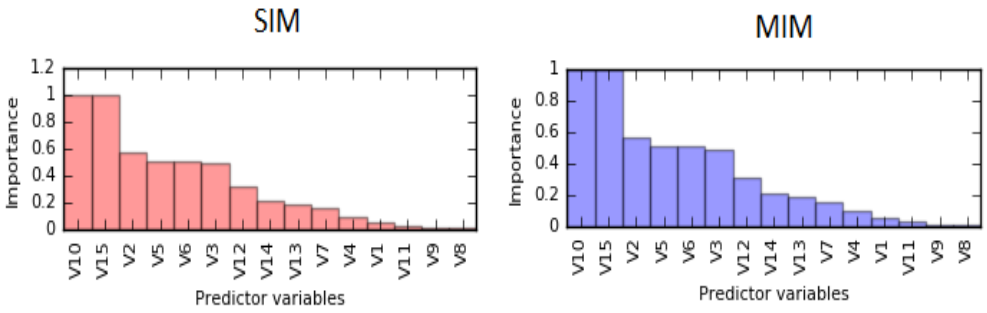


Figure 2.11: ORCT variable importance measures for the Car-evaluation data set.

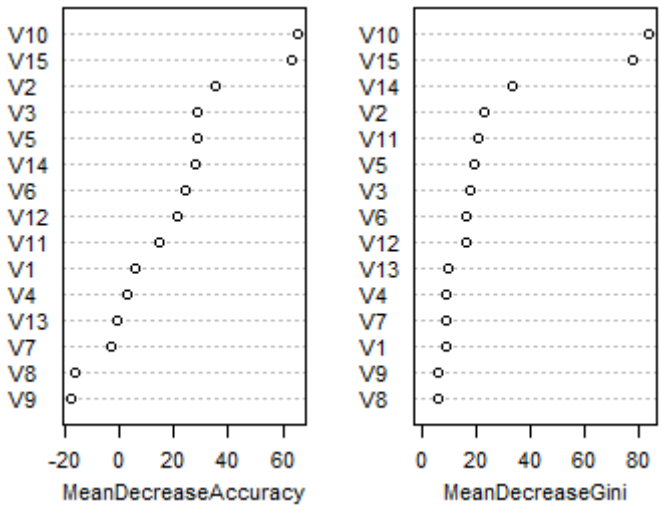


Figure 2.12: RF variable importance measures for the Car-evaluation data set.

2.3.3 Results for ORCT with constraints on expected performance

The Pima-indians-diabetes data set, from Lichman [2013], consists of a sample of 768 patients of Pima Indian heritage in the USA, on which 8 predictor variables are measured. The target variable is whether the patient shows signs of diabetes or not. Diabetics are the positive class and represent the 35% of the entire sample. Firstly, we have run our ORCT without constraints on expected performance by imposing a correct classification percentage over the positive class, ρ_+, equal to zero. See the first row in Table 2.6.

Table 2.6: Results with constraints on expected performance over the positive class in the Pima-indians-diabetes data set.

Imposed *TPR* (ρ_+)	TPR_{train}	TPR_{test}	TNR_{train}	TNR_{test}	CCR_{train}	CCR_{test}
0	60.8	56.4	90.5	87.8	80.3	76.2
62.5	63.9	59.0	88.6	86.0	80.1	76.1
65.0	65.8	60.9	87.3	84.7	79.9	75.9
67.5	68.4	62.5	85.5	83.1	79.6	75.5
70.0	71.1	64.3	83.6	81.8	79.3	75.3
72.5	73.7	67.4	81.4	80.1	78.7	75.4
75.0	75.8	68.1	79.3	77.4	78.1	73.9
77.5	78.9	72.9	77.2	75.9	77.8	74.6
80.0	81.3	73.1	74.5	72.7	76.9	72.7
82.5	83.9	76.7	71.9	69.0	76.0	71.6
85	86.5	80.9	68.4	66.6	74.6	71.6

The positive class, diabetics, is the worst classified, with an average True Positive Rate (TPR) of 60.8 over the ten training subsets and 56.4 over the ten test subsets. The negative class, non-diabetics, presents an average True Negative Rate (TNR) of 90.5 and 87.8 over the ten training and test subsets, respectively. In this setting, it is preferable to better classify diabetic patients, since diagnosing a diabetic as non-diabetic is more critical (in terms of missing medical treatment) than the other way around.

In this regard, a performance constraint over the positive class has been added to the ORCT for several values of the threshold ρ_+. We have considered a grid of values for ρ_+ varying from 62.5, a slightly higher value than the training TPR obtained with $\rho_+ = 0$, to 85.0 in steps of 2.5 units. Note that constraints on expected performance (3.8) are imposed over the training sample, so they might not be satisfied on an independent sample. Indeed, results in Table 2.6 show how these thresholds are fulfilled in the training subsets but this is not necessarily the case in the test subsets; even so, we observe that the test TPR increases with ρ_+. Figure 2.13 supports this observation, in which the $\text{TPR}_{\text{train}}$ and the TPR_{test} are depicted as a function of the imposed TPR (ρ_+). There, we can see that there exists a good linear fit. In fact, the regression

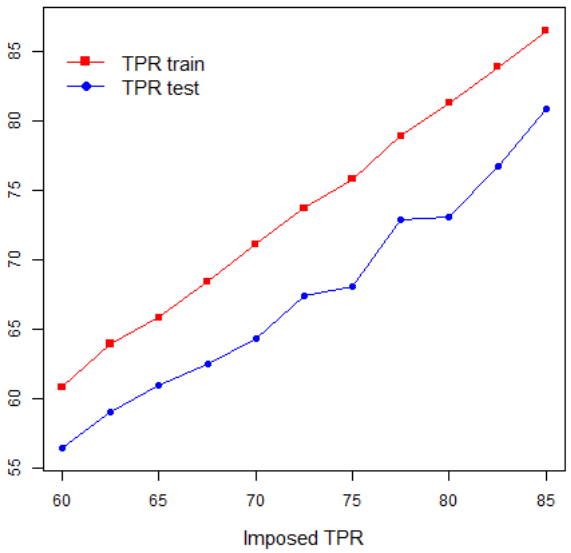


Figure 2.13: $\text{TPR}_{\text{train}}$ and TPR_{test} drawn as a function of the imposed TPR (ρ_+) for Pima-indians-diabetes data set.

.

models' coefficients and their corresponding coefficients of determination are the following:

$$\text{TPR}_{\text{train}} = -0.36 + 1.02\rho_+, R^2 = 0.9993,$$

$$\text{TPR}_{\text{test}} = -0.65 + 0.94\rho_+, R^2 = 0.9754.$$

A clear overfitting, almost independent of the threshold imposed, is also detected; but it is possible to determine the required imposed TPR in order to obtain a successful TPR_{test}. There is a price to pay for achieving such high TPRs: the TNRs decrease as we demand larger thresholds, see Figure 2.14.

2.4 Conclusions

In this chapter, we have proposed a new optimization-based approach to build classification trees. By replacing the binary decisions with randomized decisions, the resulting optimization problem is smooth and only contains continuous decision variables, allowing one to use gradient information. The computational experience reported shows that, with low running times, we outperform recent benchmarks, getting closer to and sometimes better than the performance of Random Forests. Moreover, we can model cost-sensitive constraints, having a direct and effective control on the accuracy of critical classes.

Several extensions to our approach are attractive. First, if the user wants to improve both

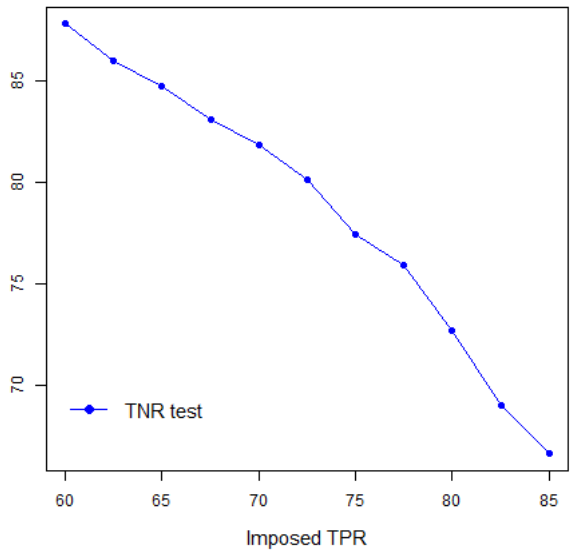


Figure 2.14: TNR_{test} depicted as a function of the imposed TPR (ρ_+) for Pima-indians-diabetes data set.

accuracy and computing times using deeper trees, at the expense of interpretability, then one can develop a local-search procedure to embed our algorithm, as done successfully in Dunn [2018] for the integer programming approach in Bertsimas and Dunn [2017]. Second, it is difficult in classic decision trees to control the number of predictor variables used. Making our approach sparse by means of regularizations, i.e., by using a lasso-type objective, is also an interesting research question, which is handled in Chapter 3.

Chapter 3

Sparsity in Optimal Randomized Classification Trees

In this chapter, we propose a continuous optimization approach that extends the methodology in Chapter 2 and build sparse optimal classification trees, with the aim of using fewer predictor variables in the cuts as well as along the whole tree. Both types of sparsity, namely local and global, are modeled by means of regularizations with polyhedral norms, the ℓ_1-norm and the ℓ_∞-norm, respectively. Theoretical results on the range of the sparsity parameters are shown. The computational experience reported supports the usefulness of our methodology. In all our data sets, local and global sparsity can be improved without harming classification accuracy. Unlike greedy approaches, our ability to easily trade in some of our classification accuracy for a gain in global sparsity is shown.

The chapter is organized as follows. In Section 3.1, we detail the construction of the Sparse Optimal Randomized Classification Tree (S-ORCT). Some theoretical properties are given in Section 3.2. In Section 3.3, our numerical experience is reported. Finally, conclusions and possible extensions are provided in Section 3.4.

3.1 Sparsity in Optimal Randomized Classification Trees

3.1.1 Introduction

As in Chapter 2, we assume given a training sample $\{(\boldsymbol{x}_i, y_i)\}_{1\leq i\leq N}$, where $\boldsymbol{x}_i$ represents the p-dimensional vector of numerical predictor variables of individual i, and $y_i \in \{1,\dots,K\}$ indicates the class membership. Without loss of generality, we assume $\boldsymbol{x}_i \in [0,1]^p$, $i = 1,\dots,N$.

Sparse Optimal Randomized Classification Trees, addressed in this chapter, extend the Optimal Randomized Classification Trees (ORCTs) in Chapter 2, briefly detailed below. An ORCT is an optimal binary classification tree of a given depth D, obtained by minimizing the expected misclassification cost over the training sample. Figure 3.1 shows the structure of an

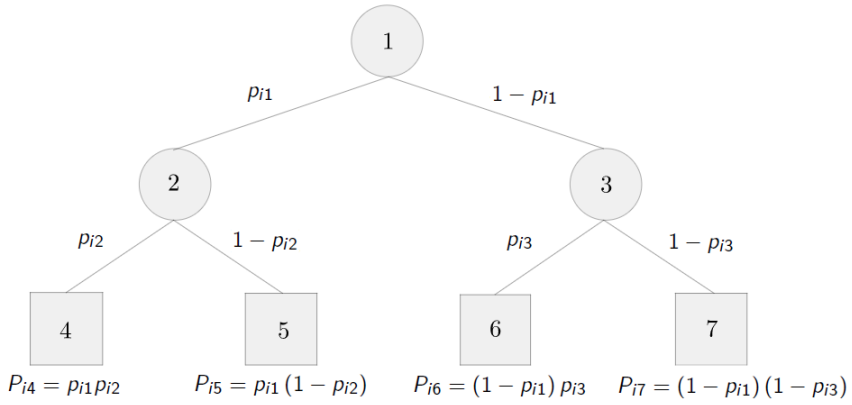


Figure 3.1: (Sparse) Optimal Randomized Classification Tree of depth $D = 2$.

ORCT of depth $D = 2$. Unlike classic decision trees, oblique cuts, on which more than one predictor variable takes part, are performed. ORCTs are modeled by means of a Non-Linear Continuous Optimization formulation. The usual deterministic yes/no rule at each branch node is replaced by a smoother rule: a probabilistic decision rule at each branch node, induced by a cumulative density function (CDF) F, is obtained. Therefore, the movements in ORCTs can be seen as randomized: at a given branch node of an ORCT, a random variable will be generated to indicate by which branch an individual has to continue. Since binary trees are built, the Bernoulli distribution is appropriate, whose probability of success will be determined by the value of this CDF, evaluated over the vector of predictor variables. More precisely, at a given branch node t of the tree, an individual with predictor variables $\boldsymbol{x}$ will go either to the left or to the right child nodes with probabilities $F\left(\frac{1}{p}\boldsymbol{a}_{\cdot t}^T\boldsymbol{x} - \mu_t\right)$ and $1 - F\left(\frac{1}{p}\boldsymbol{a}_{\cdot t}^T\boldsymbol{x} - \mu_t\right)$, respectively, where $\boldsymbol{a}_{\cdot t}$ and μ_t are decision variables. For further details on the construction of ORCTs, the reader is referred to Chapter 2. Sparse ORCT, S-ORCT, minimizes the expected misclassification cost over the training sample regularized with two polyhedral norms.

The following notation is needed:

Parameters

D	depth of the binary tree,
N	number of individuals in the training sample,
p	number of predictor variables,
K	number of classes,
$\{(\boldsymbol{x}_i, y_i)\}_{1\leq i\leq N}$	training sample, where $\boldsymbol{x}_i \in [0,1]^p$ and $y_i \in \{1,\ldots,K\}$,
I_k	set of individuals in the training sample belonging to class k, $k = 1,\ldots,K$,
$W_{y_i k}$	misclassification cost incurred when classifying an individual i, whose class is y_i, in class k, $y_i, i = 1,\ldots,N,\ k = 1,\ldots,K$,
$F(\cdot)$	the smooth CDF of a univariate continuous random variable centered at zero, used to define the probabilities for an individual to go to the left or the right child node in the tree. We will assume that f is the density of such a continuous random variable,
$\lambda^L \geq 0$	local sparsity regularization parameter,
$\lambda^G \geq 0$	global sparsity regularization parameter,

Nodes

τ_B	set of branch nodes,
τ_L	set of leaf nodes,
$N_L(t)$	set of ancestor nodes of leaf node t whose left branch takes part in the path from the root node to leaf node t, $t \in \tau_L$,
$N_R(t)$	set of ancestor nodes of leaf node t whose right branch takes part in the path from the root node to leaf node t, $t \in \tau_L$,

Decision variables

$a_{jt} \in [-1, 1]$ coefficient of predictor variable j in the oblique cut at branch node $t \in \tau_B$, with $\boldsymbol{a}$ being the $p \times |\tau_B|$ matrix of these coefficients, $\boldsymbol{a} = (a_{jt})_{j=1,\dots,p,\ t\in\tau_B}$. The expressions $\boldsymbol{a}_{j\cdot}$ and $\boldsymbol{a}_{\cdot t}$ will denote the j-th row and the t-th column of $\boldsymbol{a}$, respectively,

$\mu_t \in [-1, 1]$ location parameter at branch node $t \in \tau_B$, $\boldsymbol{\mu}$ being the vector that comprises every μ_t, i.e., $\boldsymbol{\mu} = (\mu_t)_{t\in\tau_B}$,

C_{kt} probability of being assigned to class label $k \in \{1, \dots, K\}$ for an individual at leaf node t, $t \in \tau_L$, being the $K \times |\tau_L|$ matrix such that $\boldsymbol{C} = (C_{kt})_{k=1,\dots,K,\ t\in\tau_L}$.

Probabilities

$p_{it}(\boldsymbol{a}_{\cdot t}, \mu_t)$ probability of individual i going down the left branch at branch node t. Its expression is $p_{it}(\boldsymbol{a}_{\cdot t}, \mu_t) = F\left(\frac{1}{p}\boldsymbol{a}_t^T\boldsymbol{x}_i - \mu_t\right),\ i = 1, \dots, N,\ t \in \tau_B$,

$P_{it}(\boldsymbol{a}, \boldsymbol{\mu})$ probability of individual i falling into leaf node t. Its expression is $P_{it}(\boldsymbol{a}, \boldsymbol{\mu}) = \prod_{t_l\in N_L(t)} p_{it_l}(\boldsymbol{a}_{\cdot t_l}, \mu_{t_l}) \prod_{t_r\in N_R(t)} (1 - p_{it_r}(\boldsymbol{a}_{\cdot t_r}, \mu_{t_r}))$, $i = 1, \dots, N,\ t \in \tau_L$,

$g(\boldsymbol{a}, \boldsymbol{\mu}, \boldsymbol{C})$ expected misclassification cost over the training sample. Its expression is $g(\boldsymbol{a}, \boldsymbol{\mu}, \boldsymbol{C}) = \frac{1}{N}\sum_{i=1}^{N}\sum_{t\in\tau_L} P_{it}(\boldsymbol{a}, \boldsymbol{\mu}) \sum_{k=1}^{K} W_{y_i k} C_{kt}$.

3.1.2 The formulation

With the previous parameters and decision variables, the S-ORCT is formulated as follows:

$$\min \quad g(\boldsymbol{a}, \boldsymbol{\mu}, \boldsymbol{C}) + \lambda^L \sum_{j=1}^{p} \|\boldsymbol{a}_{j\cdot}\|_1 + \lambda^G \sum_{j=1}^{p} \|\boldsymbol{a}_{j\cdot}\|_\infty \tag{3.1}$$

$$\text{s.t.} \quad \sum_{k=1}^{K} C_{kt} = 1,\ t \in \tau_L, \tag{3.2}$$

$$\sum_{t\in\tau_L} C_{kt} \geq 1,\ k = 1, \dots, K, \tag{3.3}$$

$$a_{jt} \in [-1, 1],\ j = 1, \dots, p,\ t \in \tau_B, \tag{3.4}$$

$$\mu_t \in [-1, 1],\ t \in \tau_B, \tag{3.5}$$

$$C_{kt} \in [0, 1],\ k = 1, \dots, K,\ t \in \tau_L. \tag{3.6}$$

In the objective function we have three terms, the first being the expected misclassification cost in the training sample, while the second and the third are regularization terms. The second term addresses local sparsity, since it penalizes the coefficients of the predictor variables used in the

cuts along the tree. Instead, the third term controls whether a given predictor variable is ever used across the whole tree, thus addressing global sparsity. The ℓ_∞-norm is used as a group penalty function, by forcing the coefficients linked to the same predictor variable to be shrunk simultaneously along all branch nodes. Note that both local and global sparsities are equivalent when dealing with depth $D = 1$, as there is a single cut across the whole tree.

In terms of the feasible region, for each leaf node $t \in \tau_L$, C_{kt} represents the probability that an individual at node t is assigned to class $k \in \{1, \ldots, K\}$. Constraints (3.2) force that such probabilities sum to 1, while constraints (3.3) force the sum of the probabilities along all leaf nodes $t \in \tau_B$ assigned to class k to be at least one.

Theorem 3.1 guarantees the existence of an optimal deterministic solution, i.e., such probabilities C_{kt} will all be in $\{0, 1\}$, and thus (3.6) can be replaced by

$$C_{kt} \in \{0, 1\}, \ k = 1, \ldots, K, \ t \in \tau_L. \tag{3.7}$$

Constraints (3.6) and (3.7) will be used interchangeably when needed.

Theorem 3.1. *There exists an optimal solution to Problem* (3.1)-(3.6) *such that* $C_{kt} \in \{0, 1\}, \ k = 1, \ldots, K, \ t \in \tau_L$.

Proof.

The continuity of the objective function (3.1), defined over a compact set, ensures the existence of an optimal solution of the optimization problem (3.1)-(3.6), by the Weierstrass Theorem. Let $\boldsymbol{a}^* = \left(a^*_{jt}\right)_{j=1,\ldots,p, \ t\in\tau_B}$, $\boldsymbol{\mu}^* = (\mu^*_t)_{t\in\tau_B}$, $\boldsymbol{C}^* = (C^*_{kt})_{k=1,\ldots,K, \ t\in\tau_B}$ be an optimal solution. Fixed $\boldsymbol{a}^*$, $\boldsymbol{\mu}^*$, then $\boldsymbol{C}^*$ is optimal to the following problem in the decision variables $C_{kt}, \ k = 1, \ldots, K, \ t \in \tau_L$:

$$\begin{aligned}
\min \quad & \frac{1}{N} \sum_{i=1}^{N} \sum_{t\in\tau_L} P_{it}\left(\boldsymbol{a}^*, \boldsymbol{\mu}^*\right) \sum_{k=1}^{K} W_{y_i k} C_{kt} + \lambda^L \sum_{j=1}^{p} \left\|\boldsymbol{a}^*_{j\cdot}\right\|_1 + \lambda^G \sum_{j=1}^{p} \left\|\boldsymbol{a}^*_{j\cdot}\right\|_\infty \\
\text{s.t.} \quad & \sum_{k=1}^{K} C_{kt} = 1, \ t \in \tau_L, \\
& \sum_{t\in\tau_L} C_{kt} \geq 1, \ k = 1, \ldots, K, \\
& C_{kt} \in [0, 1], \ k = 1, \ldots, K, \ t \in \tau_L.
\end{aligned}$$

This is a transportation problem, to which the integrality of an optimal solution is well-known to hold, i.e., there exists $\overline{\boldsymbol{C}} = \left(\overline{C}_{kt}\right)_{k=1,\ldots,K, \ t\in\tau_L} \in \{0, 1\}$ for all $k, \ t$ such that $\left(\boldsymbol{a}^*, \boldsymbol{\mu}^*, \overline{\boldsymbol{C}}\right)$ is also optimal for (3.1)-(3.6). □

Theorem 3.1 gives a new interpretation of constraints (3.2)-(3.3): if (3.7) is used instead of (3.6), when C_{kt} takes the value 1, then all the individuals at node $t \in \tau_L$ are labelled as k; and 0, otherwise. Constraints (3.2) state that any leaf node $t \in \tau_L$ must be labelled with exactly

one class label, and constraints (3.3) state that each class k has at least one node t with such label.

Once the optimization problem is solved, the S-ORCT predicts the class of a new unlabeled observation with predictor vector $\boldsymbol{x}$ with a probabilistic rule, namely, we estimate the probability of being in class k as $\sum_{t\in\tau_L} C_{kt}\cdot P_{\boldsymbol{x}t}\left(\boldsymbol{a},\boldsymbol{\mu}\right)$. If a deterministic classification rule is sought, we allocate to the most probable class. Moreover, if prior probabilities $\Pi_k\left(\boldsymbol{x}\right)$ are given, one can also use the Bayes rule.

ORCTs were also shown to deal effectively with controlling the correct classification rate on different classes in Chapter 2. This idea can also be applied to S-ORCTs. Hence, given the classes $k=1,\ldots,K$ to be controlled and their corresponding desired performances ρ_k, the expectation of achieving each performance guarantee can be computed with the ORCT parameters, provided that the following set of constrainsts is added to the model:

$$\sum_{i\in I_k}\sum_{t\in\tau_L} P_{it}\left(\boldsymbol{a},\boldsymbol{\mu}\right)C_{kt}\geq\rho_k|I_k|,\ k=1,\ldots,K. \tag{3.8}$$

With these constraints we have a direct control on the classification performance in each class separately. This is useful when dealing with imbalanced data sets.

3.1.3 A smooth reformulation

Problem (3.1)-(3.6) is non-smooth due to the norms $\|\cdot\|_1$ and $\|\cdot\|_\infty$ appearing in the objective function. A smooth version is easily obtained by rewritting both regularization terms using new decision variables. Since the first regularization term includes absolute values,

$$\|\boldsymbol{a}_{j\cdot}\|_1=\sum_{t\in\tau_B}|a_{jt}|\,,\ j=1,\ldots,p,$$

decision variables $a_{jt}\in[-1,1]\,,\ j=1,\ldots,p,\ t\in\tau_B$, are split into their positive and negative counterparts $a^+_{jt},\ a^-_{jt}\in[0,1]\,,\ j=1,\ldots,p,\ t\in\tau_B$, respectively, holding $a_{jt}=a^+_{jt}-a^-_{jt}$ and $|a_{jt}|=a^+_{jt}+a^-_{jt}$. Similarly, we denote $\boldsymbol{a}^+=\left(a^+_{jt}\right)_{j=1,\ldots,p,\ t\in\tau_B}$ and $\boldsymbol{a}^-=\left(a^-_{jt}\right)_{j=1,\ldots,p,\ t\in\tau_B}$. Regarding the second regularization term, new decision variables $\beta_j\in[0,1]$ are needed:

$$\|\boldsymbol{a}_{j\cdot}\|_\infty=\max_{t\in\tau_B}|a_{jt}|=\beta_j\in[0,1]\,,\ j=1,\ldots,p,$$

and have to force $\beta_j\geq|a_{jt}|=a^+_{jt}+a^-_{jt},\ j=1,\ldots,p,\ t\in\tau_B$.

We can now formulate S-ORCT as a smooth problem, thus solvable with standard continuous optimization solvers, as considered in our computational section. Indeed, we have that

problem (3.1)-(3.6) is equivalent to

$$\min \quad g\left(\boldsymbol{a}^{+}-\boldsymbol{a}^{-},\boldsymbol{\mu},\boldsymbol{C}\right)+\lambda^{L}\sum_{j=1}^{p}\sum_{t\in\tau_B}\left(a_{jt}^{+}+a_{jt}^{-}\right)+\lambda^{G}\sum_{j=1}^{p}\beta_j \tag{3.9}$$

$$\text{s.t.} \quad \sum_{k=1}^{K}C_{kt}=1,\ t\in\tau_L, \tag{3.10}$$

$$\sum_{t\in\tau_L}C_{kt}\geq 1,\ k=1,\ldots,K, \tag{3.11}$$

$$\beta_j\geq a_{jt}^{+}+a_{jt}^{-},\ j=1,\ldots,p, \tag{3.12}$$

$$a_{jt}^{+},\ a_{jt}^{-}\in[0,1],\ j=1,\ldots,p,\ t\in\tau_B, \tag{3.13}$$

$$\beta_j\in[0,1],\ j=1,\ldots,p, \tag{3.14}$$

$$\mu_t\in[-1,1],\ t\in\tau_B, \tag{3.15}$$

$$C_{kt}\in[0,1],\ k=1,\ldots,K,\ t\in\tau_L. \tag{3.16}$$

Observe that, if we are only concerned about global sparsity, and thus we set $\lambda^L=0$, the rewriting of the decision variables $a_{jt},\ j=1,\ldots,p,\ t\in\tau_B$ is no longer necessary and (3.4) replaces (3.13), and (3.12) turns into

$$\beta_j\geq a_{jt},\ j=1,\ldots,p,\ t\in\tau_B, \tag{3.17}$$

$$\beta_j\geq -a_{jt},\ j=1,\ldots,p,\ t\in\tau_B. \tag{3.18}$$

3.2 Theoretical properties

This section discusses some theoretical properties enjoyed by the S-ORCT. Let us consider the objective function of Problem (3.1)-(3.6). When taking λ^L and λ^G large enough, the first term related to the performance of the classifier becomes negligible and therefore $\boldsymbol{a}$ will shrink to $\mathbf{0}$. The tree with $\boldsymbol{a}=\mathbf{0}$ is the sparsest possible tree, though not the most promising one from an accuracy point of view, since none of the predictor variables are used to classify. In this case, the probability of an individual with predictor variables $\boldsymbol{x}$ being assigned to class k is independent of $\boldsymbol{x}$, and nothing more than the distribution of classes is available. In this section, we derive upper bounds for the sparsity parameters, λ^L and λ^G, in the sense that above these bounds the sparsest tree (with $\boldsymbol{a}^*=\mathbf{0}$) is a stationary point of the S-ORCT, that is, there exists $(\boldsymbol{a}^*=\mathbf{0},\boldsymbol{\mu}^*,\boldsymbol{C}^*)$ such that the necessary optimality condition with respect to $\boldsymbol{a}$ is satisfied. This is proven in Theorems 3.2 and 3.3.

Theorem 3.2. *Let $\sigma\in[0,1]$. For*

$$\lambda^L\geq(1-\sigma)\max_{\substack{\boldsymbol{\mu}\in[-1,1]^{|\tau_B|}\\ \boldsymbol{C}\in\{0,1\}^{K\times|\tau_L|}}}\ \max_{j=1,\ldots,p}\left\|\nabla_{\boldsymbol{a}_j}g\left(\mathbf{0},\boldsymbol{\mu},\boldsymbol{C}\right)\right\|_{\infty}\ \textit{and}$$

$$\lambda^G \geq \quad \sigma \max_{\substack{\boldsymbol{\mu}\in[-1,1]^{|\tau_B|} \\ \boldsymbol{C}\in\{0,1\}^{K\times|\tau_L|}}} \max_{j=1,\dots,p} \left\|\nabla_{\boldsymbol{a}_{j\cdot}} g\left(\mathbf{0},\boldsymbol{\mu},\boldsymbol{C}\right)\right\|_1,$$

$\boldsymbol{a}^* = \mathbf{0}$ *is a stationary point of the S-ORCT.*

Proof.

Let $\sigma,\ \lambda^L,\ \lambda^G$ be such that they satisfy the assumptions.

By Theorem 3.1, there exists $(\boldsymbol{a}^*, \boldsymbol{\mu}^*, \boldsymbol{C}^*)$ optimal solution to Problem (3.1)-(3.6) satisfying $C^*_{kt} \in \{0,1\}\ \forall k = 1,\dots,K,\ t \in \tau_L$. In the following, we will show that $(\mathbf{0}, \boldsymbol{\mu}^*, \boldsymbol{C}^*)$ is a stationary point of the S-ORCT, i.e.,

$$-\nabla_{\boldsymbol{a}} g\left(\mathbf{0},\boldsymbol{\mu}^*,\boldsymbol{C}^*\right) \in \partial_{\boldsymbol{a}} \left(\lambda^L \sum_{j=1}^{p} \left\|\boldsymbol{a}_{j\cdot}\right\|_1 + \lambda^G \sum_{j=1}^{p} \left\|\boldsymbol{a}_{j\cdot}\right\|_\infty\right)(\mathbf{0}) \tag{3.19}$$

where $\partial_{\boldsymbol{a}}$ is the subdifferential operator.

For every $\boldsymbol{a}_{j\cdot},\ j = 1,\dots,p$, we have that

$$\partial_{\boldsymbol{a}_{j\cdot}} \left(\left\|\boldsymbol{a}_{j\cdot}\right\|_1\right)(\mathbf{0}) = \mathbb{B}_\infty = \left\{\boldsymbol{q} \in \mathbb{R}^{|\tau_B|} : \left\|\boldsymbol{q}\right\|_\infty \leq 1\right\}$$
$$\partial_{\boldsymbol{a}_{j\cdot}} \left(\left\|\boldsymbol{a}_{j\cdot}\right\|_\infty\right)(\mathbf{0}) = \mathbb{B}_1 = \left\{\boldsymbol{q} \in \mathbb{R}^{|\tau_B|} : \left\|\boldsymbol{q}\right\|_1 \leq 1\right\}.$$

Hence,

$$-\nabla_{\boldsymbol{a}_{j\cdot}} g\left(\mathbf{0},\boldsymbol{\mu}^*,\boldsymbol{C}^*\right) \in \lambda^L \partial_{\boldsymbol{a}_{j\cdot}} \left(\left\|\boldsymbol{a}_{j\cdot}\right\|_1\right)(\mathbf{0}) + \lambda^G \partial_{\boldsymbol{a}_{j\cdot}} \left(\left\|\boldsymbol{a}_{j\cdot}\right\|_\infty\right)(\mathbf{0}),$$

if, and only if,

$$-\nabla_{\boldsymbol{a}_{j\cdot}} g\left(\mathbf{0},\boldsymbol{\mu}^*,\boldsymbol{C}^*\right) \in \lambda^L \mathbb{B}_\infty + \lambda^G \mathbb{B}_1,$$

if, and only if, there exist $\boldsymbol{q}_j^L,\ \boldsymbol{q}_j^G \in \mathbb{R}^{|\tau_B|}$ such that

$$\begin{aligned}
&\left\|\boldsymbol{q}_j^L\right\|_\infty \leq 1, \\
&\left\|\boldsymbol{q}_j^G\right\|_1 \leq 1, \\
&-\nabla_{\boldsymbol{a}_{j\cdot}} g\left(\mathbf{0},\boldsymbol{\mu}^*,\boldsymbol{C}^*\right) = \lambda^L \boldsymbol{q}_j^L + \lambda^G \boldsymbol{q}_j^G,
\end{aligned}$$

if, and only if, there exist $\tilde{\boldsymbol{q}}_j^L,\ \tilde{\boldsymbol{q}}_j^G \in \mathbb{R}^{|\tau_B|}$ such that

$$\begin{aligned}
&\left\|\tilde{\boldsymbol{q}}_j^L\right\|_\infty \leq \lambda^L, \\
&\left\|\tilde{\boldsymbol{q}}_j^G\right\|_1 \leq \lambda^G, \\
&-\nabla_{\boldsymbol{a}_{j\cdot}} g\left(\mathbf{0},\boldsymbol{\mu}^*,\boldsymbol{C}^*\right) = \tilde{\boldsymbol{q}}_j^L + \tilde{\boldsymbol{q}}_j^G.
\end{aligned}$$

Let us consider

$$\tilde{\boldsymbol{q}}_j^L = -\left(1-\sigma\right)\nabla_{\boldsymbol{a}_{j\cdot}} g\left(\mathbf{0},\boldsymbol{\mu}^*,\boldsymbol{C}^*\right),$$

$$\tilde{\boldsymbol{q}}_j^G = - \quad \sigma \quad \nabla_{\boldsymbol{a}_{j\cdot}} g\left(\mathbf{0}, \boldsymbol{\mu}^*, \boldsymbol{C}^*\right),$$

and check that the conditions are satisfied:

$$\left\|\tilde{\boldsymbol{q}}_j^L\right\|_\infty = (1-\sigma)\left\|\nabla_{\boldsymbol{a}_{j\cdot}} g\left(\mathbf{0}, \boldsymbol{\mu}^*, \boldsymbol{C}^*\right)\right\|_\infty \leq (1-\sigma) \max_{\substack{\boldsymbol{\mu}\in[-1,1]^{|\tau_B|} \\ \boldsymbol{C}\in\{0,1\}^{K\times|\tau_L|}}} \max_{j=1,\ldots,p} \left\|\nabla_{\boldsymbol{a}_{j\cdot}} g\left(\mathbf{0}, \boldsymbol{\mu}, \boldsymbol{C}\right)\right\|_\infty \leq \lambda^L,$$

$$\left\|\tilde{\boldsymbol{q}}_j^G\right\|_1 = \sigma\left\|\nabla_{\boldsymbol{a}_{j\cdot}} g\left(\mathbf{0}, \boldsymbol{\mu}^*, \boldsymbol{C}^*\right)\right\|_1 \leq \sigma \max_{\substack{\boldsymbol{\mu}\in[-1,1]^{|\tau_B|} \\ \boldsymbol{C}\in\{0,1\}^{K\times|\tau_L|}}} \max_{j=1,\ldots,p} \left\|\nabla_{\boldsymbol{a}_{j\cdot}} g\left(\mathbf{0}, \boldsymbol{\mu}, \boldsymbol{C}\right)\right\|_1 \leq \lambda^G,$$

$$\tilde{\boldsymbol{q}}_j^L + \tilde{\boldsymbol{q}}_j^G = -(1-\sigma)\nabla_{\boldsymbol{a}_{j\cdot}} g\left(\mathbf{0}, \boldsymbol{\mu}^*, \boldsymbol{C}^*\right) - \sigma\nabla_{\boldsymbol{a}_{j\cdot}} g\left(\mathbf{0}, \boldsymbol{\mu}^*, \boldsymbol{C}^*\right) = -\nabla_{\boldsymbol{a}_{j\cdot}} g\left(\mathbf{0}, \boldsymbol{\mu}^*, \boldsymbol{C}^*\right).$$

Therefore, the desired result follows. □

A stronger result is proven for the S-ORCT of depth $D = 1$ and $K = 2$. Since local and global sparsities are equivalent for the S-ORCT of depth $D = 1$, without loss of generality, we can assume that $\lambda^G = 0$. Therefore, the objective function of the S-ORCT of depth $D = 1$ can be written as:

$$g_1\left(\boldsymbol{a}_{\cdot 1}, \mu_1, \boldsymbol{C}\right) = g\left(\boldsymbol{a}_{\cdot 1}, \mu_1, \boldsymbol{C}\right) + \lambda^L \left\|\boldsymbol{a}_{\cdot 1}\right\|_1,$$

where

$$\begin{aligned} g\left(\boldsymbol{a}_{\cdot 1}, \mu_1, \boldsymbol{C}\right) &= \frac{1}{N}\sum_{i=1}^{N}\left[p_{i1}\left(\boldsymbol{a}_{\cdot 1}, \mu_1\right)\sum_{k=1}^{2} W_{y_i k} C_{k2} + \left(1 - p_{i1}\left(\boldsymbol{a}_{\cdot 1}, \mu_1\right)\right)\sum_{k=1}^{2} W_{y_i k} C_{k3}\right] \\ &= \frac{1}{N}\sum_{k=1}^{2}\sum_{i\in I_k}\left[p_{i1}\left(\boldsymbol{a}_{\cdot 1}, \mu_1\right)\sum_{k'\neq k} W_{kk'} C_{k'2} + \left(1 - p_{i1}\left(\boldsymbol{a}_{\cdot 1}, \mu_1\right)\right)\sum_{k'\neq k} W_{kk'} C_{k'3}\right] \end{aligned} \tag{3.20}$$

and

$$p_{i1}\left(\boldsymbol{a}_{\cdot 1}, \mu_1\right) = F\left(\frac{1}{p}\boldsymbol{a}_{\cdot 1}^T \boldsymbol{x}_i - \mu_1\right),\ i = 1, \ldots, N.$$

A technical lemma is needed to prove the desired result.

Lemma 3.1. *For any allocation rule $\boldsymbol{C}$, the objective function of the S-ORCT of depth $D = 1$, g_1, is monotonic in μ_1 when $\boldsymbol{a}_{\cdot 1} = \mathbf{0}$.*

Proof.

Fixed $\boldsymbol{a}_{\cdot 1} = (a_{j1})_{j=1,\ldots,p}$, and $\boldsymbol{C} = (C_{kt})_{k=1,2,\ t=2,3}$,

$$\left.\frac{\partial g_1}{\partial \mu_1}\right|_{\boldsymbol{a}_{\cdot 1}=\mathbf{0}} = \frac{1}{N}\sum_{k=1}^{K}\sum_{i\in I_k}\left(\sum_{k'\neq k} W_{kk'} C_{k'2} - \sum_{k'\neq k} W_{kk'} C_{k'3}\right)\left.\frac{\partial p_{i1}\left(\boldsymbol{a}_{\cdot 1}, \mu_1\right)}{\partial \mu_1}\right|_{\boldsymbol{a}_{\cdot 1}=\mathbf{0}},$$

where

$$\frac{\partial p_{i1}\left(\boldsymbol{a}_{\cdot 1},\mu_1\right)}{\partial \mu_1}=\frac{\partial F\left(\frac{1}{p}\boldsymbol{a}_{\cdot 1}^T\boldsymbol{x}_i-\mu_1\right)}{\partial\left(\frac{1}{p}\boldsymbol{a}_{\cdot 1}^T\boldsymbol{x}_i-\mu_1\right)}\frac{\partial\left(\frac{1}{p}\boldsymbol{a}_{\cdot 1}^T\boldsymbol{x}_i-\mu_1\right)}{\partial\mu_1}=-f\left(\frac{1}{p}\boldsymbol{a}_{\cdot 1}^T\boldsymbol{x}_i-\mu_1\right),\ i=1,\ldots,N,$$

and

$$\left.\frac{\partial p_{i1}\left(\boldsymbol{a}_{\cdot 1},\mu_1\right)}{\partial \mu_1}\right|_{\boldsymbol{a}_{\cdot 1}=\boldsymbol{0}}=-f\left(-\mu_1\right),\ i=1,\ldots,N.$$

Thus,

$$\begin{aligned}\left.\frac{\partial g_1\left(\boldsymbol{a}_{\cdot 1},\mu_1,\boldsymbol{C}\right)}{\partial \mu_1}\right|_{\boldsymbol{a}_{\cdot 1}=\boldsymbol{0}}&=\frac{1}{N}f\left(-\mu_1\right)\left(\sum_{i\in I_1}W_{12}\left(C_{23}-C_{22}\right)+\sum_{i\in I_2}W_{21}\left(C_{13}-C_{12}\right)\right)\\&=\frac{1}{N}f\left(-\mu_1\right)\left(W_{12}\left(C_{23}-C_{22}\right)|I_1|+W_{21}\left(1-C_{23}-1+C_{22}\right)|I_2|\right)\\&=\frac{1}{N}f\left(-\mu_1\right)\left(C_{23}-C_{22}\right)\left(W_{12}|I_1|-W_{21}|I_2|\right).\end{aligned}$$

Since f is a probability density function, the expression $\left.\frac{\partial g_1\left(\boldsymbol{a}_{\cdot 1},\mu_1,\boldsymbol{C}\right)}{\partial \mu_1}\right|_{\boldsymbol{a}_{\cdot 1}=\boldsymbol{0}}$ will always have the same sign for any value of μ_1 and the desired result follows. □

Theorem 3.3. *For*

$$\lambda^L\geq\frac{1}{N}\max_{j=1,\ldots,p}\left|-W_{21}\sum_{i\in I_2}x_{ij}+W_{12}\sum_{i\in I_1}x_{ij}\right|\max_{\mu_1\in\{-1,1\}}f\left(\mu_1\right),\tag{3.21}$$

$\boldsymbol{a}_{\cdot 1}^*=\boldsymbol{0}$ *is a stationary point of the S-ORCT of depth* $D=1$.

Proof.

Using the monotonicity of μ_1 proven in Lemma 3.1 and Theorem 3.2 with $\sigma=0$, we have that for

$$\begin{aligned}\lambda^L&\geq\max_{\substack{\mu_1\in\{-1,1\}\\ \boldsymbol{C}\in\{0,1\}^{2\times 2}}}\max_{j=1,\ldots,p}\left|\nabla_{\boldsymbol{a}_{j1}}g\left(\boldsymbol{0},\mu_1,\boldsymbol{C}\right)\right|\\&=\max_{\substack{\mu_1\in\{-1,1\}\\ \boldsymbol{C}\in\{0,1\}^{2\times 2}}}\left\|\nabla_{\boldsymbol{a}_{\cdot 1}}g\left(\boldsymbol{0},\mu_1,\boldsymbol{C}\right)\right\|_\infty,\end{aligned}\tag{3.22}$$

where g is as in (3.20), $\boldsymbol{a}_{\cdot 1}^*=\boldsymbol{0}$ is a stationary point of the S-ORCT. The remainder of the proof is devoted to rewriting (3.22) as in (3.21).

We proceed with the calculation of the gradient.

For $j = 1, \ldots, p$:

$$\frac{\partial g\left(\mathbf{0}, \mu_1, \boldsymbol{C}\right)}{\partial a_{j1}} = \left.\frac{\partial g\left(\boldsymbol{a}_{\cdot 1}, \mu_1, \boldsymbol{C}\right)}{\partial a_{j1}}\right|_{\boldsymbol{a}_{\cdot 1}=\mathbf{0}} = \frac{1}{N}\sum_{k=1}^{2}\sum_{i\in I_k}\left(\sum_{k'\neq k} W_{kk'}C_{k'2} - \sum_{k'\neq k} W_{kk'}C_{k'3}\right)\left.\frac{\partial p_{i1}\left(\boldsymbol{a}_{\cdot 1}, \mu_1\right)}{\partial a_{j1}}\right|_{\boldsymbol{a}_{\cdot 1}=\mathbf{0}},$$

where

$$\frac{\partial p_{i1}\left(\boldsymbol{a}_{\cdot 1}, \mu_1\right)}{\partial a_{j1}} = \frac{\partial F\left(\frac{1}{p}\boldsymbol{a}_1^T\boldsymbol{x}_i - \mu_1\right)}{\partial\left(\frac{1}{p}\boldsymbol{a}_1^T\boldsymbol{x}_i - \mu_1\right)}\frac{\partial\left(\frac{1}{p}\boldsymbol{a}_1^T\boldsymbol{x}_i - \mu_1\right)}{\partial a_{j1}} = \frac{x_{ij}}{p}f\left(\frac{1}{p}\boldsymbol{a}_1^T\boldsymbol{x}_i - \mu_1\right),\ i = 1, \ldots, N.$$

and

$$\left.\frac{\partial p_{i1}\left(\boldsymbol{a}_{\cdot 1}, \mu_1\right)}{\partial a_{j1}}\right|_{\boldsymbol{a}_{\cdot 1}=\mathbf{0}} = \frac{x_{ij}}{p}f\left(-\mu_1\right),\ i = 1, \ldots, N.$$

Thus,

$$\frac{\partial g\left(\mathbf{0}, \mu_1, \boldsymbol{C}\right)}{\partial a_{j1}} = \frac{1}{Np}f\left(-\mu_1\right)\left(W_{12}\sum_{i\in I_1} x_{ij}\left(C_{22} - C_{23}\right) + W_{21}\sum_{i\in I_2} x_{ij}\left(C_{12} - C_{13}\right)\right).$$

Now, we look for the maximum λ^L among every possible allocation of the decision variables $\boldsymbol{C}$, i.e.:

$$\lambda^L_{\mu_1} = \max_{\boldsymbol{C}\in\{0,1\}^{2\times 2}} \left\|\nabla_{\boldsymbol{a}_{\cdot 1}} g\left(\mathbf{0}, \mu_1, \boldsymbol{C}\right)\right\|_\infty = \max_{\overline{\boldsymbol{C}}\in\{0,1\}^{4\times 1}} \left\|D\overline{\boldsymbol{C}}\right\|_\infty,$$

where

$$D = \frac{1}{Np}f\left(-\mu_1\right)\begin{pmatrix} -W_{21}\sum_{i\in I_2} x_{i1} & W_{21}\sum_{i\in I_2} x_{i1} & -W_{12}\sum_{i\in I_1} x_{i1} & W_{12}\sum_{i\in I_1} x_{i1} \\ \vdots & \vdots & \vdots & \vdots \\ -W_{21}\sum_{i\in I_2} x_{ip} & W_{21}\sum_{i\in I_2} x_{ip} & -W_{12}\sum_{i\in I_1} x_{ip} & W_{12}\sum_{i\in I_1} x_{ip} \end{pmatrix}$$

and $\overline{\boldsymbol{C}} = \left(C_{12}, C_{13}, C_{22}, C_{23}\right)^T$.

$$\begin{aligned}
\max_{\overline{\boldsymbol{C}}\in\{0,1\}^{4\times 1}} \left\|D\overline{\boldsymbol{C}}\right\|_\infty &= \max_{\overline{\boldsymbol{C}}\in\{0,1\}^{4\times 1}}\ \max\ \left\{\left|d_1^T\ \overline{\boldsymbol{C}}\right|, \ldots, \left|d_p^T\ \overline{\boldsymbol{C}}\right|\right\} \\
&= \max_{\overline{\boldsymbol{C}}\in\{0,1\}^{4\times 1}}\ \max\ \left\{d_1^T\ \overline{\boldsymbol{C}}, -d_1^T\ \overline{\boldsymbol{C}}, \ldots, d_p^T\ \overline{\boldsymbol{C}}, -d_p^T\ \overline{\boldsymbol{C}}\right\} \\
&= \max\left\{\max_{\overline{\boldsymbol{C}}\in\{0,1\}^{4\times 1}}\ d_1^T\ \overline{\boldsymbol{C}}, \max_{\overline{\boldsymbol{C}}\in\{0,1\}^{4\times 1}}\ -d_1^T\ \overline{\boldsymbol{C}}, \ldots, \max_{\overline{\boldsymbol{C}}\in\{0,1\}^{4\times 1}}\ d_p^T\ \overline{\boldsymbol{C}}, \max_{\overline{\boldsymbol{C}}\in\{0,1\}^{4\times 1}}\ -d_p^T\ \overline{\boldsymbol{C}}\right\}.
\end{aligned}$$

A finite number of transportation problems is to be solved, with the form:

$$z = \max_{\overline{\boldsymbol{C}}\in\{0,1\}^{4\times 1}}\ \left\{\pm d_j^T\ \overline{\boldsymbol{C}}\right\}$$

$$\begin{aligned} \text{s.t.} \quad & C_{12} + C_{22} = 1 \\ & C_{13} + C_{23} = 1 \\ & C_{12} + C_{13} \geq 1 \\ & C_{22} + C_{23} \geq 1, \end{aligned}$$

for which the integrality property holds. Then, we only have as possible solutions: $\overline{\boldsymbol{C}} = (1,0,0,1)^T$ or $\overline{\boldsymbol{C}} = (0,1,1,0)^T$. Thus, the optimal objective is obtained as follows:

$$\begin{aligned} z_{opt} &= \max\left\{ \pm d_j^T \ \overline{\boldsymbol{C}}|_{\overline{\boldsymbol{C}}=(1,0,0,1)^T}, \pm d_j^T \ \overline{\boldsymbol{C}}|_{\overline{\boldsymbol{C}}=(0,1,1,0)^T} \right\} \\ &= \max\left\{ \frac{1}{Np} f(-\mu_1) \left(-W_{21} \sum_{i \in I_2} x_{ij} + W_{12} \sum_{i \in I_1} x_{ij} \right), \frac{1}{Np} f(-\mu_1) \left(W_{21} \sum_{i \in I_2} x_{ij} - W_{12} \sum_{i \in I_1} x_{ij} \right) \right\} \\ &= \frac{1}{Np} f(-\mu_1) \left| -W_{21} \sum_{i \in I_2} x_{ij} + W_{12} \sum_{i \in I_1} x_{ij} \right|. \end{aligned}$$

Let us define

$$\lambda_{\mu_1}^L = \frac{1}{Np} f(-\mu_1) \max_{j=1,\ldots,p} \left| -W_{21} \sum_{i \in I_2} x_{ij} + W_{12} \sum_{i \in I_1} x_{ij} \right|,$$

and the result holds when

$$\lambda^L \geq \max\left\{ \lambda_{\mu_1=-1}^L, \lambda_{\mu_1=1}^L \right\}.$$

□

3.3 Computational experiments

3.3.1 Introduction

The aim of this section is to illustrate the performance of our sparse optimal randomized classification trees S-ORCTs. We have run our model for a grid of values of the sparsity regularization parameters λ^L and λ^G. The message that can be drawn from our experimental experience is twofold. First, we show empirically that our S-ORCT can gain in both local and global sparsity, without harming classification accuracy. Second, we benchmark our approach against CART, the classic approach to build decision trees, which considers orthogonal cuts and therefore has the best possible local sparsity. We show that we are able to trade in some of our classification accuracy, whilst still being superior to CART, in order to be comparable to CART in terms of global sparsity.

The S-ORCT smooth formulation (3.9)-(3.16) has been implemented using Pyomo opti-

mization modeling language [Hart et al., 2017, 2011] in Python 3.5 [Python Core Team, 2015]. As solver, we have used IPOPT 3.11.1 [Wächter and Biegler, 2006], and have followed a multistart approach, where the process is repeated 20 times starting from different random initial solutions. For CART, the implementation in the `rpart` R package [Therneau et al., 2015] is used. Our experiments have been conducted on a PC, with an Intel® Core™ i7-2600 CPU 3.40GHz processor and 16 GB RAM. The operating system is 64 bits.

The remainder of the section is structured as follows. Section 3.3.2 gives details on the procedure followed to test S-ORCT. In Sections 3.3.3 and 3.3.4, respectively, we discuss the results for local and global sparsity separately, while in Section 3.3.5 we present results when both sparsities are simultaneously taken into account. Finally, Section 3.3.6 statistically compares S-ORCT versus CART in terms of classification accuracy and global sparsity.

3.3.2 Setup

An assorted collection of well-known real data sets from the UCI Machine Learning Repository [Lichman, 2013] has been chosen for the computational experiments. Table 3.1 lists their names together with their number of observations, number of predictor variables and number of classes with the corresponding class distribution. In our pursuit of building small and, therefore, less complex trees, the construction of S-ORCTs has been restricted to depth $D = 1$ for two-class problems and depth $D = 2$ for three- and four- class problems.

Table 3.1: Information about the data sets considered.

Data set	Abbrev.	N	p	K	Class distribution
Monks-problems-3	Monks-3	122	11	2	51% - 49%
Monks-problems-1	Monks-1	124	11	2	50% - 50%
Monks-problems-2	Monks-2	169	11	2	62% - 38%
Connectionist-bench-sonar	Sonar	208	60	2	55% - 45%
Ionosphere	Ionosphere	351	34	2	64% - 36%
Breast-cancer-Wisconsin	Wisconsin	569	30	2	63% - 37%
Credit-approval	Creditapproval	653	37	2	55% - 45%
Pima-indians-diabetes	Pima	768	8	2	65% - 35%
Statlog-project-German-credit	Germancredit	1000	48	2	70% - 30%
Banknote-authentification	Banknote	1372	4	2	56% - 44%
Ozone-level-detection-one	Ozone	1848	72	2	97% - 3%
Spambase	Spam	4601	57	2	61% - 39%
Iris	Iris	150	4	3	33.3%-33.3%-33.3%
Wine	Wine	178	13	3	40%-33%-27%
Seeds	Seeds	210	7	3	33.3%-33.3%-33.3%
Balance-scale	Balance	625	16	3	46%-46%-8%
Thyroid-disease-ann-thyroid	Thyroid	3772	21	3	92.5%-5%-2.5%
Car-evaluation	Car	1728	15	4	70%-22%-4%-4%

Each data set has been split into two subsets: the training subset (75%) and the test subset

(25%). The corresponding S-ORCT is built on the training subset and, then, accuracy, local and global sparsity are measured. The out-of-sample accuracy over the test subset is denoted by *acc*. Local sparsity is denoted by δ^L and reads as the average percentage of predictor variables not used per branch node:

$$\delta^L = \frac{1}{|\tau_B|} \sum_{t \in \tau_B} \frac{|\{a_{jt} = 0,\ j = 1, \ldots, p\}|}{p} \times 100.$$

Global sparsity, δ^G, is measured as the percentage of predictor variables not used at any of the branch nodes, i.e., across the whole tree:

$$\delta^G = \frac{|\{\boldsymbol{a}_{j\cdot} = \mathbf{0},\ j = 1, \ldots, p\}|}{p} \times 100.$$

Note that when $D = 1$, local and global sparsity are the same, since there is a single cut across the whole tree. The training/testing procedure has been repeated ten times in order to avoid the effect of the initial split of the data. The results shown in the tables represent the average of such ten runs to each of the three performance criteria.

In what follows, we describe the choices made for the parameters in S-ORCT. Equal misclassification weights, $W_{y_i k} = 0.5,\ k = 1, \ldots, K,\ k \neq y_i$, have been used for the experiments. We have added the set of constraints (3.8) with $\rho_k = 0.1,\ k = 1, \ldots, K$. The logistic CDF has been chosen for our experiments:

$$F(\cdot) = \frac{1}{1 + \exp(-(\cdot)\gamma)},$$

with a large value of γ, namely, $\gamma = 512$. The larger the value of γ, the closer the decision rule defined by F is to a deterministic rule. We will illustrate that a small level of randomization is enough for obtaining good results. We have trained S-ORCT, as formulated in (3.9)-(3.16), for 17×17 pairs of values for $\left(\lambda^L, \lambda^G\right)$ starting from $\lambda^L = 0$ followed by the grid $\left\{\frac{2^r}{p\,|\tau_B|},\ -12 \leq r \leq 3,\ r \in \mathbb{Z}\right\}$, and, similarly, $\lambda^G = 0$ followed by the grid $\left\{\frac{2^r}{p},\ -12 \leq r \leq 3,\ r \in \mathbb{Z}\right\}$. We start solving the optimization problem with $\left(\lambda^L, \lambda^G\right) = (0, 0)$, where the multistart approach uses 20 random initial solutions. We continue solving the optimization problem for $\lambda^L = 0$ but with larger values of λ^G. Once all values of λ^G are executed, we start the process all over again with the next value of λ^L in the grid. For pair $\left(\lambda^L, \lambda^G\right)$, we feed the corresponding optimization problem with the 20 solutions resulting from the problem solved for the previous pair. For a given initial solution, the computing time taken by the S-ORCT typically ranges from 0.33 seconds (in Monks-1) to 22.27 seconds (in Thyroid).

For CART, the default parameter setting in `rpart` is used.

3.3.3 Results for local sparsity

Tables 3.2 and 3.3 present the results of the so-called local S-ORCT, i.e., when $\lambda^G = 0$ and thus only local sparsity is taken into account. Figures 3.2 and 3.3 depict these results per data set, by showing simultaneously δ^L (blue solid line) and *acc* (red dashed line) as a function of the grid of the λ^L's considered. As expected, the larger the λ^L, the larger the δ^L. The sparsest tree is shown in most of the data sets for large values of the parameter λ^L, where the best solution in terms of sparsity is obtained but the worst possible one in terms of accuracy. In terms of accuracy, the best rates are sometimes achieved when not all the predictor variables are included in the model. For instance, best performance is reached when sparsity is about $9-25\%$ for Pima, 30% for Monks-1, 32% for Monks-2, 44% for Germancredit, 47% for Car, $52-56\%$ for Thyroid, 54% for Monks-3, $55-60\%$ for Iris, $72-90\%$ for Sonar, 81% for both Wine and Seeds and 87% for Ionosphere. We highlight the Creditapproval data set, on which one single predictor variable can already guarantee very good accuracy. For Ozone, accuracy remains over 96% for the grid of λ^L's considered. Accuracy might be slightly damaged but a great gain in sparsity is obtained. This is the case for Banknote, Spam, Balance or Wisconsin, which present a loss of accuracy lower than the 1 percentage point (p.p.), 4 p.p., 6 p.p. and 1 p.p. but 25%, 52%, 63% and 85% of local sparsity is reached, respectively.

3.3.4 Results for global sparsity

This section is devoted to the global S-ORCT, i.e., when $\lambda^L = 0$ and thus only global sparsity is taken into account. We focus on depth $D = 2$, since for $D = 1$ global sparsity is equal to local sparsity. Similarly to Subsection 3.3.3, Table 3.4 presents the results of the global S-ORCT, while Figure 3.4 visualizes these results by showing simultaneously, per data set, δ^G (blue solid line) and *acc* (red dashed line) as a function of the grid of the λ^G's considered. As for local sparsity, as λ^G grows, δ^G increases. For Iris and Seeds, a similar classification accuracy to that with all of the predictor variables is obtained while removing 75% and 29% of them, respectively. For Wine, the best rates of accuracy are obtained with $15\%-23\%$ of global sparsity. A loss of less than 10 p.p. of accuracy is observed for Balance but 25% of predictor variables are not being used, respectively. Car remains around the accuracy rate of 80% while using half of the predictor variables. Thyroid, an imbalanced data set, is over 90% of accuracy for the whole grid of λ^G's considered.

3.3.5 Results for local and global sparsity

In this section, results enforcing local and global sparsity are presented by means of heatmaps, as seen in Figure 3.5. The experiment has been conducted on data sets of $K = 3$ and 4 classes, for which S-ORCTs of depth $D = 2$ are built. For each dataset, three heatmaps are depicted as a function of the grid of the sparsity regularization parameters, λ^L and λ^G: the average out-of-sample accuracy, *acc*, and the local and global sparsity, δ^L and δ^G, respectively, obtained

Table 3.2: Results for the local S-ORCT of depth $D = 1$ as a function of λ^L, where δ^L represents the average percentage of predictor variables not used per branch node in the tree over the ten runs and *acc*, the average out-of-sample accuracy.

λ^L	Monks-3		Monks-1		Monks-2		Sonar		Ionosphere		Wisconsin		Creditapproval		Pima		Germancredit		Banknote		Ozone		Spam	
	δ^L	acc	δ^L	acc	δ^L	acc	δ^L	acc	δ^L	acc	δ^L	acc	δ^L	acc	δ^L	acc	δ^L	acc	δ^L	acc	δ^L	acc	δ^L	acc
0	0	89.7	1	77.7	3	74.3	0	75.8	0	84.1	0	96.2	1	84.1	0	75.8	0	73.5	0	99.0	0	96.5	0	89.8
2^{-12}	1	91.0	21	80.6	28	77.1	0	75.8	4	84.2	1	96.4	9	84.0	0	75.8	0	72.8	0	99.0	4	96.6	0	89.8
2^{-11}	0	91.0	21	79.0	28	77.1	0	77.5	3	84.7	4	96.1	9	83.7	0	75.6	0	72.9	0	99.0	10	96.5	0	89.8
2^{-10}	0	90.0	28	80.0	28	77.1	1	77.5	4	84.5	7	96.0	11	83.9	0	75.6	1	73.3	0	99.1	18	96.4	1	89.8
2^{-9}	0	89.3	27	82.9	28	77.1	2	77.5	4	84.4	10	96.1	13	83.7	0	76	1	73.2	3	99.1	29	96.5	2	89.8
2^{-8}	2	90.0	30	81.6	28	77.1	2	77.7	4	85.2	16	96.3	15	84.2	0	75.7	1	73.2	3	99.1	44	96.4	3	89.8
2^{-7}	0	90.7	23	78.4	28	77.1	5	76.9	8	84.6	28	96.3	16	84.2	0	75.9	2	73.4	3	99.0	62	96.4	5	89.6
2^{-6}	7	90.3	34	80.6	28	77.1	9	77.1	10	85.3	39	96.3	20	84.1	1	75.8	3	73.8	3	98.7	78	96.6	9	89.4
2^{-5}	2	90.3	32	78.4	28	77.1	18	75.4	19	85.9	50	96.3	29	84.6	9	76.2	0	74.2	23	98.5	83	96.6	25	88.8
2^{-4}	3	92.0	29	81.3	28	77.1	28	76.3	32	86.3	59	96.5	44	85.1	20	76.1	31	73.9	15	98.3	87	96.6	44	88.5
2^{-3}	15	92.7	30	83.5	28	77.1	40	77.1	49	86.2	67	96.3	62	86.1	25	75.9	37	73.4	10	98.0	90	96.7	52	86.1
2^{-2}	45	94.3	38	81.6	28	77.1	56	76.9	57	86.1	74	95.8	75	85.4	44	75.3	50	73.8	25	98.0	92	96.7	71	83
2^{-1}	54	94.7	39	81.0	32	78.6	72	78.6	74	85.6	85	95.7	95	86.3	61	74.9	69	71.8	25	97.5	95	96.7	82	78.6
2^{0}	54	94.7	62	81.0	39	76.7	85	78.1	87	86.8	87	94.7	97	86.7	81	73.7	93	69.6	25	96.7	96	96.7	97	64.4
2^{1}	54	94.7	71	78.4	95	63.3	90	78.3	91	84.7	91	92.7	97	86.7	94	65.8	98	69.5	50	85.8	97	96.7	100	60.4
2^{2}	77	74.3	84	72.6	100	64.3	98	62.9	94	75.1	95	91.2	97	86.7	100	63.4	100	69.5	50	84.0	100	96.7	100	60.4
2^{3}	93	55.7	91	72.2	100	64.3	100	51.5	100	61.1	99	64.1	97	86.7	100	63.4	100	69.5	100	56.3	100	96.7	100	60.4

over the ten runs performed. The color bar of each heatmap goes from light green to dark blue, being the latter the maximum accuracy, local sparsity or global sparsity achieved, respectively. As a general behavior, the best rates of accuracy are not always achieved only for $\left(\lambda^L, \lambda^G\right) = (0, 0)$, but also for other pairs of the chosen grid, i.e., the data set remains equally well explained while needing less information. As before, according to local sparsity, for a fixed λ^G, δ^L has a growing trend. A similar behavior is observed for δ^G when λ^L is fixed. It is also worth mentioning that small changes of λ^L quickly lead to a gain in δ^L. Nevertheless, as expected, the gain in δ^G is slower for the same range in λ^G.

Table 3.3: Results for the local S-ORCT of depth $D = 2$ as a function of λ^L, where δ^L represents the average percentage of predictor variables not used per branch node in the tree over the ten runs and *acc*, the average out-of-sample accuracy.

λ^L	Iris		Wine		Seeds		Balance		Thyroid		Car	
	δ^L	acc	δ^L	acc	δ^L	acc	δ^L	acc	δ^L	acc	δ^L	acc
0	8	95.9	15	96.6	10	94.4	33	96.6	57	92.8	20	92.7
2^{-12}	42	95.9	51	98.6	33	93.8	58	92.0	61	92.7	36	91.5
2^{-11}	42	95.9	54	98.4	38	93.8	60	91.1	59	92.9	33	91.9
2^{-10}	42	96.2	54	97.3	38	94.0	65	91.0	64	92.6	36	91.5
2^{-9}	42	95.9	56	97.5	43	93.8	67	91.2	62	92.7	36	91.4
2^{-8}	42	95.9	56	96.8	48	93.2	60	91.9	65	92.5	36	91.4
2^{-7}	42	95.9	59	96.8	48	91.3	60	91.7	70	92.1	36	91.3
2^{-6}	42	95.9	59	96.8	52	94.0	65	92.2	72	92.1	38	91.6
2^{-5}	42	95.4	59	96.8	52	94.4	58	92.6	74	92.2	40	91.3
2^{-4}	42	95.9	59	97.3	57	93.8	58	92.4	79	92.2	42	91.1
2^{-3}	42	93.2	62	97.5	67	94.6	63	91.1	83	92.1	40	91.7
2^{-2}	50	89.7	62	97.7	67	94.4	65	90.6	87	92.3	47	90.4
2^{-1}	50	92.7	64	98.2	71	93.6	67	89.2	90	92.0	51	90.2
2^{0}	58	90.0	69	96.8	76	93.6	71	88.1	91	91.9	64	87.6
2^{1}	67	90.5	77	95.2	81	90.2	75	87.2	92	92.0	71	85.4
2^{2}	75	91.1	82	89.5	81	88.5	77	82.6	95	91.8	80	80.8
2^{3}	83	88.6	90	76.4	91	73.6	83	77.3	100	92.2	91	68.2

3.3.6 Comparison S-ORCT versus CART

A statistical comparison between the proposed S-ORCT and CART, the classic approach to build decision trees, is provided in this section. CARTs, as many other approaches that implement orthogonal cuts [Bertsimas and Dunn, 2017; Firat et al., 2019; Günlük et al., 2021], are leaders in terms of local sparsity. Thus, the comparison S-ORCT versus CART is performed in terms of accuracy and global sparsity. Tables 3.2 and 3.4 for S-ORCT have been considered for the experiment.

CART has been trained and tested over the same ten runs as S-ORCT. For each pair S-ORCT$\left(\lambda^G\right)$ versus CART, two hypothesis tests for the equality of means of paired samples

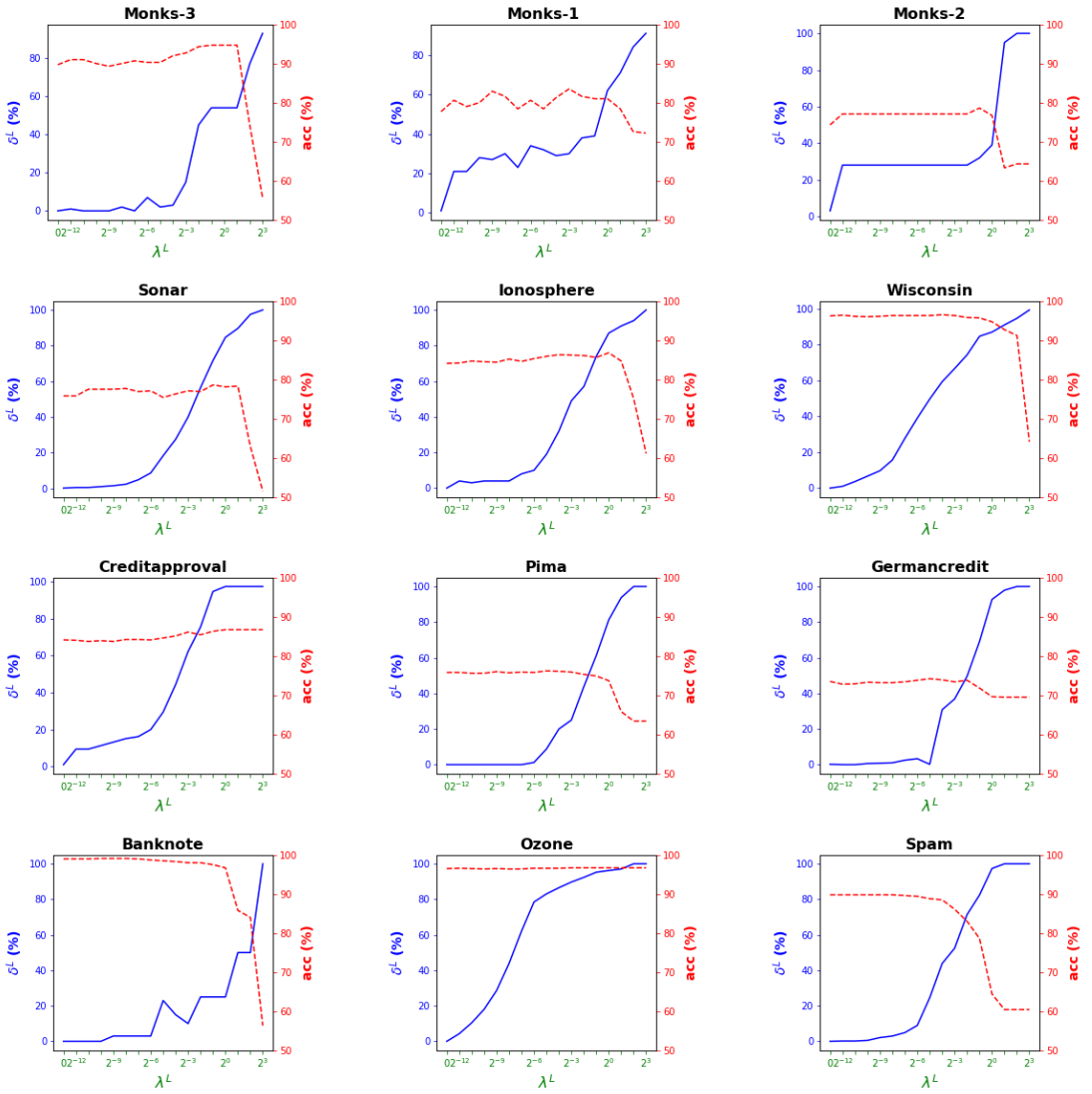


Figure 3.2: Graphical representation, for each data set, of the average percentage of predictor variables per branch node, δ^L, together with the average out-of-sample accuracy obtained, *acc*, as a function of the values of λ^L considered in the local S-ORCT construction.

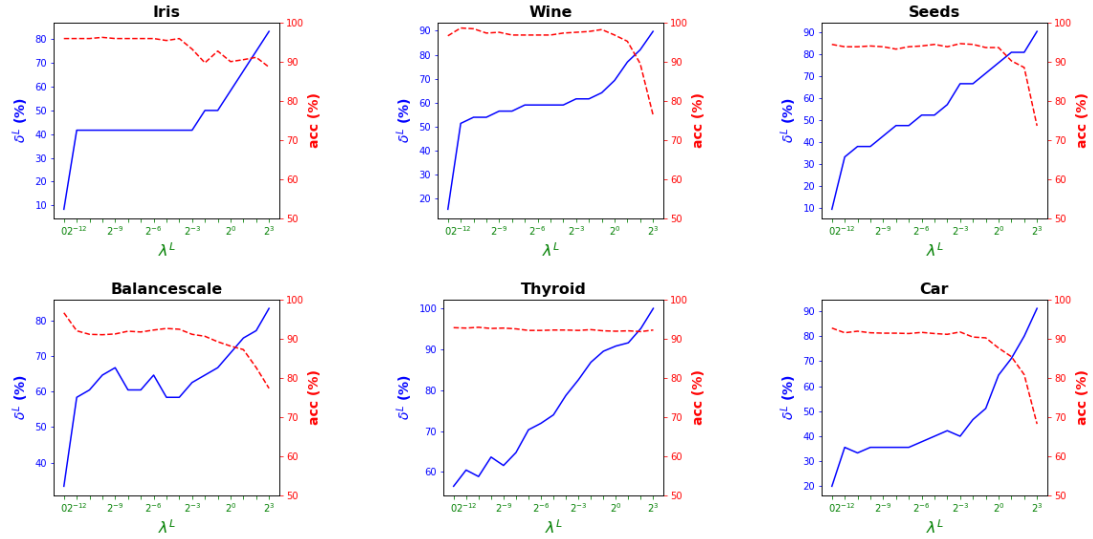


Figure 3.3: Graphical representation, for each data set, of the average percentage of predictor variables per branch node, δ^L, together with the average out-of-sample accuracy obtained, *acc*, as a function of the values of λ^L considered in the local S-ORCT construction.

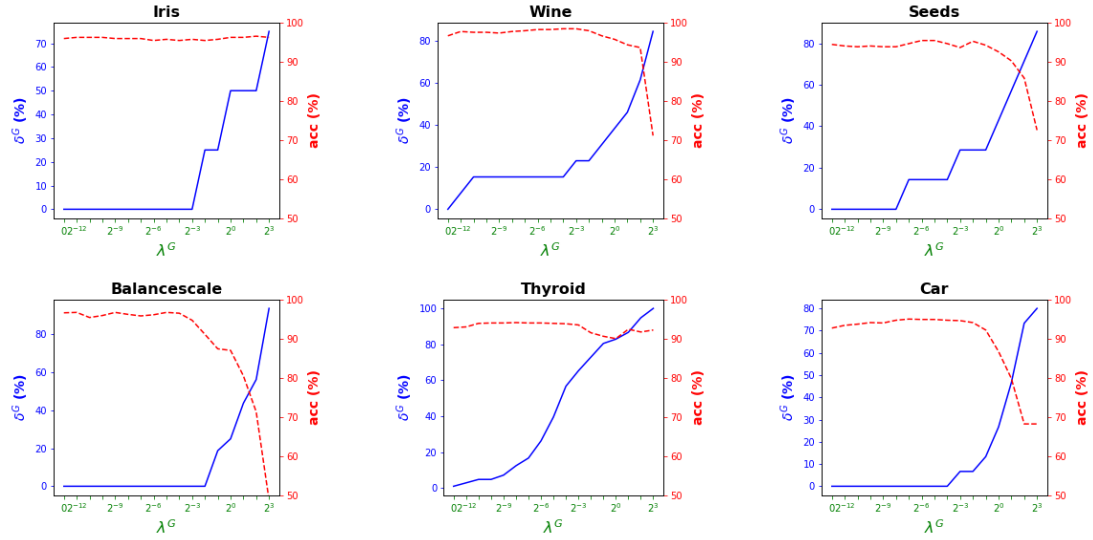


Figure 3.4: Graphical representation, for each data set, of the average percentage of predictor variables per tree, δ^G, together with the average out-of-sample accuracy obtained, *acc*, as a function of the values of λ^G considered in the global S-ORCT construction.

Table 3.4: Results for the global S-ORCT of depth $D = 2$ as a function of λ^G, where δ^G represents the average percentage of predictor variables not used per tree over ten runs and *acc*, the average out-of-sample accuracy.

λ^G	Iris		Wine		Seeds		Balance		Thyroid		Car	
	δ^G	acc	δ^G	acc	δ^G	acc	δ^G	acc	δ^G	acc	δ^G	acc
0	0	95.9	0	96.6	0	94.4	0	96.6	1	92.8	0	92.7
2^{-12}	0	96.2	18	97.7	0	94.0	0	96.7	3	93.0	0	93.4
2^{-11}	0	96.2	15	97.5	0	93.8	0	95.4	5	93.9	0	93.7
2^{-10}	0	96.2	15	97.5	0	94.0	0	95.9	5	93.9	0	94.1
2^{-9}	0	95.9	15	97.3	0	93.8	0	96.7	7	94.0	0	94.0
2^{-8}	0	95.9	15	97.7	0	93.8	0	96.2	12	94.1	0	94.7
2^{-7}	0	95.9	15	97.9	14	94.6	0	95.8	17	94.0	0	95.0
2^{-6}	0	95.4	15	98.2	14	95.4	0	96.1	26	94.0	0	94.9
2^{-5}	2	95.7	15	98.2	14	95.4	0	96.7	40	93.9	0	94.9
2^{-4}	0	95.4	15	98.4	14	94.6	0	96.5	57	93.8	0	94.7
2^{-3}	0	95.7	23	98.4	29	93.6	0	94.7	65	93.5	7	94.6
2^{-2}	25	95.4	23	97.9	29	95.2	0	91.1	73	91.5	7	94.1
2^{-1}	25	95.7	31	96.6	29	94.2	19	87.4	81	90.6	13	92.2
2^{0}	50	96.2	39	95.7	43	92.5	25	87.0	83	90.0	27	86.7
2^{1}	50	96.2	46	94.3	57	90.2	44	80.5	87	92.4	47	79.8
2^{2}	50	96.5	62	93.6	71	85.8	56	71.3	95	91.7	73	68.2
2^{3}	75	96.2	85	71.1	86	72.5	94	48.8	100	92.2	80	68.2

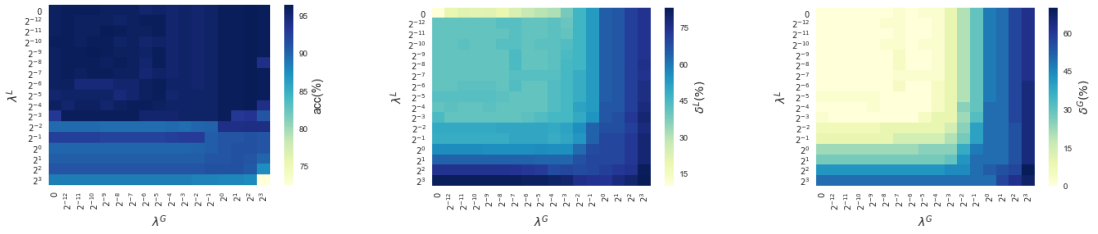
were carried out, one for accuracy and another for global sparsity, assuming normality, at a 5% significance level. For this task, the `t.test` function in R has been used. Figure 3.6 depicts, for each data set, the resulting confidence intervals (blue solid line) at the 95% confidence level for the difference in average accuracy (on the left) and global sparsity (on the right) between S-ORCT$\left(\lambda^G\right)$ and CART. The red dashed horizontal line represents the null hypothesis in each case. Except for Creditapproval and Thyroid, for the smaller values of λ^G, our approach is significantly better than, or at least comparable to our approach, CART in terms of accuracy, while CART is significantly better than, or at least comparable to, in terms of global sparsity. For the larger values of λ^G, our approach starts to be comparable and then dominate CART in terms of global sparsity at the cost of accuracy.

3.4 Conclusions

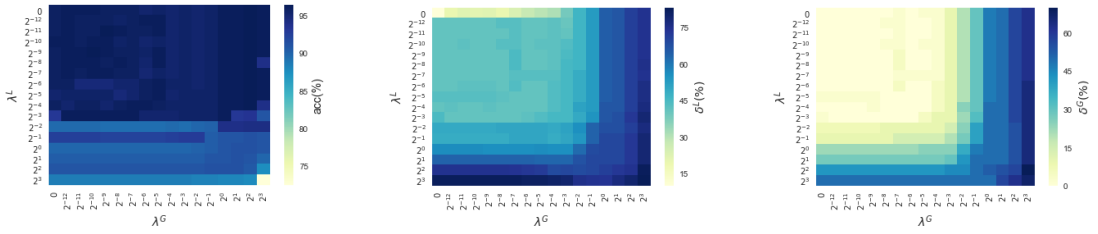
In this chapter, we have extended the methodology in Chapter 2 and proposed the Sparse Optimal Randomized Classification Tree (S-ORCT), in which a compromise between good classification accuracy and sparsity is pursued. Local and global sparsity in the tree are modeled by including in the objective function regularizations, namely, ℓ_1 and ℓ_∞, respectively. Our numerical results illustrate that our approach can improve both sparsities without harming classification accuracy. Unlike CART, we are able to easily trade in some of our classification

accuracy for a gain in global sparsity.

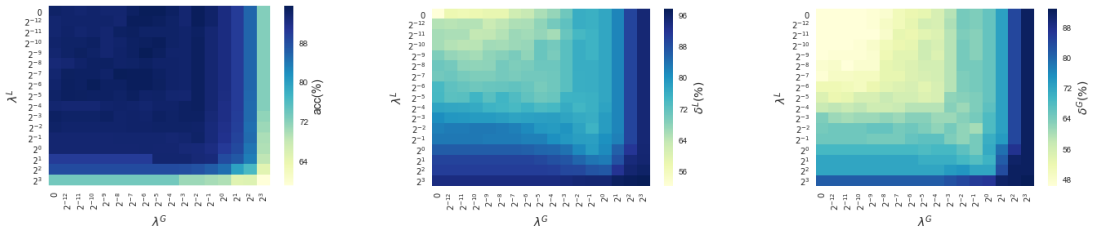
Several extensions to our approach are of interest. First, this metholodogy can be extended to a regression tree counterpart, where the response variable is continuous, as considered in Chapter 4. Second, categorical data is addressed in this chapter through the inclusion of dummy predictor variables. For a given categorical predictor variable, and by means of an ℓ_∞-norm regularization, one can link all its dummies across all the branch nodes in the tree, with the aim of better modeling its contribution to the classifier.



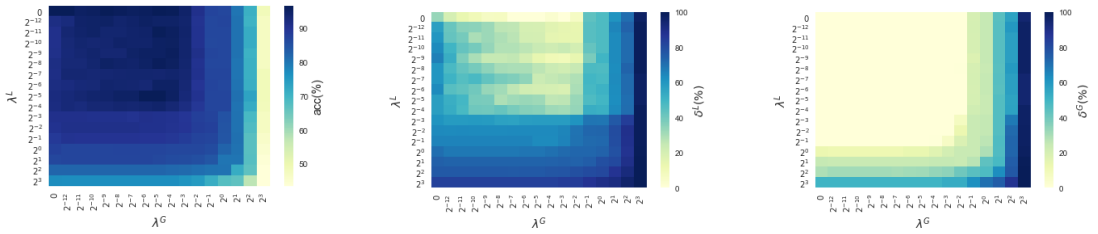
(a) Iris



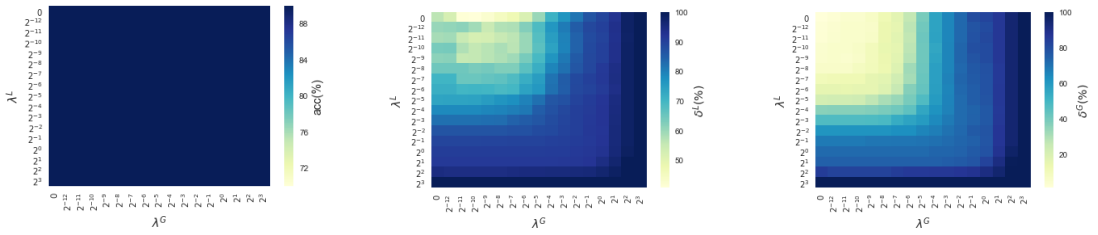
(b) Wine



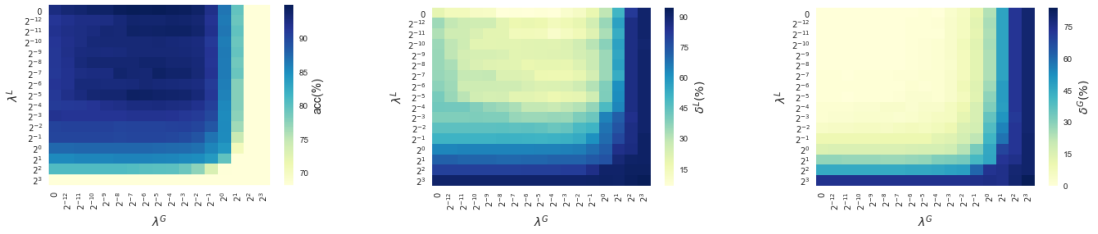
(c) Seeds



(d) Balance

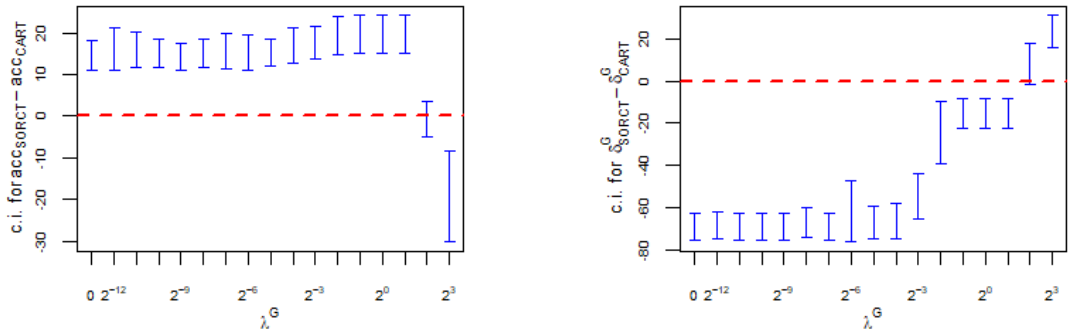


(e) Thyroid

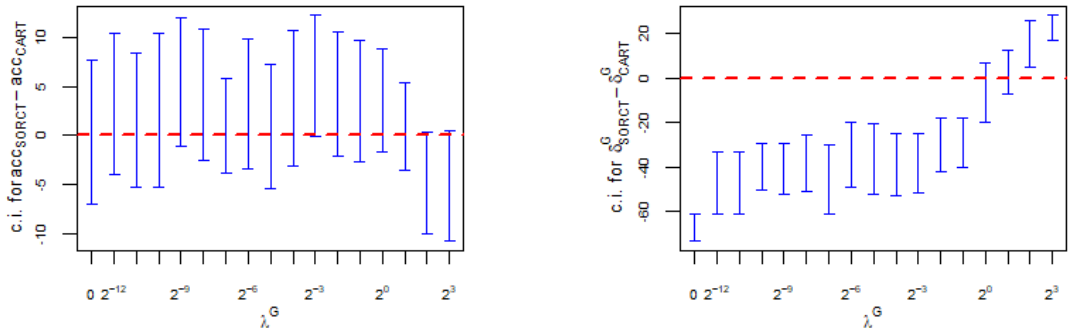


(f) Car

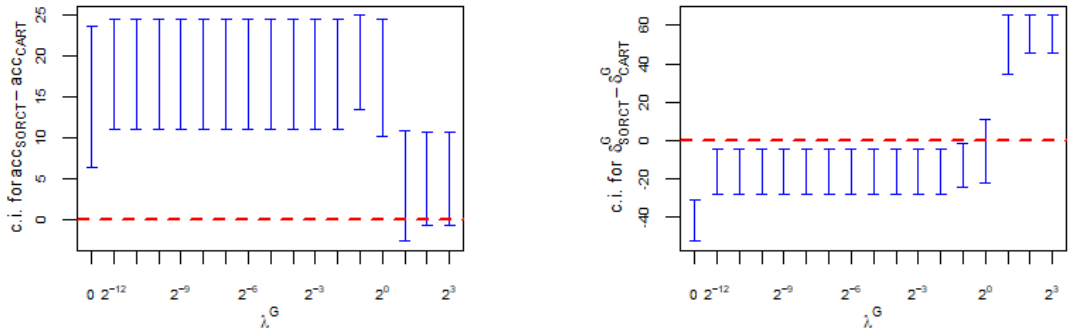
Figure 3.5: Heatmaps representation, for each data set, of the average out-of-sample accuracy, *acc*, the average percentage of predictor variables not used per branch node, δ^L, and the average percentage of predictor variables not used per tree, δ^G, respectively, as a funcion of the grid of the sparsity parameters, λ^L and λ^G, considered in the S-ORCT of depth $D = 2$ construction.



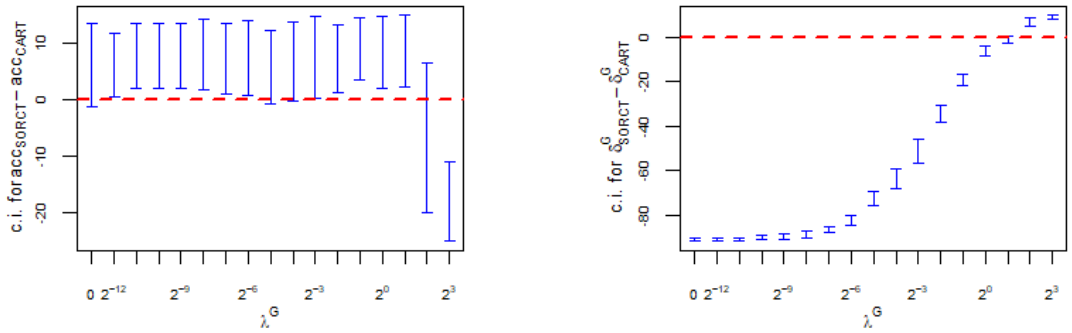
(a) Monks-3



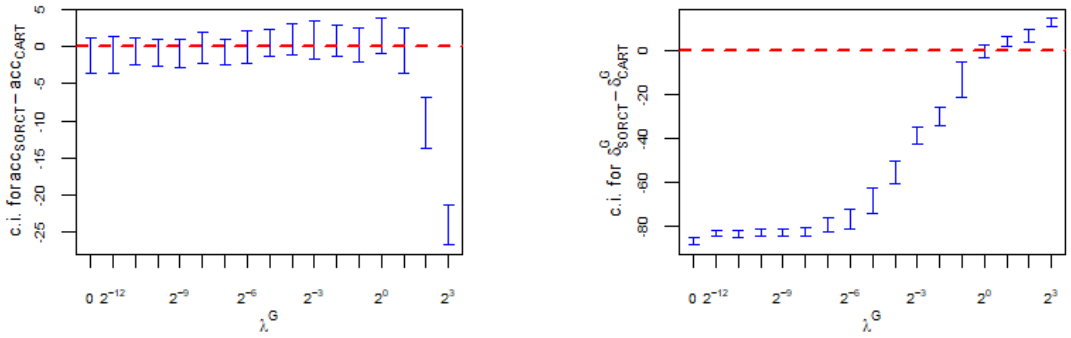
(b) Monks-1



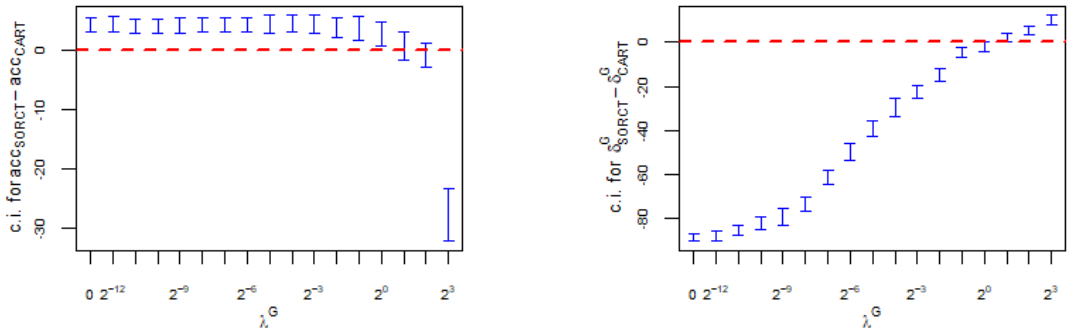
(c) Monks-2



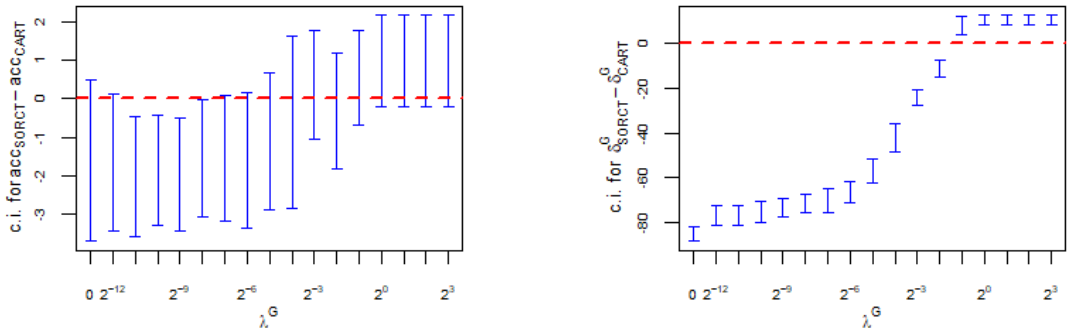
(d) Sonar



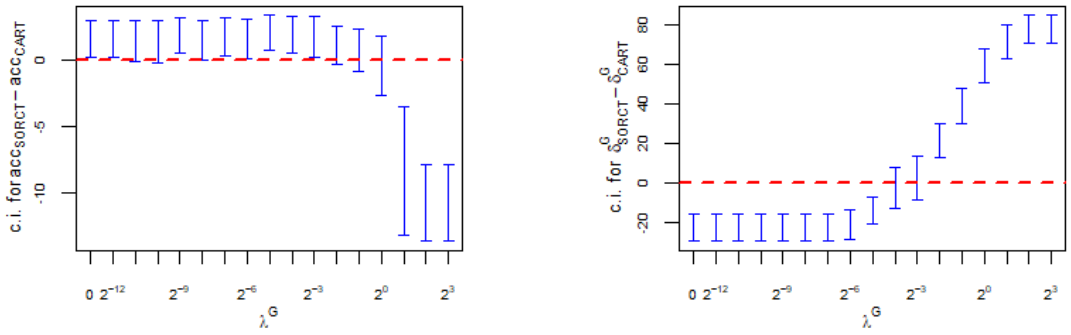
(e) Ionosphere



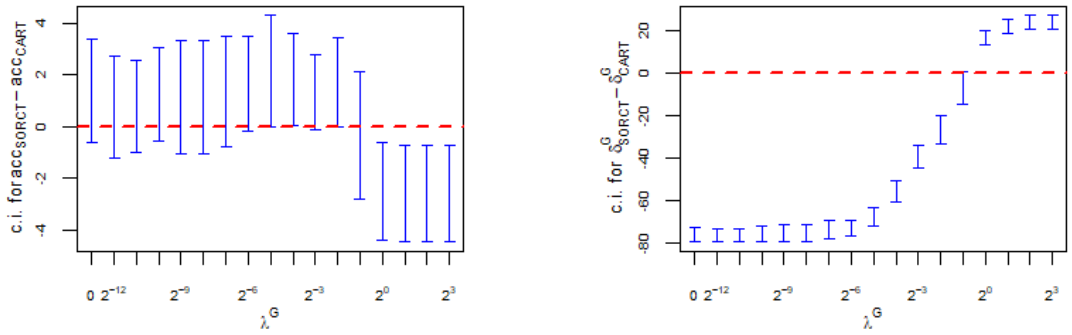
(f) Wisconsin



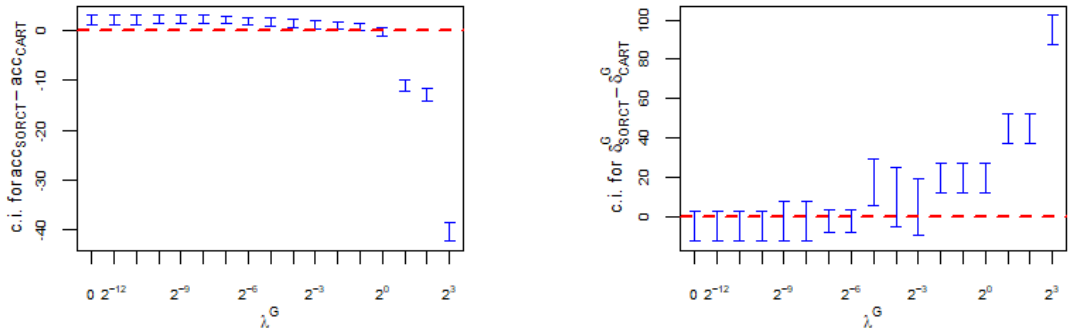
(g) Creditapproval



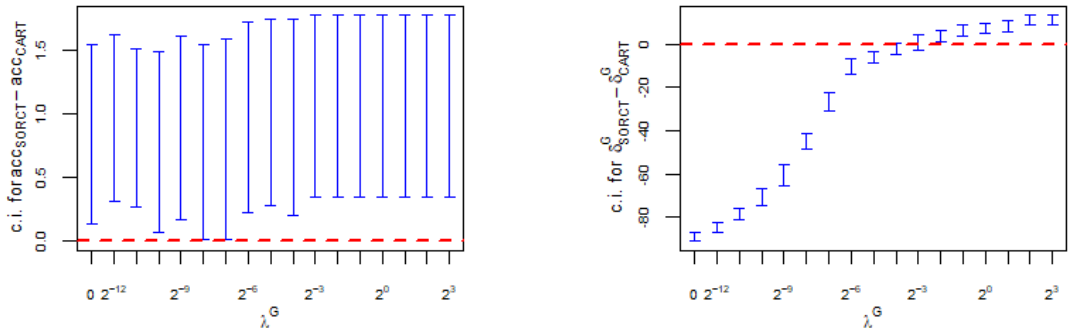
(h) Pima



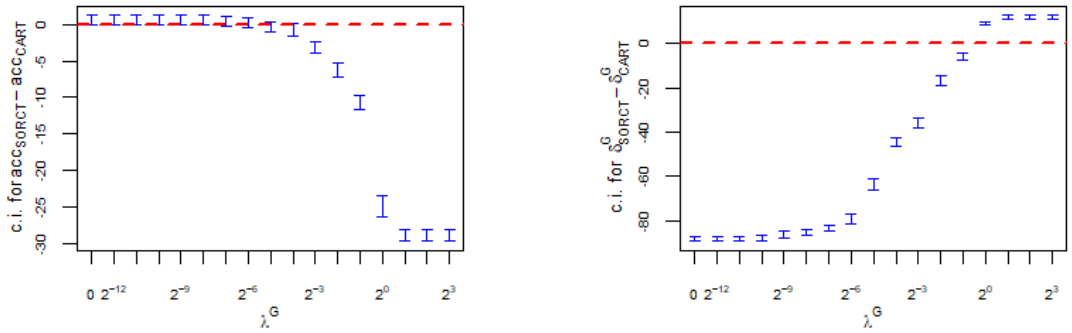
(i) Germancredit



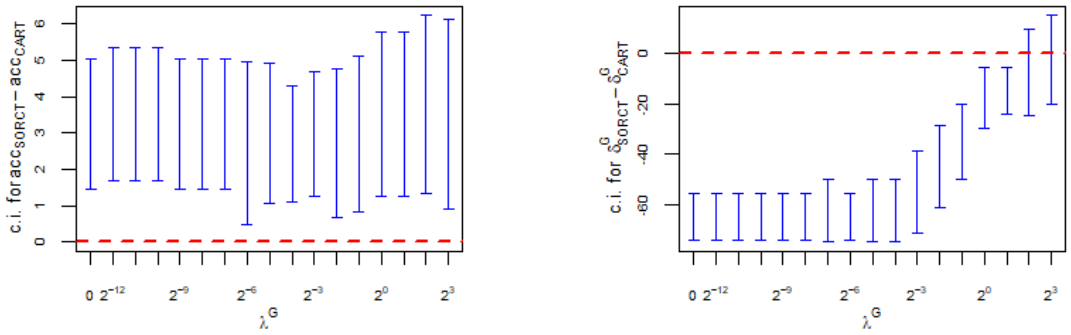
(j) Banknote



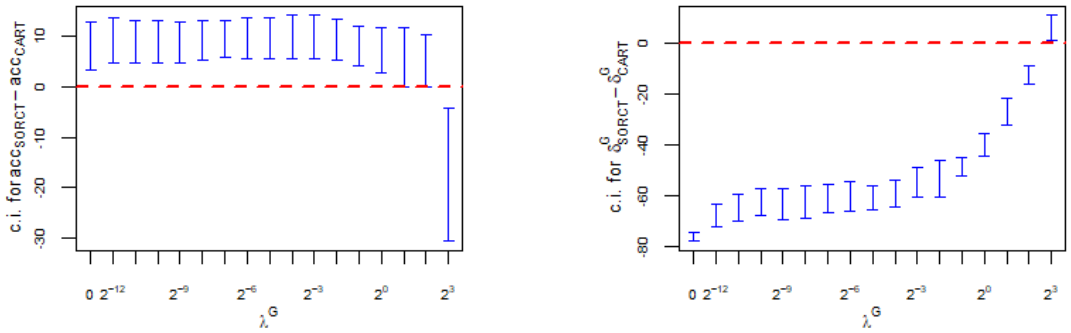
(k) Ozone



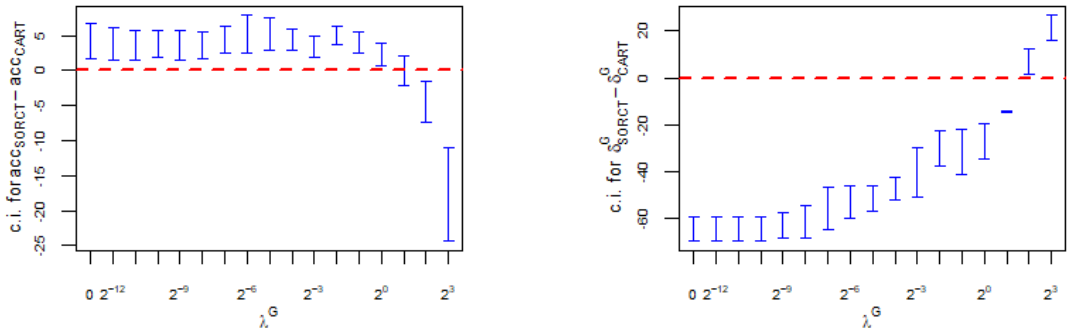
(l) Spam



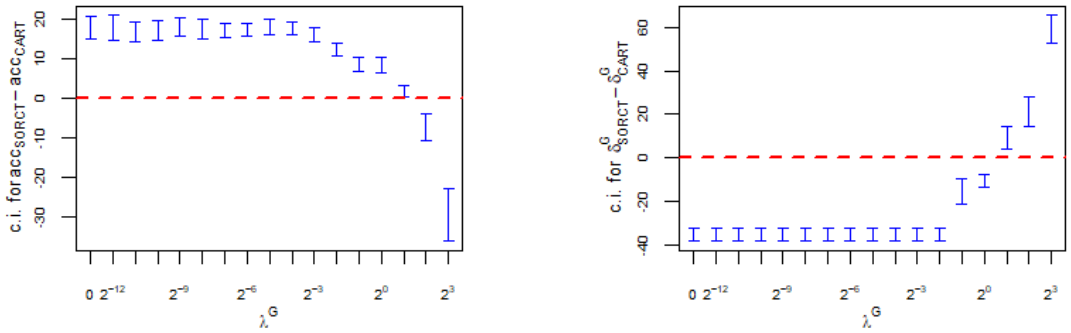
(m) Iris



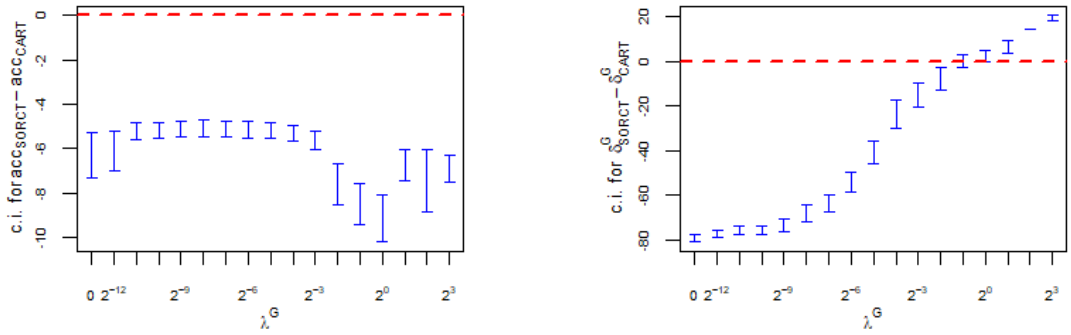
(n) Wine



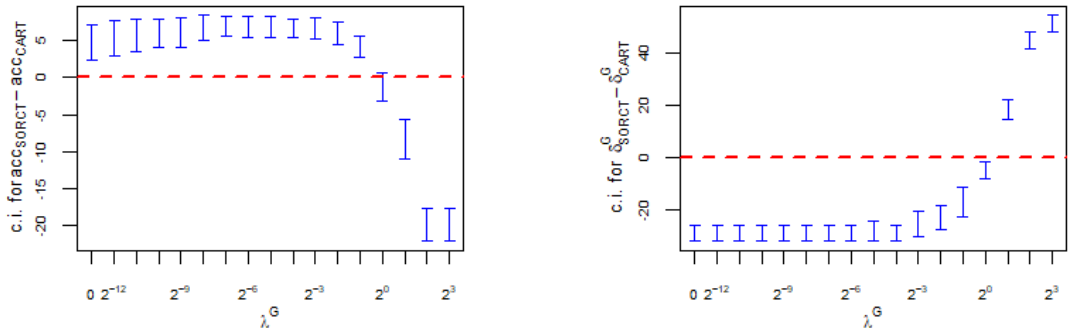
(o) Seeds



(p) Balance



(q) Thyroid



(r) Car

Figure 3.6: Graphical representation, for each data set, of the confidence intervals (blue solid line) at the 95% for the difference in average accuracy (on the left) and global sparsity (on the right) between S-ORCT(λ^G) and CART. The red dashed horizontal line represents the null hypothesis in each case.

Chapter 4

On Sparse Optimal Randomized Regression Trees

In this chapter, the methodology proposed in the previous chapters is adapted to contruct Sparse Optimal Randomized Regression Trees (S-ORRTs) to address regression problems. An S-ORRT is an optimal regression tree modeled through a continuous optimization problem, where a compromise between prediction accuracy and both types of sparsity, namely local and global, is sought. Our approach can accommodate important desirable properties for the regression task, such as cost-sensitivity and fairness. Thanks to the smoothness of the predictions with respect to the features, we can derive local explanations on the continuous predictor variables. The computational experience reported shows the outperformance of our approach in terms of prediction accuracy against standard benchmark regression methods such as CART, OLS and LASSO. Moreover, the scalability of our approach with respect to the size of the training sample is illustrated.

The chapter is organized as follows. In Section 4.1, we introduce the S-ORRT and its mathematical formulation, as well as the modeling of desirable properties. Some theoretical properties of S-ORRT are discussed in Section 4.2. In Section 4.3, our computational experience is reported. We illustrate that S-ORRT outperforms the benchmark regression methods CART, OLS and LASSO in terms of prediction accuracy. Moreover, we show our ability to easily trade in prediction accuracy for a gain in local and global sparsity, as well as our favorable scalability with respect to the size of the training sample. Finally, conclusions and possible extensions are provided in Section 4.4.

4.1 Sparse Optimal Randomized Regression Trees

4.1.1 Introduction

Let $\mathcal{I}$ be a given set of individuals. Each individual $i \in \mathcal{I}$ has associated a pair $(\boldsymbol{x}_i, y_i)$, where $\boldsymbol{x}_i$ represents the p-dimensional vector of numerical predictor variables of individual i, and $y_i \in \mathbb{R}$ indicates the value of the response variable.

A Sparse Optimal Randomized Regression Tree (S-ORRT) is an optimal binary regression tree of a given depth D, obtained by controlling simultaneously prediction accuracy and local and global sparsity. We briefly sketch here this randomized framework. For further details on the construction of optimal randomized trees, the reader is referred to Chapters 2 and 3. Figure 4.1 shows the structure of an S-ORRT of depth $D = 2$. Unlike classic decision trees, oblique cuts, on which more than one predictor variable is involved, are implemented. S-ORRTs are modeled by means of a Non-Linear Continuous Optimization (NLCO) formulation. The usual deterministic yes/no rule at each branch node is replaced by a smoother rule: a probabilistic decision rule at each branch node, induced by a cumulative density function (CDF) F, is obtained. Therefore, the movements in S-ORRTs can be seen as randomized: at a given branch node of a S-ORRT, a random variable will be generated to indicate by which branch an individual has to continue. Since binary trees are built, the Bernoulli distribution is appropriate, whose probability of success will be determined by the value of this CDF, evaluated over the

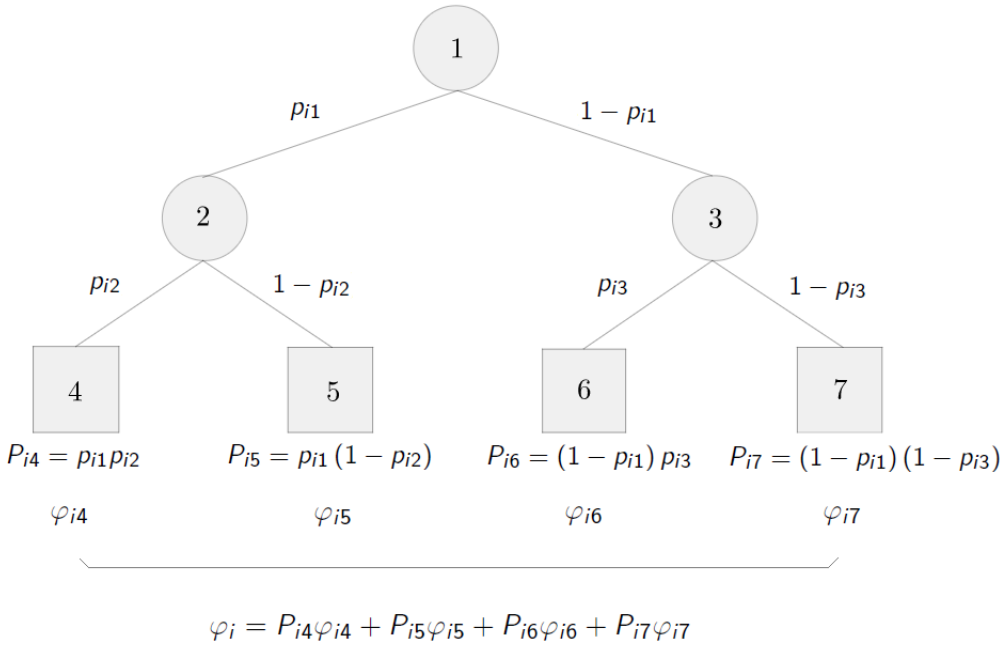


Figure 4.1: Sparse Optimal Randomized Regression Tree of depth $D = 2$.

vector of predictor variables. More precisely, at a given branch node t of the tree, an individual with predictor variables $\boldsymbol{x}_i$ will go either to the left or to the right child nodes with probabilities $F\left(\frac{1}{p}\boldsymbol{a}_{\cdot t}^T\boldsymbol{x}_i - \mu_t\right)$ and $1 - F\left(\frac{1}{p}\boldsymbol{a}_{\cdot t}^T\boldsymbol{x}_i - \mu_t\right)$, respectively, where $\boldsymbol{a}_{\cdot t}$ and μ_t are decision variables of the optimization problem that needs to be solved to build the S-ORRT. In Figure 4.1, p_{i1}, p_{i2}, p_{i3} and their complement to one denote such probabilities for the three branch nodes. With this, we have the probability of each individual in the sample falling into every leaf node. In Figure 4.1, P_{i4}, P_{i5}, P_{i6} and P_{i7} denote such probabilities. To end, we need to define how S-ORRT makes predictions. First, S-ORRT associates linear predictions to each leaf node. Then, the estimated outcome value for each individual is defined as the summation of these linear predictions, weighted by the probability of belonging to the corresponding leaf node. This is denoted by φ_{i4}, φ_{i5}, φ_{i6} and φ_{i7} in Figure 4.1.

The following notation is required:

Parameters	
D	depth of the binary tree,
p	number of predictor variables,
$\{(\boldsymbol{x}_i, y_i)\}_{i\in\mathcal{I}}$	training sample, where $\boldsymbol{x}_i \in [0,1]^p$ and $y_i \in \mathbb{R}$, with cardinality $\lvert\mathcal{I}\rvert$,
$F(\cdot)$	the smooth CDF of a univariate continuous random variable centered at zero, used to define the probabilities for an individual to go to the left or the right child node in the tree,
λ^L, λ^G	local and global sparsity regularization parameters.

Nodes

τ_B	set of branch nodes,
τ_L	set of leaf nodes,
$N_L(t)$	set of ancestor nodes of leaf node t whose left branch takes part in the path from the root node to leaf node t, $t \in \tau_L$,
$N_R(t)$	set of ancestor nodes of leaf node t whose right branch takes part in the path from the root node to leaf node t, $t \in \tau_L$.

Decision variables

$a_{jt} \in \mathbb{R}$	coefficient of predictor variable j in the oblique cut at branch node $t \in \tau_B$. The expression $\boldsymbol{a}$ will denote the $p \times \lvert\tau_B\rvert$-matrix that involve these coefficients $\boldsymbol{a} = (a_{jt})_{j=1,\ldots,p,\ t\in\tau_B}$. The expressions $\boldsymbol{a}_{j\cdot}$ and $\boldsymbol{a}_{\cdot t}$ will denote the j-th row and the t-th column of $\boldsymbol{a}$, respectively,
$\mu_t \in \mathbb{R}$	location parameter at branch node $t \in \tau_B$. The expression $\boldsymbol{\mu}$ will denote the $\lvert\tau_B\rvert$- vector that involves these coefficients $\boldsymbol{\mu} = (\mu_t)_{t\in\tau_B}$,
$\tilde{a}_{jt} \in \mathbb{R}$	coefficient of predictor variable j in the linear prediction at leaf node $t \in \tau_L$. The expression $\tilde{\boldsymbol{a}}$ will denote the $p \times \lvert\tau_L\rvert$-matrix that involves these coefficients $\tilde{\boldsymbol{a}} = (\tilde{a}_{jt})_{j=1,\ldots,p,\ t\in\tau_L}$. The expressions $\tilde{\boldsymbol{a}}_{j\cdot}$ and $\tilde{\boldsymbol{a}}_{\cdot t}$ will denote the j-th row and the t-th column of $\tilde{\boldsymbol{a}}$, respectively,
$\tilde{\mu}_t \in \mathbb{R}$	intercept of the linear prediction at leaf node $t \in \tau_L$. The expression $\tilde{\boldsymbol{\mu}}$ will denote the $\lvert\tau_L\rvert$-vector that involves these coefficients $\tilde{\boldsymbol{\mu}} = (\mu_t)_{t\in\tau_L}$.

Probabilities

$p_{it}(\boldsymbol{a}_{\cdot t}, \mu_t)$	probability of individual i going down the left branch at branch node t. Its expression is $p_{it}(\boldsymbol{a}_{\cdot t}, \mu_t) = F\left(\frac{1}{p}\boldsymbol{a}_{\cdot t}^{\top}\boldsymbol{x}_i - \mu_t\right),\ i \in \mathcal{I},\ t \in \tau_B$,
$P_{it}(\boldsymbol{a}, \boldsymbol{\mu})$	probability of individual i falling into leaf node t. Its expression is $P_{it}(\boldsymbol{a}, \boldsymbol{\mu}) = \prod_{t_l\in N_L(t)} p_{it_l}(\boldsymbol{a}_{\cdot t_l}, \mu_{t_l}) \prod_{t_r\in N_R(t)} (1 - p_{it_r}(\boldsymbol{a}_{\cdot t_r}, \mu_{t_r})), i \in \mathcal{I},\ t \in \tau_L$.

Predictions

$\varphi_{it}(\tilde{\boldsymbol{a}}_{\cdot t}, \tilde{\mu}_t)$	linear prediction of individual i at leaf node t. Its expression is $\varphi_{it}(\tilde{\boldsymbol{a}}_{\cdot t}, \tilde{\mu}_t) = \tilde{\boldsymbol{a}}_{\cdot t}^{\top}\boldsymbol{x}_i - \tilde{\mu}_t,\ i \in \mathcal{I},\ t \in \tau_L$,
$\varphi_i(\boldsymbol{a}, \boldsymbol{\mu}, \tilde{\boldsymbol{a}}, \tilde{\boldsymbol{\mu}})$	final prediction of individual i. Its expression is $\varphi_i(\boldsymbol{a}, \boldsymbol{\mu}, \tilde{\boldsymbol{a}}, \tilde{\boldsymbol{\mu}}) = \sum_{t\in\tau_L} P_{it}(\boldsymbol{a}, \boldsymbol{\mu})\varphi_{it}(\tilde{\boldsymbol{a}}_{\cdot t}, \tilde{\mu}_t),\ i \in \mathcal{I}$. In other words, for an individual i, its prediction is a weighted average of the predictions φ_{it} along the different leaf nodes, where the weights in such average depend on the individual i.

4.1.2 The formulation

With these parameters and decision variables, the S-ORRT reads as the following unconstrained NLCO problem:

$$\min_{\boldsymbol{a},\boldsymbol{\mu},\tilde{\boldsymbol{a}},\tilde{\boldsymbol{\mu}}} \quad \left\{ \text{MSE}(\boldsymbol{a},\boldsymbol{\mu},\tilde{\boldsymbol{a}},\tilde{\boldsymbol{\mu}};\mathcal{I}) \;+\; \lambda^L \sum_{j=1}^{p} \|(\boldsymbol{a}_{j\cdot},\tilde{\boldsymbol{a}}_{j\cdot})\|_1 + \; \lambda^G \sum_{j=1}^{p} \|(\boldsymbol{a}_{j\cdot},\tilde{\boldsymbol{a}}_{j\cdot})\|_\infty \right\}, \tag{4.1}$$

where

$$\text{MSE}\left(\boldsymbol{a},\boldsymbol{\mu},\tilde{\boldsymbol{a}},\tilde{\boldsymbol{\mu}};\mathcal{I}\right) = \frac{1}{|\mathcal{I}|}\sum_{i\in\mathcal{I}}\left(\varphi_i\left(\boldsymbol{a},\boldsymbol{\mu},\tilde{\boldsymbol{a}},\tilde{\boldsymbol{\mu}}\right) - y_i\right)^2.$$

The first term, prediction accuracy, is equal to the mean squared error over the training sample between the actual response values and the predictions returned by S-ORRT. The second term controls local sparsity, since it penalizes the ℓ_1-norm of the coefficients of the predictor variables used in all the nodes of the tree. The third term addresses global sparsity, which is modeled by the inclusion of a penalization term that controls whether a given predictor variable is ever used across the whole tree. Recall that each predictor variable appears at both branch (in the oblique cuts) and leaf (in the linear predictions) nodes. Then, the ℓ_∞-norm is used as a group penalty function, by forcing all the coefficients linked to the same predictor variable to be shrunk simultaneously along all branch and leaf nodes.

Since there are no decision variables directly relating to the number of individuals N, Problem (4.1) speaks favorably toward the scalability of S-ORRT with respect to the size of the training sample. Hence, although the evaluation of the first term in the objective function becomes more time demanding with larger N, the number of decision variables of the problem to be solved remains the same. This makes our approach scalable with respect to N, as illustrated in Section 4.3.4.

Once the tree model is built, the prediction of future data is done as follows. Let $(\boldsymbol{a}^*,\boldsymbol{\mu}^*,\tilde{\boldsymbol{a}}^*,\tilde{\boldsymbol{\mu}}^*)$ be the optimal solution to Problem (4.1). The expected outcome of individual $i \in \mathcal{I}$ is $\varphi_i\left(\boldsymbol{a}^*,\boldsymbol{\mu}^*,\tilde{\boldsymbol{a}}^*,\tilde{\boldsymbol{\mu}}^*\right)$. For an incoming individual with predictor vector $\mathbf{x}$, the expected outcome returned by the randomized tree is equal to

$$\mathbf{x} \to \Pi(\mathbf{x}) := \varphi_{\mathbf{x}}\left(\boldsymbol{a}^*,\boldsymbol{\mu}^*,\tilde{\boldsymbol{a}}^*,\tilde{\boldsymbol{\mu}}^*\right), \tag{4.2}$$

where $\varphi_{\mathbf{x}}$ is defined similarly to φ_i with $\mathbf{x}$ replacing $\mathbf{x}_i$. Note that $\Pi(\cdot)$ is smooth in the continuous predictor variables, since the CDF F is assumed to be a smooth function. This means that even small changes in these variables will produce changes in $\Pi(\cdot)$. This is not the case for deterministic tree models such as CART and RF, where there are no changes at all in

the expected outcome when there are small changes in the continuous predictor variables. This inherent property of our approach allows us to perform local explainability, as will be seen in Section 4.1.4.

4.1.3 A smooth reformulation

Problem (4.1) is non-smooth due to the ℓ_1 and ℓ_∞ norms appearing in the objective function. Recall that F is assumed to be continuously differentiable, therefore MSE inherits smoothness. By rewriting both regularization terms using new decision variables, we can formulate S-ORRT as a smooth problem, thus solvable with standard continuous optimization solvers, as done in our computational section.

Regarding the first regularization term of Problem (4.1), decision variables $\mathbf{a}$ and $\tilde{\mathbf{a}}$ are split into their positive and negative counterparts, $\boldsymbol{a}^+, \tilde{\boldsymbol{a}}^+ = \left(a_{jt}^+\right)_{j=1,\dots,p,\ t\in\tau_B}, \left(\tilde{a}_{jt}^+\right)_{j=1,\dots,p,\ t\in\tau_L}$ and $\boldsymbol{a}^-, \tilde{\boldsymbol{a}}^- = \left(a_{jt}^-\right)_{j=1,\dots,p,\ t\in\tau_B}, \left(\tilde{a}_{jt}^-\right)_{j=1,\dots,p,\ t\in\tau_L}$, respectively, such that $a_{jt} = a_{jt}^+ - a_{jt}^-$, $\tilde{a}_{jt} = \tilde{a}_{jt}^+ - \tilde{a}_{jt}^-$, $|a_{jt}| = a_{jt}^+ + a_{jt}^-$, $|\tilde{a}_{jt}| = \tilde{a}_{jt}^+ + \tilde{a}_{jt}^-$ and $a_{jt}^+, a_{jt}^-, \tilde{a}_{jt}^+, \tilde{a}_{jt}^- \geq 0$, thus having

$$\|(\boldsymbol{a}_{j\cdot}, \tilde{\boldsymbol{a}}_{j\cdot})\|_1 = \sum_{t\in\tau_B} |a_{jt}| + \sum_{t\in\tau_L} |\tilde{a}_{jt}| = \sum_{t\in\tau_B} \left(a_{jt}^+ + a_{jt}^-\right) + \sum_{t\in\tau_L} \left(\tilde{a}_{jt}^+ + \tilde{a}_{jt}^-\right),\ j = 1,\dots,p.$$

New decision variables $\boldsymbol{\beta} = (\beta_j)_{j=1,\dots,p}$ are used to model the second regularization term of Problem (4.1):

$$\|(\boldsymbol{a}_{j\cdot}, \tilde{\boldsymbol{a}}_{j\cdot})\|_\infty = \max\left(\{|a_{jt}|\}_{t\in\tau_B} \cup \{|\tilde{a}_{jt}|\}_{t\in\tau_L} \right) = \beta_j,\ j = 1,\dots,p,$$

where $\beta_j \geq 0$. We also need to impose $\beta_j \geq \pm\, a_{jt},\ j = 1,\dots,p,\ t \in \tau_B$, and $\beta_j \geq \pm\, \tilde{a}_{jt},\ j = 1,\dots,p,\ t \in \tau_L$. Hence, we have that Problem (4.1) is equivalent to the following smooth reformulation:

$$\min_{\substack{(\boldsymbol{a}^+,\boldsymbol{a}^-,\boldsymbol{\mu})\in\mathbb{R}^{(2p+1)|\tau_B|} \\ (\tilde{\boldsymbol{a}}^+,\tilde{\boldsymbol{a}}^-,\tilde{\boldsymbol{\mu}})\in\mathbb{R}^{(2p+1)|\tau_L|} \\ \boldsymbol{\beta}\in\mathbb{R}^p}} \quad \text{MSE}\left(\boldsymbol{a}^+ - \boldsymbol{a}^-, \boldsymbol{\mu}, \tilde{\boldsymbol{a}}^+ - \tilde{\boldsymbol{a}}^-, \tilde{\boldsymbol{\mu}}; \mathcal{I}\right) \tag{4.3}$$

$$+ \lambda^L \sum_{j=1}^{p} \left(\sum_{t\in\tau_B} \left(a_{jt}^+ + a_{jt}^-\right) + \sum_{t\in\tau_L} \left(\tilde{a}_{jt}^+ + \tilde{a}_{jt}^-\right) \right) \tag{4.4}$$

$$+ \lambda^G \sum_{j=1}^{p} \beta_j \tag{4.5}$$

$$\text{s.t.} \quad \beta_j \geq a_{jt}^+ + a_{jt}^-,\ j = 1,\dots,p,\ t \in \tau_B, \tag{4.6}$$

$$\beta_j \geq \tilde{a}_{jt}^+ + \tilde{a}_{jt}^-,\ j = 1,\dots,p,\ t \in \tau_L, \tag{4.7}$$

$$a_{jt}^+, a_{jt}^-, \tilde{a}_{jt}^+, \tilde{a}_{jt}^-, \beta_j \geq 0,\ j = 1,\dots,p,\ t \in \tau_B \cup \tau_L. \tag{4.8}$$

4.1.4 Desirable properties

As we show in this section, our approach can easily accommodate important desirable properties in the regression task, such as cost-sensitivity and fairness, as well as local explainability.

Cost-sensitivity As a regression method, S-ORRT seeks a rule yielding a good overall prediction accuracy, although, at times, there are groups of individuals in which predicion errors are more critical. It is then more adequate not only to focus on the overall prediction accuracy, but also ensuring a certain level of performance in those groups. S-ORRT is flexible enough to allow incorporating constraints on expected performance [Blanquero et al., 2021b; Günlük et al., 2021] over critical groups. Let $\mathcal{J}_1, \ldots, \mathcal{J}_r$ be different samples, possibly subsamples of $\mathcal{I}$. Given a threshold value ρ_j for the desired performance on sample $\mathcal{J}_j$, one can simply add the following constraints to Problem (4.1):

$$\text{MSE}(\boldsymbol{a}, \boldsymbol{\mu}, \tilde{\boldsymbol{a}}, \tilde{\boldsymbol{\mu}}; \mathcal{J}_j) \leq \rho_j, \quad j = 1, \ldots, r.$$

Fairness The increase in automatization in decision-making has evinced the bias present on historical data, leading to models that may discriminate groups sharing sensitive features such as gender or race. In this line, we seek for a model that avoids such discrimination and is fair to a sensitive group [Aghaei et al., 2019; Obermeyer et al., 2019]. Let $\mathcal{S} \subset \mathcal{I}$ be a group of individuals to be protected against discrimination by Problem (4.1). There are different ways to handle fairness. For instance, we may impose that the prediction errors for individuals in $\mathcal{S}$ does not differ much from the prediction errors in the whole training sample $\mathcal{I}$. This can be modeled through the following constraint

$$|\text{MSE}(\boldsymbol{a}, \boldsymbol{\mu}, \tilde{\boldsymbol{a}}, \tilde{\boldsymbol{\mu}}; \mathcal{S}) - \text{MSE}(\boldsymbol{a}, \boldsymbol{\mu}, \tilde{\boldsymbol{a}}, \tilde{\boldsymbol{\mu}}; \mathcal{I})| \leq C,$$

for $C \geq 0$ sufficiently small. Alternatively, we may impose that the average prediction for individuals in $\mathcal{S}$ does not differ much from the average in the whole training sample $\mathcal{I}$, i.e.,

$$|\bar{\varphi}(\boldsymbol{a}, \boldsymbol{\mu}, \tilde{\boldsymbol{a}}, \tilde{\boldsymbol{\mu}}; S) - \bar{\varphi}(\boldsymbol{a}, \boldsymbol{\mu}, \tilde{\boldsymbol{a}}, \tilde{\boldsymbol{\mu}}; \mathcal{I})| \leq C, \tag{4.9}$$

where $\bar{\varphi}(\boldsymbol{a}, \boldsymbol{\mu}, \tilde{\boldsymbol{a}}, \tilde{\boldsymbol{\mu}}; \mathcal{J}) = \dfrac{1}{|\mathcal{J}|} \displaystyle\sum_{i \in \mathcal{J}} \varphi_i(\boldsymbol{a}, \boldsymbol{\mu}, \tilde{\boldsymbol{a}}, \tilde{\boldsymbol{\mu}})$ and $C \geq 0$ sufficiently small. Fairness as in Equation (4.9) is illustrated for the `Boston Housing` data set [Harrison Jr and Rubinfeld, 1978]. See Table 4.2 for a description of the response and predictor variables. Suppose that our sensitive group $\mathcal{S}$ is composed by individuals above the third quartile of predictor variable `B`, that is, those census tracts where there is a high proportion of black population. The S-ORRT without fairness constraints, and $\lambda^L = \lambda^G = 0$, yields a mean squared error

of 9.6462, with an average prediction on housing values over $\mathcal{I}$ equal to 22.5333. A lower average value is obtained over $\mathcal{S}$, 21.3263, producing an absolute difference of $C_0 = 1.2070$. See the first row in Table 4.1. The next rows represent the results when fairness constraints

Table 4.1: Results of S-ORRT without and with fairness constraints on $\mathcal{S}$ in the `Boston Housing` data set, where $C_0 = 1.2070$.

τ	$C = \tau \cdot C_0$	MSE $(\cdot; \mathcal{I})$	$\bar{\varphi}(\cdot; \mathcal{I})$	$\bar{\varphi}(\cdot; \mathcal{S})$
-	-	9.6462	22.5333	21.3263
0.75	0.9053	9.7586	22.5327	21.6275
0.5	0.6035	10.0282	22.5334	21.9298
0.25	0.3018	10.5051	22.5330	22.1312
0	0	11.2401	22.5332	22.5332

over $\mathcal{S}$ are added to the model for several values of the threshold $C = \tau \cdot C_0$, with τ varying in $\{0.75, 0.5, 0.25, 0\}$. As shown, one can obtain a S-ORRT which is fair to our sensitive group $\mathcal{S}$, since $\bar{\varphi}(\boldsymbol{a}, \boldsymbol{\mu}, \tilde{\boldsymbol{a}}, \tilde{\boldsymbol{\mu}}; \mathcal{I}) = \bar{\varphi}(\boldsymbol{a}, \boldsymbol{\mu}, \tilde{\boldsymbol{a}}, \tilde{\boldsymbol{\mu}}; \mathcal{S})$, at the expense of slightly harming prediction accuracy.

Local explainability The goal of local explainability [Lundberg et al., 2020; Lundberg and Lee, 2017; Molnar et al., 2020; Ribeiro et al., 2016] is to identify the predictor variables that have the largest impact on the individual predictions, found in Equation (4.2). For nonlinear models one can make use of generic post-hoc approaches to build local explanations, such as the so-called Local Interpretable Model-agnostic Explanations (LIME) [Ribeiro et al., 2016]. Instead, and as advocated by Rudin [2019], one can work with models that derive local explanations directly [Gevrey et al., 2003], as we do on the continuous predictor variables thanks to the smoothness of Π. For simplicity, we consider a problem where all predictor variables are continuous. For an individual with predictor variables $\boldsymbol{x}^0$, we analyze how sensitive Π is to an infinitesimal change $\boldsymbol{\Delta} \in \mathbb{R}^p$, i.e., how large is the difference $\Pi(\boldsymbol{x}^0 + \boldsymbol{\Delta}) - \Pi(\boldsymbol{x}^0)$. By linearizing Π close to $\boldsymbol{x}^0$, we have

$$\Pi(\boldsymbol{x}^0 + \boldsymbol{\Delta}) \approx \Pi(\boldsymbol{x}^0) + \sum_{j=1}^{p} \frac{\partial \Pi}{\partial x_j}(\boldsymbol{x}^0) \cdot \Delta_j.$$

Thus, the vector of partial derivatives

$$\left(\frac{\partial \Pi}{\partial x_j}(\boldsymbol{x}^0) \right)_{j=1,\ldots,p} \tag{4.10}$$

gives full information on the sensitivity of the outcomes Π around $\boldsymbol{x}^0$. A positive value of coordinate j of the vector of partial derivatives means a direct relationship between predictor variable j and prediction of the response variable of individual $\boldsymbol{x}^0$; and an inverse relationship, otherwise. As opposed to linear regression, where there is one single coefficient per predictor

variable that indicates its impact in prediction for any individual equally, here we have different impacts of each predictor variable tailored to each particular individual.

Local explainability is illustrated below for the `Boston Housing` data set in Table 4.2.

Table 4.2: Information about the `Boston Housing` data set, which consists of a collection of 506 observations about housing values for census tracts of the Boston metropolitan area.

Variable	Name	Description
Predictor	`CRIM`	crime rate by town
	`ZN`	proportion of residential land zoned for lots greater than 25,000 squared feet
	`INDUS`	proportion of nonretail business acres per town
	`CHAS`	1 if tract bounds river; 0 otherwise
	`NOX`	nitrogen oxide concentration in parts per hundred million
	`RM`	average number of rooms in owner units
	`AGE`	proportion of owner units built prior to 1940
	`DIS`	weighted distances to five employment centers in the Boston region
	`RAD`	index of accessibility to radial highways
	`TAX`	full value property tax rate per ten thousands of dollars
	`PTRATIO`	pupil-teacher ratio by town school district
	`B`	black proportion of population
	`LSTAT`	proportion of population that is lower status
Response	`MEDV`	median value of owner-occupied homes in thousands of dollars

An S-ORRT with $\lambda^L = 0$ and $\lambda^G = \dfrac{2^2}{13}$ was built on this data set, obtaining a mean squared error and an R-squared equal to 15.5654 and 0.8156, respectively. Figure 4.2 depicts the local explanations for all individuals in the dataset by means of parallel coordinates. Each predictor variable is represented by a vertical parallel axis. Each individual is represented by a series of lines connected across all the axes. The position each individual takes on each axis reflects the impact the corresponding predictor variable has on its prediction, that is, each of the coordinates of vector (4.10). The color that represents each individual in the parallel coordinates goes from light pink to purple depending on the reliability of the prediction, measured as the ratio between the individual squared error and the mean squared error. Thus, purple refers to the best reliable predictions according to the model. All predictor variables were normalized before training the model to the 0-1 interval, in such a way that a fair comparative analysis between them could be performed. A larger absolute value on the axis represents a larger impact caused by the corresponding predictor variable on the prediction. Since `CRIM` gauges the threat to well-being that households perceive, it has a negative effect on housing values. A similar pattern is observed for `NOX`, `DIS`, `TAX`, `PTRATIO`, as well as for `LSTAT`, which means that an area with a high amount of lower status population would have less valuable households. Other predictor variables have a positive effect on housing values. For instance, `RM`, which represents spaciousness and it can be observed that it is directly related to a higher housing value.

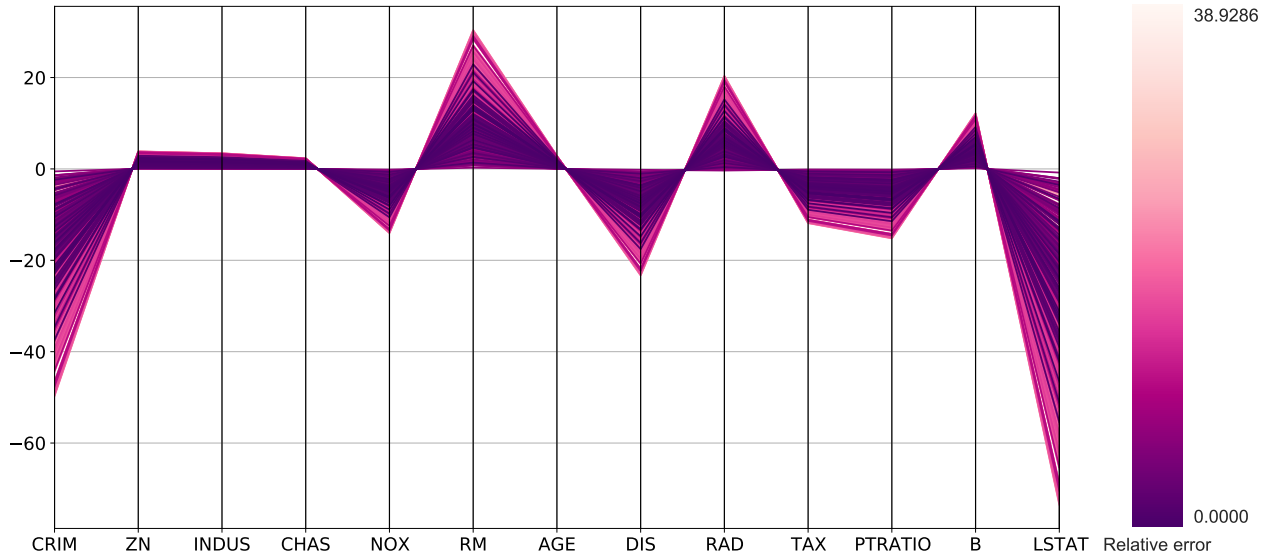


Figure 4.2: Local explainability for `Boston Housing` data set derived from the S-ORRT with $\lambda^L = 0$ and $\lambda^G = \dfrac{2^2}{13}$ and a mean squared error and an R-squared of 15.5654 and 0.8156, respectively.

4.2 Theoretical properties

In this section, some theoretical properties enjoyed by S-ORRT, as formulated in Problem (4.1), are analyzed. In particular, we pay attention to the sparsest tree, obtained when the optimal solution of S-ORRT includes $(\boldsymbol{a}^*, \tilde{\boldsymbol{a}}^*) = \mathbf{0}$, and thus no predictor variable is used in the predictions. This is attained when the sparsity regularization parameters, λ^L and λ^G, are taken large enough, and the first term related to the prediction accuracy of the regressor becomes negligible. In the following, we study the optimal prediction returned by S-ORRT with $(\boldsymbol{a}^*, \tilde{\boldsymbol{a}}^*) = \mathbf{0}$, and derive upper bounds for λ^L and λ^G in the sense that above them the sparsest tree (with $(\boldsymbol{a}^*, \tilde{\boldsymbol{a}}^*) = \mathbf{0}$) is a stationary point of the S-ORRT, that is, there exists $(\boldsymbol{a}^* = \mathbf{0}, \boldsymbol{\mu}^*, \tilde{\boldsymbol{a}}^* = \mathbf{0}, \tilde{\boldsymbol{\mu}})$ such that the necessary optimality condition with respect to $(\boldsymbol{a}^*, \tilde{\boldsymbol{a}}^*)$ is satisfied. In Section 4.3, we illustrate when these upper bounds are already reached, by showing that above certain values of λ^L and λ^G, the highest levels of local and global sparsity, respectively, are achieved. See in Figure 4.3 that for $\left(\lambda^L, \lambda^G\right) = \left(\dfrac{2^2}{120}, \dfrac{2^2}{40}\right)$, the sparsest S-ORRT is already obtained, while not producing the best performance in terms of prediction accuracy.

First, observe that, for any $\boldsymbol{a}$, $\boldsymbol{\mu}$ and $\tilde{\boldsymbol{a}}$ fixed, Problem (4.1) can be easily reformulated as a linear regression problem. Indeed, we have that the final prediction of each individual is

$$\varphi_i\left(\boldsymbol{a}, \boldsymbol{\mu}, \tilde{\boldsymbol{a}}, \tilde{\boldsymbol{\mu}}\right) = \sum_{t \in \tau_L} P_{it}\left(\boldsymbol{a}, \boldsymbol{\mu}\right)\left(\tilde{\boldsymbol{a}}_{\cdot t}^{\top} \boldsymbol{x}_i - \tilde{\mu}_t\right), \ i \in \mathcal{I},$$

and thus, defining

$$\eta_i\left(\boldsymbol{a}, \boldsymbol{\mu}, \tilde{\boldsymbol{a}}\right) = \sum_{t \in \tau_L} P_{it}\left(\boldsymbol{a}, \boldsymbol{\mu}\right)\left(\tilde{\boldsymbol{a}}_{\cdot t}^{\top} \boldsymbol{x}_i - y_i\right),\ i \in \mathcal{I},$$

the MSE term in Problem (4.1) can be rewritten as

$$\frac{1}{|\mathcal{I}|} \sum_{i \in \mathcal{I}} \left(\eta_i\left(\boldsymbol{a}, \boldsymbol{\mu}, \tilde{\boldsymbol{a}}\right) - \sum_{t \in \tau_L} P_{it}\left(\boldsymbol{a}, \boldsymbol{\mu}\right) \mu_t \right)^2,$$

or, in matrix form,

$$\frac{1}{|\mathcal{I}|} \left\| \boldsymbol{\eta}\left(\boldsymbol{a}, \boldsymbol{\mu}, \tilde{\boldsymbol{a}}\right) - \boldsymbol{P}\left(\boldsymbol{a}, \boldsymbol{\mu}\right) \tilde{\boldsymbol{\mu}} \right\|^2,$$

where

$$\boldsymbol{\eta}\left(\boldsymbol{a}, \boldsymbol{\mu}, \tilde{\boldsymbol{a}}\right) = \left(\eta_i\left(\boldsymbol{a}, \boldsymbol{\mu}, \tilde{\boldsymbol{a}}\right)\right)_{i \in \mathcal{I}}$$

and

$$\boldsymbol{P}\left(\boldsymbol{a}, \boldsymbol{\mu}\right) = \left[\ P_{it}\left(\boldsymbol{a}, \boldsymbol{\mu}\right)\ \right]_{i \in \mathcal{I},\ t \in \tau_L}.$$

Then, minimizing MSE for $\boldsymbol{a}, \boldsymbol{\mu}, \tilde{\boldsymbol{a}}$ fixed amounts to finding the Ordinary Least Squares solution with design matrix $\boldsymbol{P}\left(\boldsymbol{a}, \boldsymbol{\mu}\right)$ and response vector $\boldsymbol{\eta}\left(\boldsymbol{a}, \boldsymbol{\mu}, \tilde{\boldsymbol{a}}\right)$. With this, the following is shown:

Proposition 4.1. *For $\left(\boldsymbol{a}^*, \boldsymbol{\mu}^*, \tilde{\boldsymbol{a}}^*\right)$ fixed, $\tilde{\boldsymbol{\mu}}^*$ minimizes MSE $\left(\boldsymbol{a}^*, \boldsymbol{\mu}^*, \tilde{\boldsymbol{a}}^*, \tilde{\boldsymbol{\mu}}\right)$ if, and only if,*

$$\boldsymbol{P}^{\top}\left(\boldsymbol{a}^*, \boldsymbol{\mu}^*\right) \boldsymbol{\eta}\left(\boldsymbol{a}^*, \boldsymbol{\mu}^*, \tilde{\boldsymbol{a}}^*\right) = \boldsymbol{P}^{\top}\left(\boldsymbol{a}^*, \boldsymbol{\mu}^*\right) \boldsymbol{P}\left(\boldsymbol{a}^*, \boldsymbol{\mu}^*\right) \tilde{\boldsymbol{\mu}}^*.$$

In particular, for the sparsest solution $\left(\boldsymbol{a}^*, \tilde{\boldsymbol{a}}^*\right) = \mathbf{0}$, we have the following corollary.

Corollary 4.1. *For any $\boldsymbol{\mu}$, the vector $\tilde{\boldsymbol{\mu}}^* = \left(\ -\bar{y},\ \cdots,\ -\bar{y}\ \right)^{\top}$, with $\bar{y} = \dfrac{1}{|\mathcal{I}|} \sum\limits_{i \in \mathcal{I}} y_i$, minimizes* MSE $\left(\boldsymbol{a}^* = \mathbf{0}, \boldsymbol{\mu}^*, \tilde{\boldsymbol{a}}^* = \mathbf{0}, \tilde{\boldsymbol{\mu}}; \mathcal{I}\right)$*, and then the prediction is $\varphi_i\left(\boldsymbol{a}^* = \mathbf{0}, \boldsymbol{\mu}^*, \tilde{\boldsymbol{a}}^* = \mathbf{0}, \tilde{\boldsymbol{\mu}}^*\right) = \bar{y}$ for all $i \in \mathcal{I}$.*

Proof.

Observe that with $\left(\boldsymbol{a}^*, \tilde{\boldsymbol{a}}^*\right) = \mathbf{0}$, by construction, $P_{it}\left(\mathbf{0}, \boldsymbol{\mu}^*\right)$ is independent of i. Hence, $\boldsymbol{P}\left(\mathbf{0}, \boldsymbol{\mu}^*\right)$ is a matrix with all its rows identical to a vector $u\left(\boldsymbol{\mu}^*\right)$, with $\sum\limits_{t \in \tau_L} u_t\left(\boldsymbol{\mu}^*\right) = 1$.

Moreover, $\boldsymbol{\eta}\left(\mathbf{0}, \boldsymbol{\mu}^*, \mathbf{0}\right) = -\left(y_i\right)_{i \in \mathcal{I}}$, and thus, for $\boldsymbol{a}^* = \mathbf{0}$, $\boldsymbol{\mu}^*$ and $\tilde{\boldsymbol{a}}^* = \mathbf{0}$, the vector $\tilde{\boldsymbol{\mu}}^* = \left(\ -\bar{y},\ \ \cdots,\ \ -\bar{y}\ \right)^{\top}$ satisfies the system of linear equations in Proposition 4.1.

□

As stated, when λ^L and λ^G are taken large enough in Problem (4.1), the sparsest possible tree (with $\left(\boldsymbol{a}^*, \tilde{\boldsymbol{a}}^*\right) = \mathbf{0}$) is obtained though possibly not yielding the best prediction accuracy, since none of the predictor variables is used to fit the model. As observed in Figure 4.3, it turns out that the solution $\left(\boldsymbol{a}^*, \tilde{\boldsymbol{a}}^*\right) = \mathbf{0}$ is not only the limit case when λ^L and λ^G tend to infinity, but it is actually obtained already for finite values of them. This is shown in the following.

Proposition 4.2. *Let* $(\boldsymbol{a}^*, \tilde{\boldsymbol{a}}^*) = \mathbf{0}$, $\boldsymbol{\mu}^* \in \mathbb{R}^{|\tau_B|}$, *and* $\tilde{\boldsymbol{\mu}}^* = \begin{pmatrix} -\bar{y}, & \cdots, & -\bar{y} \end{pmatrix}^\top$. *Let* $\sigma \in [0, 1]$,

$$\begin{aligned} \lambda^l &= (1-\sigma) \max_{j=1,\dots,p} \left\| \nabla_{(\boldsymbol{a}_{j\cdot}, \tilde{\boldsymbol{a}}_{j\cdot})} \text{MSE}\left(\mathbf{0}, \boldsymbol{\mu}^*, \mathbf{0}, \tilde{\boldsymbol{\mu}}^*; \mathcal{I}\right) \right\|_\infty \text{ and} \\ \lambda^g &= \quad \sigma \quad \max_{j=1,\dots,p} \left\| \nabla_{(\boldsymbol{a}_{j\cdot}, \tilde{\boldsymbol{a}}_{j\cdot})} \text{MSE}\left(\mathbf{0}, \boldsymbol{\mu}^*, \mathbf{0}, \tilde{\boldsymbol{\mu}}^*; \mathcal{I}\right) \right\|_1 . \end{aligned}$$

Then, for any pair $\left(\lambda^L, \lambda^G\right)$ *such that* $\lambda^L \geq \lambda^l$ *and* $\lambda^G \geq \lambda^g$, $(\boldsymbol{a}^*, \boldsymbol{\mu}^*, \tilde{\boldsymbol{a}}^*, \tilde{\boldsymbol{\mu}}^*)$ *is a stationary point of Problem* (4.1).

Proof.

First, let us consider the necessary optimality conditions for $(\boldsymbol{a}^*, \tilde{\boldsymbol{a}}^*)$. For $\lambda^L \geq \lambda^l$ and $\lambda^G \geq \lambda^g$, we have, by construction, that

$$\begin{aligned} \lambda^L &\geq (1-\sigma) \left\| \nabla_{(\boldsymbol{a}_{j\cdot}, \tilde{\boldsymbol{a}}_{j\cdot})} \text{MSE}\left(\mathbf{0}, \boldsymbol{\mu}^*, \mathbf{0}, \tilde{\boldsymbol{\mu}}^*; \mathcal{I}\right) \right\|_\infty, \forall j = 1, \dots, p, \\ \lambda^G &\geq \quad \sigma \quad \left\| \nabla_{(\boldsymbol{a}_{j\cdot}, \tilde{\boldsymbol{a}}_{j\cdot})} \text{MSE}\left(\mathbf{0}, \boldsymbol{\mu}^*, \mathbf{0}, \tilde{\boldsymbol{\mu}}^*; \mathcal{I}\right) \right\|_1, \forall j = 1, \dots, p. \end{aligned}$$

Hence,

$$\begin{aligned} -(1-\sigma)\, \nabla_{(\boldsymbol{a}_{j\cdot}, \tilde{\boldsymbol{a}}_{j\cdot})} \text{MSE}\left(\boldsymbol{a}^*, \boldsymbol{\mu}^*, \tilde{\boldsymbol{a}}^*, \tilde{\boldsymbol{\mu}}^*; \mathcal{I}\right) &\in \lambda^L \partial_{(\boldsymbol{a}_{j\cdot}, \tilde{\boldsymbol{a}}_{j\cdot})} \left(\left\| (\boldsymbol{a}_{j\cdot}, \tilde{\boldsymbol{a}}_{j\cdot}) \right\|_1 \right) \Big|_{(\boldsymbol{a}_{j\cdot}, \tilde{\boldsymbol{a}}_{j\cdot}) = \mathbf{0}}, \forall j = 1, \dots, p, \\ -\sigma\, \nabla_{(\boldsymbol{a}_{j\cdot}, \tilde{\boldsymbol{a}}_{j\cdot})} \text{MSE}\left(\boldsymbol{a}^*, \boldsymbol{\mu}^*, \tilde{\boldsymbol{a}}^*, \tilde{\boldsymbol{\mu}}^*; \mathcal{I}\right) &\in \lambda^G \partial_{(\boldsymbol{a}_{j\cdot}, \tilde{\boldsymbol{a}}_{j\cdot})} \left(\left\| (\boldsymbol{a}_{j\cdot}, \tilde{\boldsymbol{a}}_{j\cdot}) \right\|_\infty \right) \Big|_{(\boldsymbol{a}_{j\cdot}, \tilde{\boldsymbol{a}}_{j\cdot}) = \mathbf{0}}, \forall j = 1, \dots, p, \end{aligned}$$

and thus, $\forall j = 1, \dots, p$,

$$-\nabla_{(\boldsymbol{a}_{j\cdot}, \tilde{\boldsymbol{a}}_{j\cdot})} \text{MSE}\left(\boldsymbol{a}^*, \boldsymbol{\mu}^*, \tilde{\boldsymbol{a}}^*, \tilde{\boldsymbol{\mu}}^*; \mathcal{I}\right) \in \lambda^L \partial_{(\boldsymbol{a}_{j\cdot}, \tilde{\boldsymbol{a}}_{j\cdot})} \left(\left\| (\boldsymbol{a}_{j\cdot}, \tilde{\boldsymbol{a}}_{j\cdot}) \right\|_1 \right) \Big|_{(\boldsymbol{a}_{j\cdot}, \tilde{\boldsymbol{a}}_{j\cdot}) = \mathbf{0}} + \lambda^G \partial_{(\boldsymbol{a}_{j\cdot}, \tilde{\boldsymbol{a}}_{j\cdot})} \left(\left\| (\boldsymbol{a}_{j\cdot}, \tilde{\boldsymbol{a}}_{j\cdot}) \right\|_\infty \right) \Big|_{(\boldsymbol{a}_{j\cdot}, \tilde{\boldsymbol{a}}_{j\cdot}) = \mathbf{0}},$$

having that,

$$-\nabla_{(\boldsymbol{a}, \tilde{\boldsymbol{a}})} \text{MSE}\left(\boldsymbol{a}^*, \boldsymbol{\mu}^*, \tilde{\boldsymbol{a}}^*, \tilde{\boldsymbol{\mu}}^*; \mathcal{I}\right) \in \lambda^L \partial_{(\boldsymbol{a}, \tilde{\boldsymbol{a}})} \left(\sum_{j=1}^p \left\| (\boldsymbol{a}_{j\cdot}, \tilde{\boldsymbol{a}}_{j\cdot}) \right\|_1 \right) \Big|_{(\boldsymbol{a}, \tilde{\boldsymbol{a}}) = (\boldsymbol{a}^*, \tilde{\boldsymbol{a}}^*)} + \lambda^G \partial_{(\boldsymbol{a}, \tilde{\boldsymbol{a}})} \left(\sum_{j=1}^p \left\| (\boldsymbol{a}_{j\cdot}, \tilde{\boldsymbol{a}}_{j\cdot}) \right\|_\infty \right) \Big|_{(\boldsymbol{a}, \tilde{\boldsymbol{a}}) = (\boldsymbol{a}^*, \tilde{\boldsymbol{a}}^*)},$$

i.e.:

$$\mathbf{0} \in \partial_{(\boldsymbol{a}, \tilde{\boldsymbol{a}})} \left(\text{MSE}\left(\boldsymbol{a}^*, \boldsymbol{\mu}^*, \tilde{\boldsymbol{a}}^*, \tilde{\boldsymbol{\mu}}^*; \mathcal{I}\right) + \lambda^L \sum_{j=1}^p \left\| (\boldsymbol{a}_{j\cdot}, \tilde{\boldsymbol{a}}_{j\cdot}) \right\|_1 + \lambda^G \sum_{j=1}^p \left\| (\boldsymbol{a}_{j\cdot}, \tilde{\boldsymbol{a}}_{j\cdot}) \right\|_\infty \right) \Big|_{(\boldsymbol{a}, \tilde{\boldsymbol{a}}) = (\boldsymbol{a}^*, \tilde{\boldsymbol{a}}^*)}.$$

For $(\boldsymbol{a}^*, \tilde{\boldsymbol{a}}^*) = \mathbf{0}$, Corollary 4.1 shows that the chosen $\tilde{\boldsymbol{\mu}}^*$ minimizes MSE, and is thus optimal for Problem (4.1). As a consequence, $\tilde{\boldsymbol{\mu}}^*$ satisfies the necessary optimality conditions.

Finally, let us analyze the optimality conditions for $\boldsymbol{\mu} = \boldsymbol{\mu}^*$. Observe that

$$\nabla_{\boldsymbol{\mu}}\text{MSE}\left(\boldsymbol{a}, \boldsymbol{\mu}, \tilde{\boldsymbol{a}}, \tilde{\boldsymbol{\mu}}\right) = \frac{1}{|\mathcal{I}|}\sum_{i\in\mathcal{I}} 2\left(\varphi_i\left(\boldsymbol{a}, \boldsymbol{\mu}, \tilde{\boldsymbol{a}}, \tilde{\boldsymbol{\mu}}\right) - y_i\right)\nabla_{\boldsymbol{\mu}}\varphi_i\left(\boldsymbol{a}, \boldsymbol{\mu}, \tilde{\boldsymbol{a}}, \tilde{\boldsymbol{\mu}}\right).$$

Since $(\boldsymbol{a}^*, \tilde{\boldsymbol{a}}^*) = \mathbf{0}$, $\nabla_{\boldsymbol{\mu}}\varphi_i\left(\boldsymbol{a}^*, \boldsymbol{\mu}^*, \tilde{\boldsymbol{a}}^*, \tilde{\boldsymbol{\mu}}^*\right)$ does not depend on $i \in \mathcal{I}$, say $\nabla_{\boldsymbol{\mu}}\varphi_i\left(\boldsymbol{a}^*, \boldsymbol{\mu}^*, \tilde{\boldsymbol{a}}^*, \tilde{\boldsymbol{\mu}}^*\right) = v$ for all $i \in \mathcal{I}$. Hence,

$$\nabla_{\boldsymbol{\mu}}\text{MSE}\left(\boldsymbol{a}^*, \boldsymbol{\mu}^*, \tilde{\boldsymbol{a}}^*, \tilde{\boldsymbol{\mu}}^*\right) = \frac{1}{|\mathcal{I}|}\sum_{i\in\mathcal{I}} 2\left(\varphi_i\left(\boldsymbol{a}^*, \boldsymbol{\mu}^*, \tilde{\boldsymbol{a}}^*, \tilde{\boldsymbol{\mu}}^*\right) - y_i\right)v.$$

By Corollary 4.1, $\varphi_i\left(\boldsymbol{a}^*, \boldsymbol{\mu}^*, \tilde{\boldsymbol{a}}^*, \tilde{\boldsymbol{\mu}}^*\right) = \bar{y}$, and thus $\sum_{i\in\mathcal{I}}\left(\varphi_i\left(\boldsymbol{a}^*, \boldsymbol{\mu}^*, \tilde{\boldsymbol{a}}^*, \tilde{\boldsymbol{\mu}}^*\right) - y_i\right) = 0$, implying

$$\nabla_{\boldsymbol{\mu}}\text{MSE}\left(\boldsymbol{a}^*, \boldsymbol{\mu}^*, \tilde{\boldsymbol{a}}^*, \tilde{\boldsymbol{\mu}}^*\right) = 0,$$

and the desired result follows.

□

4.3 Computational experiments

The aim of this section is to illustrate the performance of our sparse optimal randomized regression tree (S-ORRT) using both real-world and synthetic data sets. Section 4.3.1 gives details on the procedure followed to test our approach in the real-world data sets. In Section 4.3.2, we discuss the prediction accuracy of S-ORRT, against several benchmark regression methods. In Section 4.3.3, we illustrate our ability to trade in some of the prediction accuracy of S-ORRT for a gain in local and global sparsity. Finally, in Section 4.3.4 we illustrate the scalability of S-ORRT in terms of the number of individuals in the training sample, using a synthetic data set.

4.3.1 Setup

A collection of well-known real-world data sets from the UCI Machine Learning Repository [Lichman, 2013] has been chosen. Table 4.3 lists their names, the abbreviations used throughout this section to refer to them, together with their number of observations and predictor variables.

Each data set has been randomly split into two subsets: the training subset (75%) and the test subset (25%). The corresponding tree model is built on the training subset and, then, three performance criteria, namely prediction accuracy, local and global sparsity, are assessed. The prediction accuracy is evaluated by the out-of-sample R-squared (R^2) in the test subset:

$$R^2 = 1 - \frac{\text{MSE}_{\text{test}}}{\text{V}_{\text{test}}},$$

Table 4.3: Information about the real-world data sets considered.

Data set	Abbreviation	N	p
Boston-housing	BH	506	13
Red-wine	RW	1599	11
White-wine	WW	4898	11
Parkinson-motor	PM	5874	16
Parkinson-total	PT	5874	16
Ailerons	A	7153	40
Cpu-act	CA	8192	21
Cart-artificial	CAr	40768	10
Friedman-artificial	FA	40768	10

where MSE_{test} is the mean squared error obtained by the regression method in the test subset and V_{test} is the variance of the actual response vector in the test subset. The higher the R^2, the better the model in terms of prediction accuracy.

The control of local and global sparsity is one of the key features of S-ORRT, as has been pointed out previously. Local sparsity, δ^L, is measured as the average percentage of predictor variables not used per node:

$$\delta^L = \frac{1}{p}\sum_{j=1}^{p}\frac{|\{a_{jt}=0,\ t\in\tau_B\}|+|\{\tilde{a}_{jt}=0,\ t\in\tau_L\}|}{|\tau_B|+|\tau_L|}\times 100.$$

Global sparsity, δ^G, is measured as the percentage of predictor variables not used at any of the nodes, i.e., across the whole tree:

$$\delta^G = \frac{|\{(\boldsymbol{a}_{j\cdot},\tilde{\boldsymbol{a}}_{j\cdot})=\mathbf{0},\ j=1,\ldots,p\}|}{p}\times 100.$$

The higher δ^L and δ^G, the better the model in terms of local and global sparsity, respectively.

The training/testing procedure has been repeated ten times. The results shown in Table 4.4 and Figure 4.3 represent the average of such ten runs for the above-mentioned performance criteria.

The logistic CDF has been chosen for our experiments:

$$F(\cdot) = \frac{1}{1+\exp(-(\cdot)\gamma)},$$

with a large value of γ, namely, $\gamma = 512$. We will illustrate that this small level of randomization is enough for obtaining good results.

The S-ORRT smooth formulation (4.3)-(4.8) has been implemented using the `scipy.optimize` package [Virtanen et al., 2020] in Python 3.7 [Python Core Team, 2015]. As a solver, we have used the SLSQP method [Kraft, 1988] that allows one to use gradient information. The predictor variables have been previously normalized to the $[0,1]$ interval, and the decision variables

$\boldsymbol{a}$, $\boldsymbol{\mu}$, $\tilde{\boldsymbol{a}}$ and $\tilde{\boldsymbol{\mu}}$ have been restricted to the $[-1, 1]$ interval. Our experiments have been conducted on a PC, with an Intel® Core™ i7-9700 CPU 3.00 GHz processor (8 cores) and 64 GB RAM. The operating system is 64 bits.

4.3.2 Comparison of prediction performance

In this section, we focus on illustrating the prediction accuracy of all the methods tested on the real-world data sets. S-ORRT at depths $D = 1,\ 2$ and 3 with $\lambda^L = \lambda^G = 0$ is compared against three types of benchmark regression methods. The first type corresponds to standard regression methods, such as CART, the classic approach to build decision trees, with no restrictions on depth, and OLS. The second type is the leader regression method in terms of sparsity, LASSO. Finally, in the third type we have two sophisticated tree-based regression methods competitive in terms of prediction accuracy, such as ORT-H LS in Dunn [2018], a Mathematical Programming based approach that employs a local-search heuristic for building oblique trees with linear predictions at maximum depth $D = 10$; and Random Forest (RF), an ensemble of CARTs using a boostrap aggregating scheme. Table 4.4 presents the average out-of-sample prediction accuracy R^2, while in parenthesis we show how the method ranks in terms of its prediction accuracy. For a given data set, a rank of "1" indicates that the method is the best in terms of out-of-sample R^2 while a rank of "8" indicates that the method performed the worst. The average R^2 and rank of each method across all data sets are found at the bottom of the table.

For S-ORRT, we have followed a multistart approach, where the process is repeated 1000 times starting from different random initial solutions. For a given initial solution, the computing time taken by the S-ORRT typically ranges from 0.01 seconds (in BH for $D = 1$) to 2.08 seconds (in A and FA for $D = 3$). The default parameter setting in `rpart` [Therneau et al., 2015], `glmnet` [Friedman et al., 2010] and `randomForest` [Liaw and Wiener, 2002] R packages have been used for running CART, OLS and LASSO, and RF, respectively. For ORT-H LS, the results are taken from Dunn [2018], since open-source implementations were not available.

We start discussing the results for our S-ORRT with depth $D = 3$. S-ORRT outperforms CART, OLS and LASSO, yielding increases in the R^2 up to 34 percentage points (p.p.) with respect to CART, and up to 24 p.p. with respect to OLS and LASSO, both with comparable performance. Regarding ORT-H LS, S-ORRT presents an average prediction accuracy 2 p.p. lower, however S-ORRT manages to be comparable in CAr and outperform in RW by 6 p.p.

Finally, although RF reports the best overall performance across all the methods, S-ORRT is comparable to RF in A and CAr, while S-ORRT has the best prediction accuracy in FA.

With depth $D = 2$, the conclusions for S-ORRT are similar to those obtained using depth $D = 3$. With depth $D = 1$, S-ORRT still manages to be powerful in some data sets, despite the low complexity of the model. S-ORRT outperforms CART, OLS and LASSO in six of the data sets considered, all except for BH, RW and CA. ORT-H LS generally outperforms S-ORRT

Table 4.4: Comparison between S-ORRT with $\lambda^L = \lambda^G = 0$, CART, OLS, LASSO, ORT-H LS and RF in terms of out-of-sample R-squared, R^2, on real-world data sets in Table 5.1.

Data set	Out-of-sample average R^2							
	CART	OLS	LASSO	ORT-H LS	RF	S-ORRT $D=1$	S-ORRT $D=2$	S-ORRT $D=3$
BH	0.7416(5)	0.7391(7)	0.7401(6)	0.8040(2)	**0.8759**(1)	0.5987(8)	0.7931(3)	0.7785(4)
RW	0.3055(7)	0.3619(3)	0.3605(5)	0.3040(8)	**0.4874**(1)	0.3482(6)	0.3730(2)	0.3613(4)
WW	0.2539(8)	0.2714(6)	0.2699(7)	0.3490(2)	**0.5196**(1)	0.3121(5)	0.3291(4)	0.3337(3)
PM	0.1020(6)	0.0878(8)	0.0900(7)	0.2810(2)	**0.3426**(1)	0.1878(5)	0.2121(4)	0.2400(3)
PT	0.1294(6)	0.0849(8)	0.0863(7)	0.3160(2)	**0.3545**(1)	0.1724(5)	0.1965(4)	0.2445(3)
A	0.6466(8)	0.8167(7)	0.8173(6)	**0.8360**(1)	0.8211(3)	0.8207(5)	0.8288(2)	0.8211(3)
CA	0.9324(5)	0.7272(8)	0.7273(7)	**0.9840**(1)	0.9829(2)	0.8282(6)	0.9535(4)	0.9540(3)
CAr	0.8771(6)	0.7045(7)	0.7045(7)	**0.9480**(1)	0.9425(5)	**0.9480**(1)	**0.9480**(1)	**0.9480**(1)
FA	0.6058(8)	0.7222(7)	0.7223(6)	**0.9560**(1)	0.9245(4)	0.8493(5)	0.9501(3)	0.9505(2)
Average	0.5105(6.5)	0.5017(6.7)	0.5020(6.4)	0.6420(2.2)	**0.6946**(2.1)	0.5628(5.1)	0.6205(3.0)	0.6257(2.8)

at depth $D = 1$, with the exception of CAr, where S-ORRT is comparable, and RW, where S-ORRT is superior in 4 p.p. With respect to RF, S-ORRT is outperformed in general, but has a comparable prediction accuracy in A and CAr.

In summary, these numerical results illustrate that, in terms of prediction accuracy, S-ORRT with $D = 2, 3$ outperforms the standard benchmark regression methods (CART and OLS) and the benchmark regression method in sparsity (LASSO). Regarding more sophisticated tree-based approaches, ORT-H LS and RF show slightly better prediction accuracies, although S-ORRT is competitive in some data sets. Unlike CART, ORT-H LS and RF, our approach has a direct control on global desirable properties such as sparsity, cost-sensitivity and fairness.

4.3.3 Prediction accuracy and sparsity tradeoff

The aim of this section is to illustrate that, in contrast to sophisticated tree-based regression methods that rely on greedy or local-search approaches, such as RF and ORT-H LS, our S-ORRT is able to trade in some of its prediction accuracy for a gain in local and global sparsity. For the sake of conciseness, we illustrate this in the Ailerons data set. We have solved Problem (4.3)-(4.8) with depth $D = 1$ for the sparsity parameters λ^L and λ^G in a grid. We have taken the grid $\{0\} \cup \{2^r,\ -12 \leq r \leq 5,\ r \in \mathbb{Z}\}$, normalized by the number of predictor variables, and in the case of λ^L by the number of nodes too. We start solving the optimization problem with $\left(\lambda^L, \lambda^G\right) = (0, 0)$. We continue with $\lambda^L = 0$ but for larger values of λ^G. Once all $\left(0, \lambda^G\right)$ are executed, we start the process all over again with the next value of λ^L in the grid. The solutions found to Problem (4.3)-(4.8) for fixed $\left(\lambda^L, \lambda^G\right)$ are given as initial solutions to the next problem to be solved in the grid.

Figure 4.3 illustrates these results by means of three heatmaps: one for the prediction accuracy, R^2, another one for the local sparsity, δ^L, and the final one for the global sparsity, δ^G. The color bar of each heatmap goes from light green to dark blue, the latter indicating the best (maximum) R^2, δ^L or δ^G achieved, respectively. By definition, the sparsest tree is obtained

for large of values of λ^L, λ^G. We can observe that the best rates of prediction accuracy are not only achieved for $\left(\lambda^L, \lambda^G\right) = (0, 0)$. Clearly, the R^2 remains almost constant for pair of values $\left(\lambda^L, \lambda^G\right)$ that verify $\lambda^L \leq \bar{\lambda}^L$ and $\lambda^G \leq \bar{\lambda}^G$ where $\left(\bar{\lambda}^L, \bar{\lambda}^G\right) = \left(\frac{2^{-2}}{120}, \frac{2^{-4}}{40}\right)$. In this range, where we have the best prediction accuracy, we can dramatically enhance both the local and the global sparsity. Indeed, the local sparsity improves from 1% to 84% and the global sparsity from 0% to 52%. For larger values of λ^L and λ^G, our S-ORRT keeps improving sparsity but, in this case, at the cost of diminishing R^2.

Figure 4.4 reflects, against CART and RF, our ability to trade off prediction accuracy and global sparsity in Ailerons data set. The value of both performance measures are drawn through blue points for every pair of the sparsity regularization parameters $\left(\lambda^L, \lambda^G\right)$ considered in the S-ORRT construction. The values for CART and RF are depicted with a green diamond and an orange cross, respectively. It can be seen that S-ORRT outperforms CART in both prediction accuracy and global sparsity for several pairs of $\left(\lambda^L, \lambda^G\right)$. With respect to RF, S-ORRT is comparable in terms of prediction accuracy, while improving global sparsity in 50%.

4.3.4 Scalability depending on the number of individuals: a simulation study

In this section we illustrate that S-ORRT scales up well with the size of the training sample N. To this aim, we measure the computing time taken by S-ORRT to reach a solution with 30% improvement on the mean squared error of CART.

We have designed a synthetic data set with $p = 25$ predictor variables and N taking values in $\left\{10^5, 10^6, 10^7\right\}$. The first two predictor variables, X_1 and X_2, define two balanced groups of individuals. They are generated following bivariate normal distributions, $\mathcal{N}\left(\boldsymbol{\eta}_k, \boldsymbol{\Sigma}_k\right), k = 1, 2$. $\boldsymbol{\eta}_1 = (0.50, 0.75)^\top$ and $\boldsymbol{\Sigma}_1 = \begin{pmatrix} 0.005 & 0 \\ 0 & 0.00375 \end{pmatrix}$ for Group 1, and $\boldsymbol{\eta}_2 = (0.25, 0.50)^\top$ and $\boldsymbol{\Sigma}_2 = \boldsymbol{\Sigma}_1$ for Group 2. The remaining 23 predictor variables were generated following a uniform distribution, $\mathcal{U}(0, 1)$. The response variable for Group 1 is equal to $Y = X_3 + 2X_4 + 5 + \varepsilon$ while for Group 2 is equal to $Y = -X_5 - 2X_6 - 5 + \varepsilon$, where $\varepsilon \sim \mathcal{N}(0, 0.5)$. Thus, $X_7, \ldots, X_{25}$ have no impact in the response variable. An S-ORRT tree of depth $D = 1$ with $\lambda^L = \lambda^G = 0$ is built. We feed Problem (4.3)-(4.8) with an initial solution, obtained from a heuristic procedure based on the RF variable importance measure. That is, in a first step we solve Problem (4.3)-(4.8) with a multistart approach in which the predictor variables with low RF variable importance, namely $X_j,\ j = 3, \ldots, 25$, do not play a role. This heuristic solution is given as the initial one to solve Problem (4.3)-(4.8) with the whole set of predictor variables. The procedure has been repeated 10 times and average results are presented. Figure 4.5 shows, as a function of N, the total computing time spent for the whole procedure. Both axes are on logarithmic scale. We can see that for this simulation study, the computing times have a linear trend with respect to the number of individuals.

4.4 Conclusions

In this chapter, we have adapted the continuous optimization-based approach to build classification trees previously presented in Chapters 2 and 3 to consider regression trees. Local explanations on the continuous predictor space can be derived thanks to the smoothness of the predictions with respect to the features. Unlike CART and RF, we can directly model desirable properties such as sparsity, cost-sensitivity and fairness. The computational experience reported shows that our method outperforms CART, as well as OLS and LASSO, in terms of prediction accuracy. Finally, we show that our approach scales up well when the size of the training sample grows.

Several extensions to our approach are attractive. First, the linear prediction made at each leaf node can easily be extended to a non-linear one. This would be obtained by simply replacing the linear functions φ_{it} with other functions, such as those in a Generalized Additive Model. Second, in many applications, processes are continuously monitored over time, yielding data of funcional nature. Adapting our approach to this kind of data is an interesting question, which is addressed in Chapter 5.

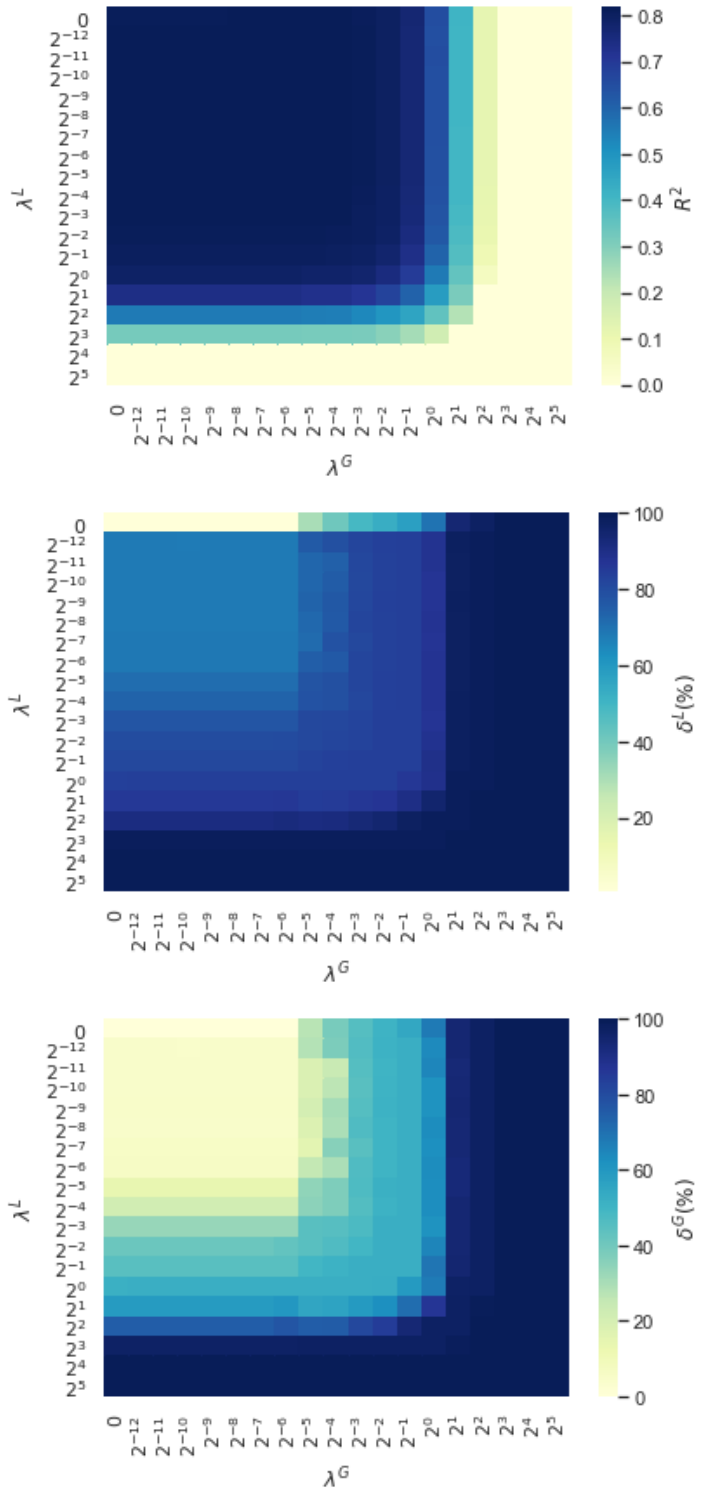


Figure 4.3: Heatmaps representation, for Ailerons data set, of the average R-squared obtained, R^2, the average percentage of predictor variables not used per node, δ^L, and the average percentage of predictor variables not used per tree, δ^G, respectively, as a function of the grid of the sparsity regularization parameters, λ^L and λ^G, considered in the S-ORRT construction.

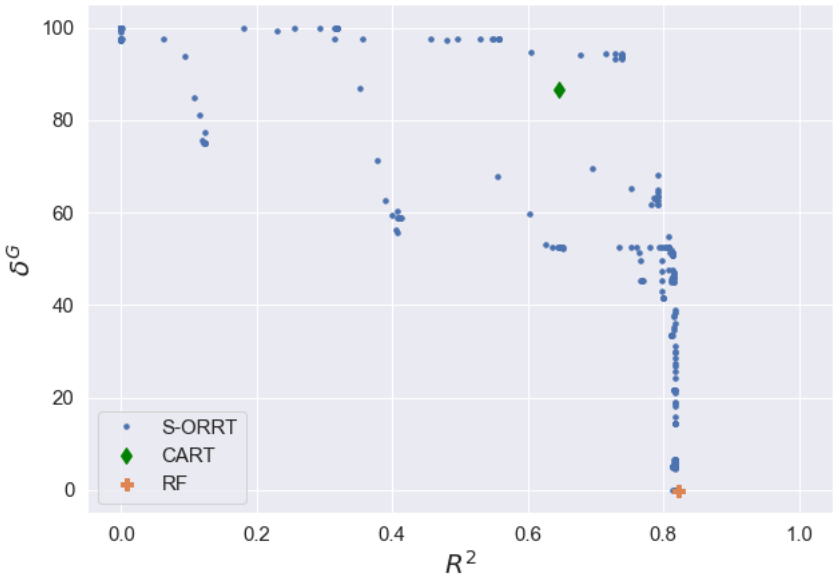


Figure 4.4: Scatterplot representation, for Ailerons data set, of the average R-squared obtained, R^2, and the average percentage of predictor variables not used per tree, δ^G. Blue points refer to the solution of every pair of the sparsity regularization parameters (λ^L, λ^G) considered in the S-ORRT construction; the green diamond, to CART solution; and the orange cross, to RF solution.

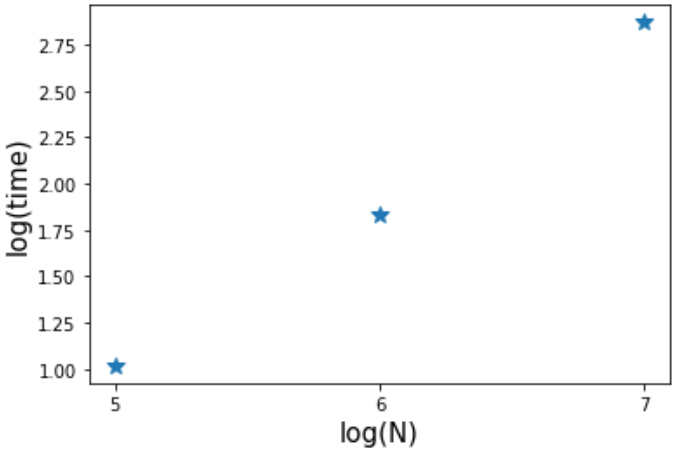


Figure 4.5: Scalability of S-ORRT in logarithmic scale, where the computing time is measured in seconds as a function of N varying in $\{10^5, 10^6, 10^7\}$

Chapter 5

Sparse Optimal Randomized Regression Trees in Functional Data Analysis

In this chapter, we tailor sparse optimal randomized regression trees in Chapter 4 to handle functional predictor variables, called hereafter S-ORRT-FD. In principle, one could directly use the approach in Chapter 4 after discretizing the functions and converting them to vectors. Yet, we would face the curse of dimensionality as well as the impossibility to fully exploit the intrinsic characteristics of functional data. Whilst fitting S-ORRT-FD, the detection of a reduced number of intervals that are critical for prediction, as well as the control of their length, is performed. In the degenerate case of intervals with length equal to zero, critical instants would be detected instead. Similarly to previous chapters, local and global sparsities can be modeled through the inclusion of LASSO-type regularization terms over the coefficients associated to functional predictor variables. The resulting optimization problem is formulated as a nonlinear continuous and smooth model with linear constraints. We illustrate the performance of our approach on real-world and synthetic data sets.

The chapter is organized as follows. In Section 5.1, we introduce the S-ORRT-FD and its mathematical formulation. Our computational experience is reported in Section 5.2. We illustrate that S-ORRT-FD is competitive with state-of-the-art regression benchmarks. Moreover, we show our ability to trade off prediction accuracy and sparsity, in the form of controlling the number of critical intervals and the proportion of the curves to be used for prediction. Finally, conclusions and possible extensions are provided in Section 5.3.

5.1 Sparse Optimal Randomized Regression Trees for Functional Data

5.1.1 Introduction

Let $\mathcal{I}$ be a given set of N individuals. Each individual $i \in \mathcal{I}$ is represented by a pair $(\boldsymbol{x}_i, y_i)$. The first element $\boldsymbol{x}_i \in \mathcal{F}^p$ is composed by p functional predictor variables, i.e., $\boldsymbol{x}_i = (x_{i1}(\cdot), \ldots, x_{ip}(\cdot))$, where $x_{ij}(\cdot) : [\underline{s}, \overline{s}] \rightarrow \mathbb{R},\ j = 1, \ldots, p$, belongs to the set $\mathcal{F}$ of Riemann integrable functions in the interval $[\underline{s}, \overline{s}]$. Note that numerical predictor variables in previous chapters are included by defining them as constant functions. The second element of the pair, $y_i \in \mathbb{R}$, indicates the value of the response variable.

Sparse Optimal Randomized Regression Trees for Functional Data (S-ORRT-FD) is a generalization of the approach introduced in Chapter 4 for vectorial data. An optimal binary regression tree of a given depth D is to be built, obtained by controlling simultaneously prediction accuracy and some kind of sparsity. Figure 5.1 shows the structure of an S-ORRT-FD of depth $D = 2$. S-ORRT-FDs are modeled by means of a Non-Linear Continuous Optimization (NLCO) formulation. Oblique cuts are implemented at the set of branch nodes τ_B. Linear predictions are associated to the set of leaf nodes τ_L. The usual deterministic yes/no rule at each branch node is replaced by a probabilistic decision rule, induced by a univariate continuously differentiable cumulative density function (CDF) F, evaluated over the vector of functional

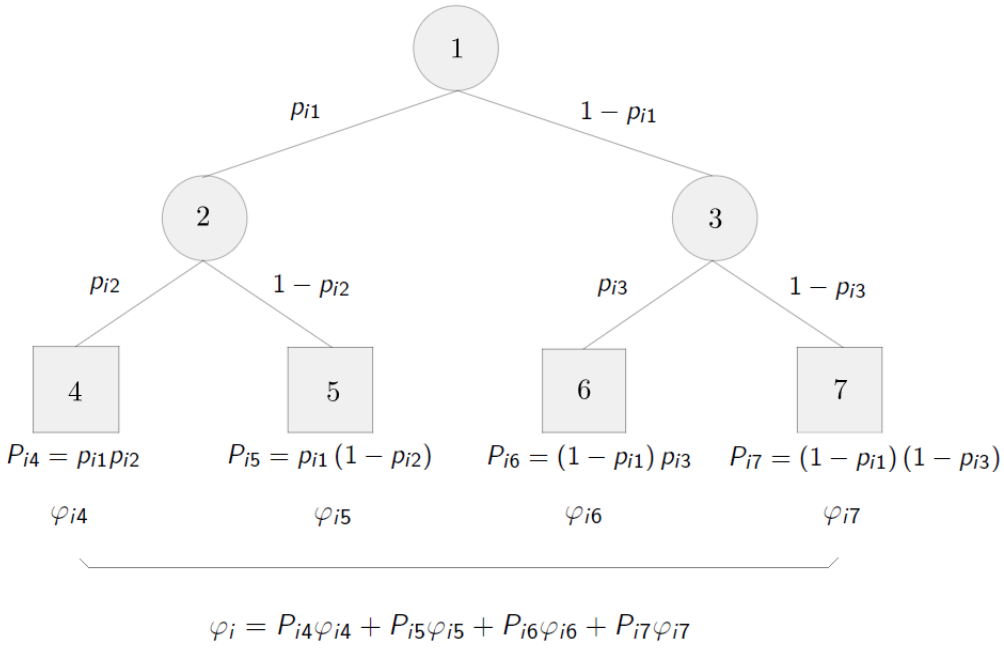


Figure 5.1: Sparse Optimal Randomized Regression Tree of depth $D = 2$ for Functional Data with $\{p_{it}\}_{t=1,2,3}$, $\{P_{it}\}_{t=4,5,6,7}$ and $\{\varphi_{it}\}_{t=4,5,6,7}$ defined in Equations (5.2), (5.3) and (5.4), respectively.

predictor variables. With this, we have the probability of each individual in the sample falling into every leaf node, that will represent the weights that the linear predictions will have in the estimated outcome value.

In order to model oblique splits at branch nodes $t \in \tau_B$, we need to define, for each functional predictor variable $j = 1, \ldots, p$ and each $t \in \tau_B$, the functional decision variables $a_{jt}(\cdot) : [\underline{s}, \overline{s}] \rightarrow \mathbb{R}$ that denote the coefficient functions, as well as the decision variables $\mu_t,\ t \in \tau_B$ as intercepts. Then, the probability of individual $i = 1, \ldots, N$ going down the left branch at branch node $t \in \tau_B$ would be defined by:

$$F\left(\frac{1}{p}\sum_{j=1}^{p}\int_{\underline{s}}^{\overline{s}} a_{jt}(s)x_{ij}(s)ds \ - \ \mu_t\right). \tag{5.1}$$

It may happen that simply using a finite number of intervals [Blanquero et al., 2020a] in the domain of the functional predictor variables is sufficient to produce an accurate analysis of the whole curve, which is in turn more interpretable and saves both monitoring and storage costs. With the aim of detecting critical intervals for prediction, we will assume that $a_{jt}(\cdot), j = 1, \ldots, p,\ t \in \tau_B$, are piecewise constant functions. Let H denote the number of pieces where constants are different from zero, or critical intervals, and $a_{jth} \in \mathbb{R}, j = 1, \ldots, p, t \in \tau_B, h = 1, \ldots, H$, the decision variables representing the coefficient of functional variable j at branch

node t in the critical interval h. New decision variables are to be defined to represent the lower and upper ends of such intervals, namely $[s_{2h-1}, s_{2h}], h = 1, \ldots, H$, where $s_h,\ h = 1, \ldots, 2H$, should belong to the interval $[\underline{s}, \overline{s}]$. In this way, each functional decision variable turns into

$$a_{jt}(s) = \begin{cases} \dfrac{a_{jth}}{s_{2h} - s_{2h-1}}, & \text{if } s \in [s_{2h-1}, s_{2h}], h = 1, \ldots, H \\ 0, & \text{otherwise.} \end{cases},$$

where coefficients a_{jth} have been scaled according to the length of their corresponding critical interval. See Figure 5.2 for an example of a piecewise constant function with $H = 3$ critical intervals.

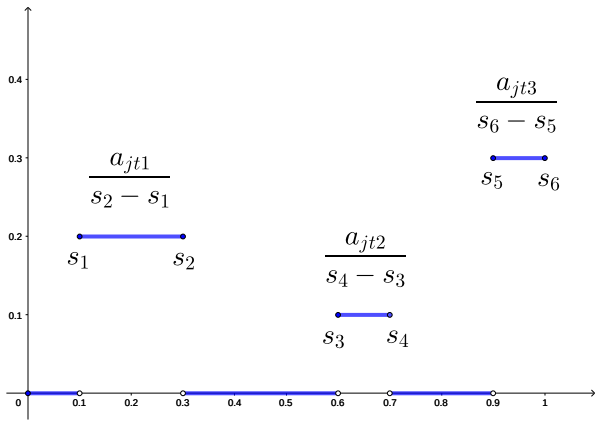


Figure 5.2: Example of a piecewise constant function with $H = 3$ critical intervals and domain $[0, 1]$

We have to add the constraints $s_h \leq s_{h+1},\ j = 1, \ldots, p,\ h = 1, \ldots, 2H - 1$, to ensure that intervals are ordered and do not overlap.

Let $\boldsymbol{a}$, $\boldsymbol{\mu}$ and $\boldsymbol{s}$ denote the $p \times |\tau_B| \times H$-matrix, the $|\tau_B|$-vector and the $2H$-vector of the coefficients $\boldsymbol{a} = (a_{jth})_{j=1,\ldots,p,\ t\in\tau_B,\ h=1,\ldots,H}$, $\boldsymbol{\mu} = (\mu_t)_{t\in\tau_B}$, and $\boldsymbol{s} = (s_h)_{h=1,\ldots,2H}$, respectively. Then, the probability of individual $i = 1, \ldots, N$ going down the left branch at branch node $t \in \tau_B$, defined in Equation (5.1), turns into:

$$p_{it}\left(\boldsymbol{a}_{\cdot t\cdot}, \mu_t, \boldsymbol{s}\right) = F\left(\frac{1}{p}\sum_{j=1}^{p}\sum_{h=1}^{H}\frac{a_{jth}}{s_{2h} - s_{2h-1}}\int_{s_{2h-1}}^{s_{2h}} x_{ij}(s)ds \ - \ \mu_t\right). \tag{5.2}$$

The expression $\boldsymbol{a}_{\cdot t\cdot}$ denotes the $p \times H$-matrix of the coefficients in $\boldsymbol{a}$ related to branch node t.

The probability of individual $i = 1, \ldots, N$ falling into leaf node $t \in \tau_L$ is:

$$P_{it}\left(\boldsymbol{a}, \boldsymbol{\mu}, \boldsymbol{s}\right) = \prod_{t_l\in\mathcal{N}_L(t)} p_{it_l}\left(\boldsymbol{a}_{\cdot t_l\cdot}, \mu_{t_l}, \boldsymbol{s}\right) \prod_{t_r\in\mathcal{N}_R(t)} \left(1 - p_{it_r}\left(\boldsymbol{a}_{\cdot t_r\cdot}, \mu_{t_r}, \boldsymbol{s}\right)\right), \tag{5.3}$$

where $N_L(t)$ and $N_R(t)$ are the sets of ancestor nodes of leaf node t whose left and right branch, respectively, takes part in the path from the root node to leaf node t, $t \in \tau_L$.

Similarly to oblique cuts, real decision variables $\tilde{\boldsymbol{a}} = (\tilde{a}_{jtk})_{j=1,\ldots,p,\ t\in\tau_L,\ h=1,\ldots,H}$ and $\tilde{\boldsymbol{\mu}} = (\tilde{\mu}_t)_{t\in\tau_L}$ are to be defined in order to provide linear predictions at leaf nodes $t \in \tau_L$ for each individual $i = 1, \ldots, N$:

$$\varphi_{it}\left(\tilde{\boldsymbol{a}}_{\cdot t\cdot}, \tilde{\mu}_t, \boldsymbol{s}\right) = \sum_{j=1}^{p}\sum_{h=1}^{H} \frac{\tilde{a}_{jth}}{s_{2h} - s_{2h-1}} \int_{s_{2h-1}}^{s_{2h}} x_{ij}(s)ds \ - \ \mu_t, \tag{5.4}$$

where coefficients $\tilde{a}_{jth}$ have been scaled according to the length of their corresponding critical interval. The expression $\tilde{\boldsymbol{a}}_{\cdot t\cdot}$ denotes the $p \times H$-matrix of the coefficients in $\tilde{\boldsymbol{a}}$ related to leaf node t.

5.1.2 The formulation

With these parameters and decision variables, the S-ORRT-FD reads as the following NLCO problem with linear constraints:

$$\min_{\substack{(\boldsymbol{a},\boldsymbol{\mu})\in\mathbb{R}^{(pH+1)|\tau_B|}\\ (\tilde{\boldsymbol{a}},\tilde{\boldsymbol{\mu}})\in\mathbb{R}^{(pH+1)|\tau_L|}\\ \boldsymbol{s}\in[\underline{s},\overline{s}]^{2H}}} \quad \frac{1}{N}\sum_{i=1}^{N}\Big(\sum_{t\in\tau_L} P_{it}\left(\boldsymbol{a},\boldsymbol{\mu},\boldsymbol{s}\right)\varphi_{it}\left(\tilde{\boldsymbol{a}}_{\cdot t\cdot},\tilde{\mu}_t,\boldsymbol{s}\right) - y_i\Big)^2 \tag{5.5}$$

$$+ \ \lambda^G \sum_{j=1}^{p} \left\|\left(\boldsymbol{a}_{j\cdot\cdot}, \tilde{\boldsymbol{a}}_{j\cdot\cdot}\right)\right\|_\infty \tag{5.6}$$

$$+ \ \lambda^{\text{length}} \sum_{h=1}^{H} \left(s_{2h} - s_{2h-1}\right) \tag{5.7}$$

$$\text{s.t.} \qquad s_h \leq s_{h+1},\ h = 1, \ldots, 2H-1, \tag{5.8}$$

where the expressions $\boldsymbol{a}_{j\cdot\cdot}$ and $\tilde{\boldsymbol{a}}_{j\cdot\cdot}$ denote the $|\tau_B| \times H$- and the $|\tau_L| \times H$-matrices of the coefficients in $\boldsymbol{a}$ and $\tilde{\boldsymbol{a}}$ relating to funcional predictor variable j, respectively.

The first term, prediction accuracy, is equal to the mean squared error (MSE) over the training sample between the actual response values and the predictions returned by S-ORRT-FD, which are weighted averages of the linear predictions along the different leaf nodes, where the weights in such average depend on the probability of individual i falling into such leaf node. Note that the number of critical intervals allowed for prediction H is fixed in advance.

The second term, parametrized by λ^G, addresses global sparsity [Febrero-Bande et al., 2019], which is modeled by the inclusion of a penalization term that controls whether a given functional predictor variable is ever used across the whole tree. Recall that each functional predictor variable may appear at either branch (in the oblique cuts) and leaf (in the linear predictions) nodes. Then, the ℓ_∞-norm is used as a group penalty function, by forcing all the

coefficients linked to the same functional predictor variable to be shrunk simultaneously along all branch and leaf nodes.

It may occur that several functional predictor variables are related; for instance, one funcional characteristic could appear together with its derivatives. In such cases, the corresponding ℓ_∞-norm term will comprise the coeffcients of all of these series in order to force that all of them shrink to zero at the same time.

The third term, parametrized by λ^{length}, controls the proportion of the curves to be used in the prediction returned by S-ORRT-FD. This is done by penalizing the length of the critical intervals.

Remark 5.1. *Note that if the length of a critical interval $[s_{2h-1}, s_{2h}]$ tends to zero, a critical instant [Aneiros and Vieu, 2014, 2016; Berrendero et al., 2019; Kong et al., 2016] would be detected instead, thanks to the integral form of the mean value theorem. This situation occurs for high enough values of λ^{length}.*

As in previous chapters, local sparsity can also be controlled by penalizing the ℓ_1-norm of the coefficients of the predictor variables used in the cuts along the tree, that is, including in the objective the following term:

$$\lambda^L \sum_{j=1}^{p} \left\| (\boldsymbol{a}_{j\cdot\cdot}, \tilde{\boldsymbol{a}}_{j\cdot\cdot}) \right\|_1 ,$$

where λ^L is the local sparsity parameter.

5.1.3 A smooth reformulation

Problem (5.5)-(5.8) is non-smooth due to the ℓ_∞-norm appearing in the objective function. Recall that F is assumed to be continuously differentiable, therefore the MSE inherits smoothness. By rewriting such regularization term using new decision variables, we can formulate S-ORRT-FD as a smooth problem, thus solvable with standard continuous optimization solvers, as done in our computational section. Let $\boldsymbol{\beta} = (\beta_j)_{j=1,\dots,p}$, the regularization term of Problem (5.5)-(5.8) can be rewritten as follows:

$$\left\| (\boldsymbol{a}_{j\cdot\cdot}, \tilde{\boldsymbol{a}}_{j\cdot\cdot}) \right\|_\infty = \max\Big(\{|a_{jth}|\}_{\substack{t\in\tau_B \\ h=1,\dots,H}} \cup \{|\tilde{a}_{jth}|\}_{\substack{t\in\tau_L \\ h=1,\dots,H}} \Big) = \beta_j,\ j = 1,\dots,p,$$

where $\beta_j \geq 0$. We also need to impose $\beta_j \geq \pm\, a_{jt},\ j = 1,\dots,p,\ t \in \tau_B$, and $\beta_j \geq \pm\, \tilde{a}_{jt},\ j = 1,\dots,p,\ t \in \tau_L$. Hence, we have that Problem (5.5)-(5.8) is equivalent to the following smooth reformulation:

$$\min_{\substack{(\boldsymbol{a},\boldsymbol{\mu})\in\mathbb{R}^{(pH+1)|\tau_B|} \\ (\tilde{\boldsymbol{a}},\tilde{\boldsymbol{\mu}})\in\mathbb{R}^{(pH+1)|\tau_L|} \\ \boldsymbol{s}\in[\underline{s},\overline{s}]^{2H},\ \boldsymbol{\beta}\in\mathbb{R}^p}} \frac{1}{N}\sum_{i=1}^{N}\Big(\sum_{t\in\tau_L} P_{it}\left(\boldsymbol{a},\boldsymbol{\mu},\boldsymbol{s}\right)\varphi_{it}\left(\tilde{\boldsymbol{a}}_t,\tilde{\mu}_t,\boldsymbol{s}\right) - y_i\Big)^2 \tag{5.9}$$

$$+ \ \lambda^G \sum_{j=1}^{p} \beta_j \tag{5.10}$$

$$+ \ \lambda^{\text{length}} \sum_{h=1}^{H} (s_{2h} - s_{2h-1}) \tag{5.11}$$

$$\text{s.t.} \quad s_h \leq s_{h+1}, \ h = 1, \ldots, 2H-1, \tag{5.12}$$

$$\beta_j \geq \pm a_{jth}, j = 1, \ldots, p, t \in \tau_B, h = 1, \ldots, H, \tag{5.13}$$

$$\beta_j \geq \pm \tilde{a}_{jth}, j = 1, \ldots, p, t \in \tau_L, h = 1, \ldots, H. \tag{5.14}$$

5.2 Computational experiments

The aim of this section is to illustrate the performance of our sparse optimal randomized regression trees for functional data. Section 5.2.1 gives details on the procedure followed to test our approach. In Section 5.2.2, we discuss the prediction accuracy of our approach, against several benchmark regression methods. Finally, in Section 5.2.3 we illustrate our ability to trade in some of our prediction accuracy for a gain in sparsity.

5.2.1 Setup

Well-known publicly available functional data sets have been chosen for the computational experiments. Table 5.1 lists their names together with their number of observations, the number of points where the evaluations of the original functions are known and the domain, as well as the source where they can be downloaded. All the data sets are univariate, that is, $p = 1$, and coming from real-world applications, except for FHV which is simulated. See Figure 5.3 for a graphical representation of them.

Table 5.1: Information about the functional data sets considered.

Data set	N	$\#points$	$[\underline{s}, \overline{s}]$	Source
Tecator	215	100	$[850, 1050]$	Febrero-Bande and de la Fuente [2012]
Sunflower	111	309	$[0, 1]$	Picheny et al. [2019]
Sugar	268	571	$[275, 560]$	Aneiros and Vieu [2014]
FHV	1500	100	$[0, 2\pi]$	Ferraty et al. [2010]

Two performance criteria, namely prediction accuracy and sparsity, are assesed. The prediction accuracy is evaluated either by the mean squared error (MSE), or by the sum of the squared errors (SSE) for the sake of comparison with benchmarks. The lower the MSE or the SSE, the better the model in terms of prediction accuracy. The sparsity is evaluated by δ^{length}, as follows:

$$\delta^{\text{length}} = \left(1 - \frac{1}{\overline{s} - \underline{s}} \sum_{h=1}^{H} (s_{2h} - s_{2h-1})\right) \times 100.$$

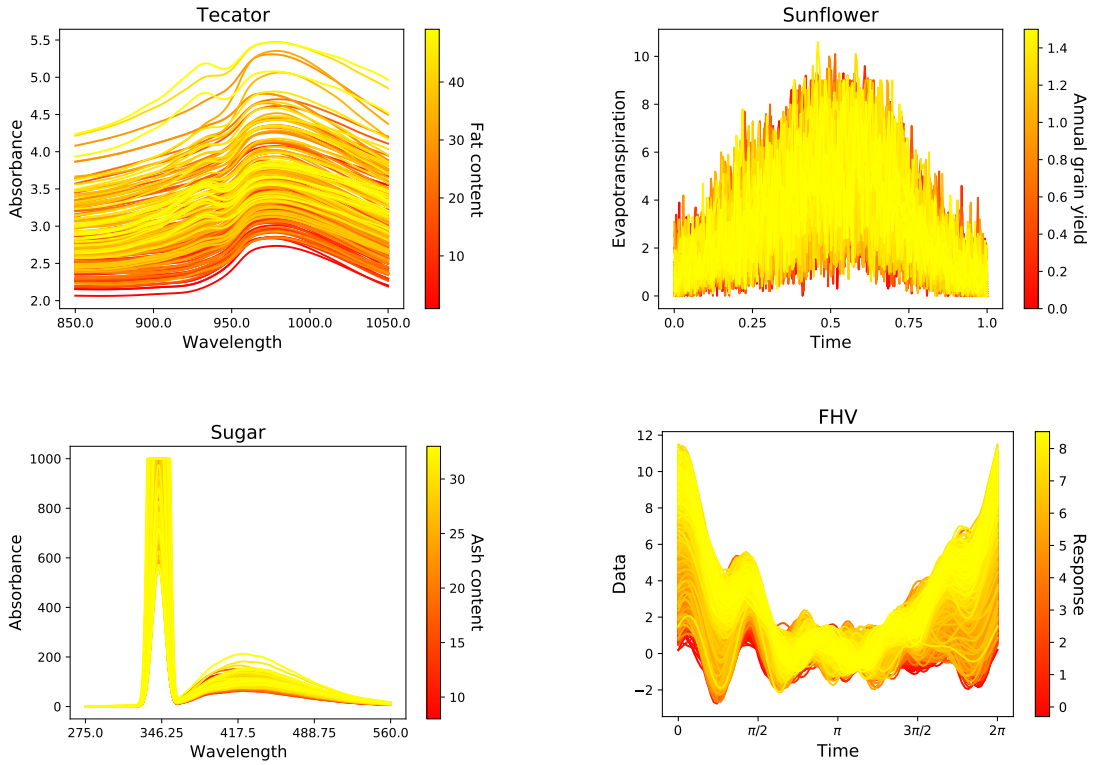


Figure 5.3: Graphical representation, for each data set, of the evaluations of the original predictor variable function (Y-axis) known for a set of points in their domain (X-axis). Each observation in the data set is colored according to the response variable. The higher the value of the response variable, the colder the color.

The higher the δ^{length}, the better the model in terms of sparsity.

A ten-fold cross-validation procedure has been applied in Section 5.2.2, and Table 5.2 represents the average results of such ten test subsets. In Section 5.2.3, the performance criteria illustrated in Figure 5.4 are evaluated over the whole data set in the training phase.

The logistic CDF has been chosen for our experiments:

$$F(\cdot) = \frac{1}{1 + \exp(-(\cdot)\gamma)},$$

with a large value of γ, namely, $\gamma = 512$. We will illustrate that this small level of randomization is enough for obtaining good results.

The formulation has been implemented using the `scipy.optimize` package [Virtanen et al., 2020] in Python 3.7 [Python Core Team, 2015]. As solver, we have used the SLSQP method [Kraft, 1988] that allows one to use gradient information. The response variable has been normalized to the $[-1, 1]$ interval. As seen in Table 5.3, in real-world applications, the functional predictor variables are known for particular points of their domains. Hence, smoothing techniques were applied as a preprocessing step so that an approximation to the original

function can be obtained. In our study, each individual has been previously turned into a smooth cubic spline approximation using the `scipy.interpolate` package, where the domain $[\underline{s}, \overline{s}]$ has been shifted to $[0, 1]$ w.l.g. For this reason, the decision variables $\boldsymbol{s}$ have been restricted to the $[0, 1]$ interval. The decision variables $\boldsymbol{a}$ and $\boldsymbol{\mu}$ have been restricted to the $[-1, 1]$ interval for the sake of numerical stability with exponentials. Our experiments have been conducted on a PC, with an Intel® Core™ i7-7700 CPU 3.60GHz processor and 32 GB RAM. The operating system is 64 bits.

5.2.2 Results for S-ORRT-FD

In this section, we focus on testing the prediction accuracy of our approach. S-ORRT-FD with depths $D = 1, 2$ and H in the grid $\{1, \ldots, 10\}$ is compared against two benchmark regression methods. The former is SVR-FD [Blanquero et al., 2020a], the state-of-the-art Machine Learning model for functional data based on Support Vector Regression. The same grid for H is used in SVR-FD, as well as three different variants $d = 0, 1, 2$, which include, respectively, the situations where just the information of the raw functional data, or their monotonicity, or both their monotonicity and convexity are considered. The latter is Random Forest (RF) [Breiman, 2001], a sophisticated tree-based regression method competitive in terms of prediction accuracy. In contrast to S-ORRT-FD and SVR-FD, RF has no direct control on the domain of the functional predictor variables being used. For comparison purposes with SVR-FD, Table 5.2 presents the average out-of-sample prediction accuracy SSE of all the methods.

For S-ORRT-FD, we have followed a multistart approach, where the process is repeated 500 times starting from different initial solutions. The solutions found for H are given as initial solutions to the optimization problem for $H + 1$, starting with random initial solutions for $H = 1$. The computing time taken by the S-ORRT-FD for a batch of ten values of H and 500 initial solutions typically ranges from 40 seconds (in Sunflower for $D = 1$) to 500 seconds (in FHV for $D = 2$). For SVR-FD, the results are taken from Blanquero et al. [2020a]. They are comparable since ten-fold cross-validation was also performed, and the response variable was normalized to the $[-1, 1]$ interval. The default parameter setting in `randomForest` [Liaw and Wiener, 2002] R package has been used for running RF.

With respect to SVR-FD, S-ORRT-FD is competitive according to the results obtained in Table 5.2 for the data sets considered. The best values of SSE for each value of H between SVR-FD and S-ORRT-FD have been highlighted in bold. For Tecator, S-ORRT-FD outperforms SVR-FD for values of $H \geq 3$. For Sunflower, S-ORRT-FD beats SVR-FD for values of $H \leq 5$. For Sugar, S-ORRT-FD tends to dominate in terms of prediction accuracy for each value of H, while for FHV it is the other way around.

With respect to RF, S-ORRT-FD always manages to find a small number of critical intervals H for which a better prediction accuracy is achieved for the data sets considered. This is the case for Tecator with $H = 1$ and $D = 1$, Sunflower with $H = 1$ and $D = 1$, Sugar with $H = 2$ and $D = 1$, and FHV with $H = 5$ and $D = 2$.

Table 5.2: Comparison between S-ORRT-FD, SVR-FD and RF in terms of out-of-sample average SSE.

Data set	Method	Out-of-sample average SSE									
		H									
		1	2	3	4	5	6	7	8	9	10
Tecator	S-ORRT-FD $D=1$	4.74	0.52	**0.21**	0.22	0.20	0.19	0.18	0.17	0.16	0.16
	S-ORRT-FD $D=2$	4.89	0.55	**0.21**	**0.14**	**0.17**	**0.13**	**0.13**	**0.11**	**0.10**	**0.11**
	SVR-FD $d=0$	0.25	**0.23**	0.24	0.29	0.41	0.40	0.45	0.45	0.49	0.50
	SVR-FD $d=1$	0.47	0.24	0.37	0.38	0.42	0.47	0.49	0.50	0.52	0.53
	SVR-FD $d=2$	**0.23**	0.36	0.39	0.43	0.47	0.55	0.58	0.60	0.61	0.61
	RF	2.13									
Sunflower	S-ORRT-FD $D=1$	**2.36**	2.69	**3.19**	**2.83**	**3.41**	6.00	6.50	4.62	4.34	4.65
	S-ORRT-FD $D=2$	2.57	**2.67**	3.68	4.24	4.21	4.73	6.40	5.91	5.85	5.40
	SVR-FD $d=0$	4.69	3.90	4.12	4.19	4.09	4.01	3.99	3.99	**3.92**	3.98
	SVR-FD $d=1$	4.29	3.86	4.11	4.32	4.24	4.08	4.07	4.01	9.95	3.98
	SVR-FD $d=2$	3.83	3.86	3.80	3.92	4.03	**3.91**	**3.93**	**3.96**	4.11	**3.94**
	RF	2.43									
Sugar	S-ORRT-FD $D=1$	**1.66**	**0.87**	2.12	2.06	2.44	**1.32**	**1.34**	**1.30**	**1.09**	**1.12**
	S-ORRT-FD $D=2$	1.76	1.26	**1.36**	2.28	**1.50**	1.58	1.85	1.44	3.74	2.75
	SVR-FD $d=0$	1.80	1.83	1.84	**1.84**	1.82	1.83	1.86	1.84	1.85	1.87
	SVR-FD $d=1$	2.08	1.81	1.82	1.88	1.84	1.88	1.85	1.83	1.82	1.84
	SVR-FD $d=2$	1.76	1.88	1.92	1.90	1.90	1.90	1.88	1.86	1.86	1.88
	RF	1.21									
FHV	S-ORRT-FD $D=1$	3.95	2.59	1.12	0.86	0.71	0.61	0.47	0.35	0.37	0.34
	S-ORRT-FD $D=2$	3.62	2.50	0.89	0.60	0.46	0.44	0.44	0.42	0.41	0.38
	SVR-FD $d=0$	**0.08**	**0.08**	**0.08**	**0.10**	**0.14**	**0.15**	**0.16**	**0.18**	0.19	0.21
	SVR-FD $d=1$	0.23	0.12	0.11	0.13	0.15	0.17	0.17	**0.18**	**0.18**	**0.18**
	SVR-FD $d=2$	0.12	0.13	0.12	0.14	0.15	0.16	0.17	0.19	0.21	0.25
	RF	0.51									

In summary, these numerical results illustrate that S-ORRT-FD is competitive with the state-of-the-art Machine Learning model tailored to functional data, SVR-FD, and outperforms the standard benchmark regression method RF. Furthermore, our approach can easily control global desirable properties such as sparsity, as seen in the next section.

5.2.3 Prediction accuracy and sparsity tradeoff

In the previous section, we have shown that considering a small number of critical intervals H, good results for S-ORRT-FD are achieved in terms of prediction accuracy. Nevertheless, the solution obtained could be not sparse, in the sense that the whole domain of the functional predictor variables could have been used for prediction. In this section, we illustrate that S-ORRT-FD is able to trade in some of its prediction accuracy for a gain in sparsity, measured as the proportion of the curves not being used. For the sake of conciseness, we illustrate this in the Tecator data set. We have solved Problem (5.9)-(5.14) with depth $D = 1$ and $H = 10$

for the sparsity parameters $\lambda^G = 0$ and λ^{length} in the grid $\{0\} \cup \{2^r,\ -10 \leq r \leq 10,\ r \in \mathbb{Z}\}$. We start solving the optimization problem with $\lambda^{\text{length}} = 0$ for 500 random initial solutions, and we continue for larger values of λ^{length}. The solutions found for fixed λ^{length}, are given as initial solutions to the next problem to be solved in the grid.

Figure 5.4 illustrates the different solutions obtained for MSE and δ^{length} in the grid of λ^{length} considered. Similarly to Figure 5.3, the functional predictor variable for Tecator data set is represented, and each individual is colored depending on their value of the response variable. In grey, critical intervals detected are represented. As expected, the larger the value of λ^{length}, the smaller the proportion of the curve to be used. Indeed, we can dramatically enhance sparsity δ^{length} from 17.9% to 95.8%, at the cost of slightly damaging the value of the MSE from 0.006 to 0.016. Notice that critical instants are being detected for large values of λ^{length}, according to Remark 5.1.

5.3 Conclusions

Many approaches on building optimal classification and regression trees for vectorial data have been recently proposed in the literature. In this chapter, we tailor the continuous optimization approach, proposed in previous chapters to construct a regression tree, in order to consider functional data in addition. Critical intervals for prediction are detected simultaneously. While being competitive in terms of prediction accuracy to benchmark methods, including RF, our approach can directly control the desired number of critical intervals, as well as the number of functional predictor variables and the proportion of the curves to be used along the tree.

Several extensions to our approach are of interest. First, the coefficient functions $a_{jt}(\cdot)$ and $\tilde{a}_{jt}(\cdot)$ defined at branch and leaf nodes, respectively, could be extended to more sophisticated funcions rather than piecewise constant ones. Second, this methodology can be extended straightaway to a classification problem with functional predictor variables using Chapters 2 and 3. Third, tailoring desirable properties introduced in Chapters 2 and 4, such as cost-sensitivity, fairness or local explainability, to the context of FDA deserves further investigation.

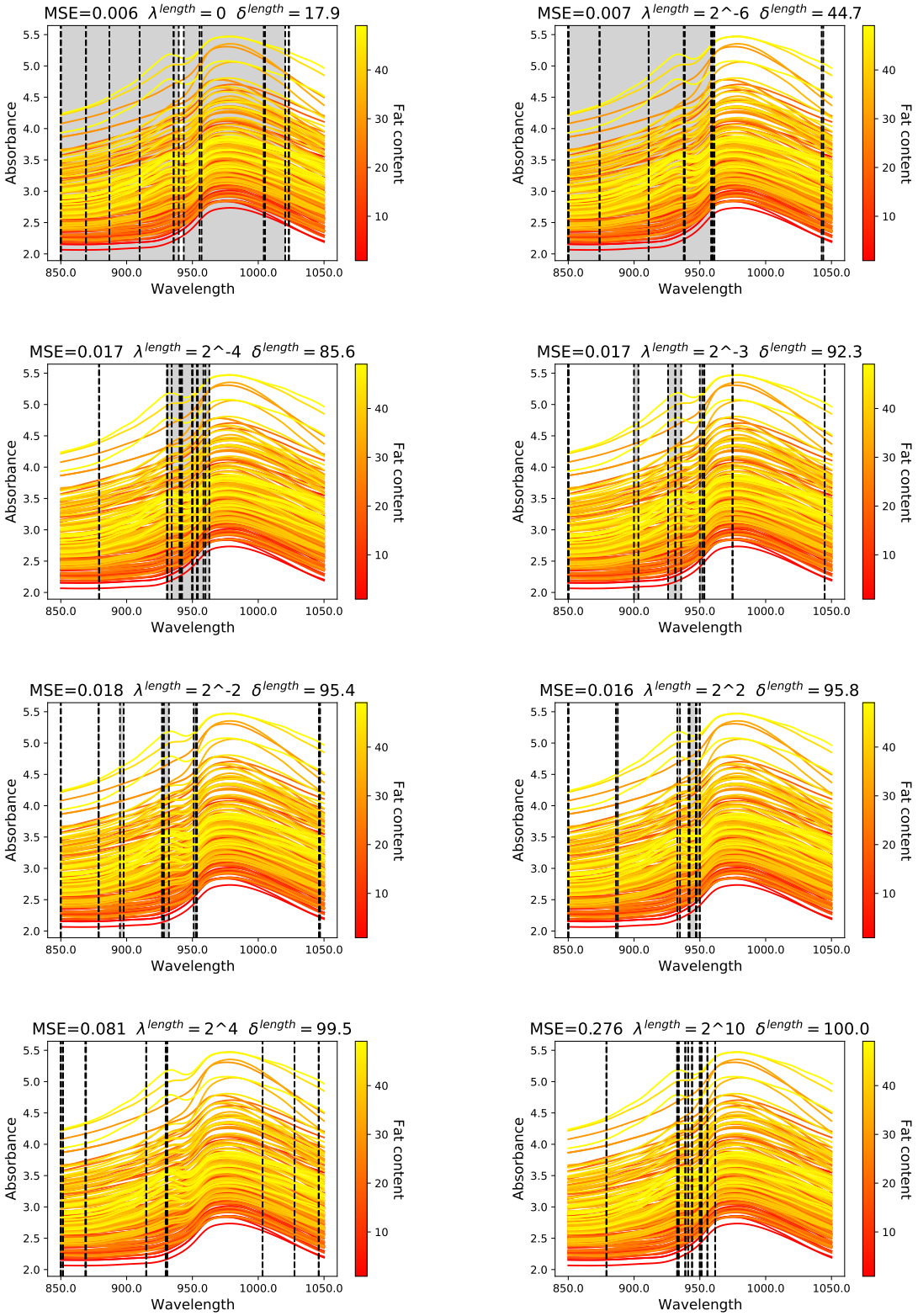


Figure 5.4: Critical intervals detection for the Tecator data set as a function of λ^{length}. Two performance criteria, prediction accuracy and sparsity, are evaluated by the MSE and δ^{length}, respectively, where δ^{length} represents the proportion of the curve not being used.

Chapter 6

General conclusions and future work

The impressive advances in hardware and software in the last decades have allowed the development of more powerful versions of classification and regression trees. In this PhD dissertation, we propose Continuous Optimization formulations and numerical solutions approaches to build optimal classification and regression trees that scale up well with the size of the training sample and are competitive in terms of prediction accuracy against benchmarks. We illustrate how these powerful formulations enhance the flexibility of tree models, being better suited to incorporate desirable properties such as sparsity, cost-sensitivity, explainability, and fairness, and to deal with complex data, such as functional data.

The research addressed in this PhD dissertation poses new challenges to be considered as future work, as detailed below.

First, tailoring optimal trees to other kinds of complex data that are not captured appropriately by standard implementations of these models such as time series data [Barrow and Crone, 2016; Carrizosa et al., 2013; Saha et al., 2021], spatial data [Georganos et al., 2021], text data [Carrizosa et al., 2018a; Martens and Provost, 2014; Ramon et al., 2020], image data [Affonso et al., 2017; Liu et al., 2018; Qing et al., 2020; Wang et al., 2021] or network data [Óskarsdóttir et al., 2022] is an interesting research avenue.

Second, while a (weighted) sum of squared residuals has been used as loss function in our regression models, other losses, such as the mean absolute error [Giloni et al., 2006] or quantile regression [Koenker and Hallock, 2001; Kriegler and Berk, 2010] can be considered, yielding optimization problems which deserve further analysis. Indeed, these losses need to be rewritten, in a similar fashion as for regularization terms, to ensure the smoothness of the objective function. For these losses, none of the decision variables is directly associated with the individuals, and therefore the dimension of the optimization problem behind regression still does not depend on the size of the training sample.

Third, one can address the problem of building, for any individual, a counterfactual explanation [Fernández et al., 2020; Lucic et al., 2022; Mothilal et al., 2020]. See Karimi et al. [2020]; Sokol and Flach [2019]; Verma et al. [2020] for recent surveys on counterfactual explanations. This is an explanation about how predictor variables need to change in order to obtain a different prediction. Finding counterfactuals amounts to solving (mixed-integer) global optimization problems, whose structure needs to be exploited to obtain fast algorithms, a must if counterfactuals are to be obtained for large data sets.

Fourth, it is known that bagging trees tends to enhance accuracy. An appropiate bagging scheme of our approach, where a collection of trees is built in order to have a global control on certain desirable properties, is also an interesting open question. A parallelization framework would be suitable to make the training of the collection of trees tractable.

Fifth, theoretical properties on the asymptotic performance of tree models need to be studied. Assuming data to be a random sample from a given distribution, an important question is to identify the statistical convergence of the random sequence of optimal trees and optimal values (e.g. optimal expected squared error in a regression tree) when the size of the training set

goes to infinity. Very limited results are available in the literature, making strong assumptions on the structure of the tree models. The reader is referred to Biau et al. [2008]; Denil et al. [2013]; Scornet [2016]; Scornet et al. [2015] for some results in this line.

Finally, the role of Mathematical Optimization in Machine Learning [Gambella et al., 2021] has been evinced through this dissertation. Nevertheless, the inverse problem has gained interest lately [Bengio et al., 2021; Lodi and Zarpellon, 2017; Václavík et al., 2018]. In this sense, one could use the optimal trees developed in this thesis to aid the resolution of Mathematical Optimization problems, as done in Bertsimas and Oztürk [2022].

List of Figures

List of Tables

References

Affonso, C., Rossi, A. L. D., Vieira, F. H. A., and Ponce de Leon Ferreira de Carvalho, A. C. (2017). Deep learning for biological image classification. *Expert Systems with Applications*, 85:114–122.

Aghaei, S., Azizi, M., and Vayanos, P. (2019). Learning optimal and fair decision trees for non-discriminative decision-making. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 1418–1426.

Aghaei, S., Gomez, A., and Vayanos, P. (2020). Learning optimal classification trees: Strong max-flow formulations. *arXiv preprint arXiv:2002.09142*.

Aglin, G., Nijssen, S., and Schaus, P. (2020). Learning optimal decision trees using caching branch-and-bound search. In *Thirty-Fourth AAAI Conference on Artificial Intelligence*.

Ahuja, R., Magnanti, T., and Orlin, J. (1993). *Network Flows: Theory, Algorithms, and Applications*. Prentice Hall, New Jersey.

Akyüz, M. H. and Birbil, Ş. İ. (2021). Discovering classification rules for interpretable learning with linear programming. *arXiv preprint arXiv:2104.10751*.

Altmann, A., Toloşi, L., Sander, O., and Lengauer, T. (2010). Permutation importance: a corrected feature importance measure. *Bioinformatics*, 26(10):1340–1347.

Alvarez, A., Louveaux, Q., and Wehenkel, L. (2017). A machine learning-based approximation of strong branching. *INFORMS Journal on Computing*, 29(1):185–195.

Aneiros, G. and Vieu, P. (2014). Variable selection in infinite-dimensional problems. *Statistics & Probability Letters*, 94:12–20.

Aneiros, G. and Vieu, P. (2016). Sparse nonparametric model for regression with functional covariate. *Journal of Nonparametric Statistics*, 28(4):839–859.

Aouad, A., Elmachtoub, A., Ferreira, K., and McNellis, R. (2019). Market segmentation trees. *arXiv preprint arXiv:1906.01174*.

Athey, S. (2018). The impact of machine learning on economics. In *The Economics of Artificial Intelligence: An Agenda*. University of Chicago Press.

Athey, S. and Imbens, G. (2015). Machine learning for estimating heterogeneous causal effects. Technical report, Research Papers 3350, Stanford University, Graduate School of Business.

Athey, S., Tibshirani, J., and Wager, S. (2019). Generalized random forests. *The Annals of Statistics*, 47(2):1148–1178.

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

Balakrishnan, S. and Madigan, D. (2006). Decision trees for functional variables. In *Sixth International Conference on Data Mining (ICDM'06)*, pages 798–802.

Barros, R., Basgalupp, M., De Carvalho, A., and Freitas, A. (2011). A survey of evolutionary algorithms for decision-tree induction. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 42(3):291–312.

Barrow, D. and Crone, S. (2016). A comparison of adaboost algorithms for time series forecast combination. *International Journal of Forecasting*, 32(4):1103–1119.

Belli, E. and Vantini, S. (2021). Measure inducing classification and regression trees for functional data. *Statistical Analysis and Data Mining*, 15(5):553–569.

Bénard, C., Biau, G., Da Veiga, S., and Scornet, E. (2019). SIRUS: making random forests interpretable. *arXiv preprint arXiv:1908.06852*.

Bénard, C., Biau, G., Da Veiga, S., and Scornet, E. (2021). Interpretable random forests via rule extraction. In *International Conference on Artificial Intelligence and Statistics*, pages 937–945.

Bengio, Y., Lodi, A., and Prouvost, A. (2021). Machine learning for combinatorial optimization: a methodological tour d'horizon. *European Journal of Operational Research*, 290(2):405–421.

Benítez-Peña, S., Bogetoft, P., and Morales, D. R. (2020). Feature selection in data envelopment analysis: A mathematical optimization approach. *Omega*, 96:102068.

Benítez-Peña, S., Carrizosa, E., Guerrero, V., Jiménez-Gamero, M. D., Martín-Barragán, B., Molero-Río, C., Ramírez-Cobo, P., Morales, D. R., and Sillero-Denamiel, M. R. (2021). On sparse ensemble methods: An application to short-term predictions of the evolution of covid-19. *European Journal of Operational Research*, 295(2):648–663.

Bennett, K. (1992). *Decision tree construction via linear programming*. Center for Parallel Optimization, Computer Sciences Department, University of Wisconsin.

Bennett, K. and Blue, J. (1996). Optimal decision trees. *Rensselaer Polytechnic Institute Math Report*, 214.

Bennett, K. and Mangasarian, O. (1992). Robust linear programming discrimination of two linearly inseparable sets. *Optimization Methods and Software*, 1:23–24.

Berrendero, J. R., Bueno-Larraz, B., and Cuevas, A. (2019). An rkhs model for variable selection in functional linear regression. *Journal of Multivariate Analysis*, 170:25–45.

Bertsimas, D. and Digalakis, V. (2022). The backbone method for ultra-high dimensional sparse machine learning. *Machine Learning*, 111:2161–2212.

Bertsimas, D. and Dunn, J. (2017). Optimal classification trees. *Machine Learning*, 106(7):1039–1082.

Bertsimas, D., Dunn, J., and Mundru, N. (2019). Optimal prescriptive trees. *INFORMS Journal on Optimization*, 1(2):164–183.

Bertsimas, D., O'Hair, A., Relyea, S., and Silberholz, J. (2016). An analytics approach to designing combination chemotherapy regimens for cancer. *Management Science*, 62(5):1511–1531.

Bertsimas, D. and Oztürk, B. (2022). Global optimization via optimal decision trees. Technical report, Massachusetts Institute of Technology.

Bertsimas, D. and Shioda, R. (2007). Classification and regression via integer optimization. *Operations Research*, 55(2):252–271.

Biau, G., Devroye, L., and Lugosi, G. (2008). Consistency of random forests and other averaging classifiers. *Journal of Machine Learning Research*, 9:2015–2033.

Biau, G. and Scornet, E. (2016). A random forest guided tour. *TEST*, 25(2):197–227.

Birbil, S., Edali, M., and Yüceoğlu, B. (2020). Rule covering for interpretation and boosting. *arXiv preprint arXiv:2007.06379*.

Blake, C. and Merz, C. (1998). UCI Repository of Machine Learning Databases. http://www.ics.uci.edu/∼mlearn/MLRepository.html. University of California, Irvine, Department of Information and Computer Sciences.

Blanquero, R., Carrizosa, E., Jiménez-Cordero, A., and Martín-Barragán, B. (2020a). Selection of time instants and intervals with support vector regression for multivariate functional data. *Computers & Operations Research*, 123:105050.

Blanquero, R., Carrizosa, E., Molero-Río, C., and Romero Morales, D. (2020b). Sparsity in optimal randomized classification trees. *European Journal of Operational Research*, 284(1):255 – 272.

Blanquero, R., Carrizosa, E., Molero-Río, C., and Romero Morales, D. (2022a). Sparse optimal randomized regression trees in functional data analysis. Technical report, Institute of Mathematics of the University of Seville.

Blanquero, R., Carrizosa, E., Molero-Río, C., and Romero Morales, D. (2021a). Optimal randomized classification trees. *Computers & Operations Research*, 132:105281.

Blanquero, R., Carrizosa, E., Molero-Río, C., and Romero Morales, D. (2022b). On sparse optimal regression trees. *European Journal of Operational Research*, 299(3):1045–1054.

Blanquero, R., Carrizosa, E., Ramírez-Cobo, P., and Sillero-Denamiel, M. R. (2021b). A cost-sensitive constrained lasso. *Advances in Data Analysis and Classification*, 15:121–158.

Bottou, L., Curtis, F., and Nocedal, J. (2018). Optimization methods for large-scale machine learning. *SIAM Review*, 60(2):223–311.

Breiman, L. (2001). Random forests. *Machine Learning*, 45(1):5–32.

Breiman, L., Friedman, J., Stone, C. J., and Olshen, R. A. (1984). *Classification and regression trees*. CRC Press.

Brodley, C. and Utgoff, P. (1995). Multivariate decision trees. *Machine Learning*, 19(1):45–77.

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

Carrizosa, E., Guerrero, V., Hardt, D., and Romero Morales, D. (2018a). On building online visualization maps for news data streams by means of mathematical optimization. *Big Data*, 6(2):139–158.

Carrizosa, E., Guerrero, V., and Romero Morales, D. (2018b). Visualizing data as objects by DC (difference of convex) optimization. *Mathematical Programming, Series B*, 169:119–140.

Carrizosa, E., Guerrero, V., Romero Morales, D., and Satorra, A. (2020). Enhancing interpretability in factor analysis by means of mathematical optimization. *Multivariate Behavioral Research*, 55(5):748–762.

Carrizosa, E., Kurishchenko, K., Marín, A., and Romero Morales, D. (2022a). Interpreting clusters via prototype optimization. *Omega*, 107:102543.

Carrizosa, E., Molero-Río, C., and Romero Morales, D. (2021a). Mathematical optimization in classification and regression trees. *TOP*, 29(1):5–33.

Carrizosa, E., Mortensen, L., Romero Morales, D., and Sillero-Denamiel, M. (2022b). The tree based linear regression model for hierarchical categorical variables. *Expert Systems with Applications*, 203:117423.

Carrizosa, E., Nogales-Gómez, A., and Romero Morales, D. (2017). Clustering categories in support vector machines. *Omega*, 66:28–37.

Carrizosa, E., Olivares-Nadal, A., and Ramírez-Cobo, P. (2013). Time series interpolation via global optimization of moments fitting. *European Journal of Operational Research*, 230(1):97–112.

Carrizosa, E., Restrepo, M. G., and Romero Morales, D. (2021b). On clustering categories of categorical predictors in generalized linear models. *Expert Systems with Applications*, 182:115245.

Carrizosa, E. and Romero Morales, D. (2013). Supervised classification and mathematical optimization. *Computers & Operations Research*, 40(1):150–165.

Casalicchio, G., Molnar, C., and Bischl, B. (2019). Visualizing the feature importance for black box models. In Berlingerio, M., Bonchi, F., Gärtner, T., Hurley, N., and Ifrim, G., eds., *Machine Learning and Knowledge Discovery in Databases*, pages 655–670, Cham. Springer International Publishing.

Chaovalitwongse, W. A., Fan, Y.-J., and Sachdeo, R. C. (2008). Novel optimization models for abnormal brain activity classification. *Operations Research*, 56(6):1450–1460.

Cohen, S., Dror, G., and Ruppin, E. (2007). Feature selection via coalitional game theory. *Neural Computation*, 19(7):1939–1961.

Cousins, C. and Riondato, M. (2019). CaDET: interpretable parametric conditional density estimation with decision trees and forests. *Machine Learning*, 108(8-9):1613–1634.

Csárdi, G. and Truong, A. (2012). oblique.tree. `https://github.com/cran/oblique.tree`.

Cuevas, A. (2014). A partial overview of the theory of statistics with functional data. *Journal of Statistical Planning and Inference*, 147:1–23.

Cui, Z., Chen, W., He, Y., and Chen, Y. (2015). Optimal action extraction for random forests and boosted trees. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 179–188.

Dash, S., Günlük, O., and Wei, D. (2018). Boolean decision rules via column generation. In *Advances in Neural Information Processing Systems*, pages 4655–4665.

Demiriz, A., Bennett, K., and Shawe-Taylor, J. (2002). Linear programming boosting via column generation. *Machine Learning*, 46:225–254.

Demirović, E. and Stuckey, P. (2021). Optimal decision trees for nonlinear metrics. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 3733–3741.

Demirović, E., Lukina, A., Hebrard, E., Chan, J., Bailey, J., Leckie, C., Ramamohanarao, K., and Stuckey, P. J. (2022). MurTree: Optimal Classification Trees via Dynamic Programming and Search. *Journal of Machine Learning Research*, 23(26):1–47.

Deng, H. and Runger, G. (2012). Feature selection via regularized trees. In *The 2012 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE.

Deng, H. and Runger, G. (2013). Gene selection with guided regularized random forest. *Pattern Recognition*, 46(12):3483–3489.

Denil, M., Matheson, D., and Freitas, N. (2013). Consistency of online random forests. In *International Conference on Machine Learning*, pages 1256–1264.

Dougherty, J., Kohavi, R., and Sahami, M. (1995). Supervised and unsupervised discretization of continuous features. In *Machine Learning Proceedings 1995*, pages 194–202.

Duarte Silva, A. (2017). Optimization approaches to supervised classification. *European Journal of Operational Research*, 261(2):772–788.

Dunn, J. (2018). *Optimal trees for prediction and prescription*. Unpublished doctoral dissertation, Massachusetts Institute of Technology.

Esteve, M., Aparicio, J., Rabasa, A., and Rodriguez-Sala, J. (2020). Efficiency analysis trees: a new methodology for estimating production frontiers through decision trees. *Expert Systems with Applications*, 162:113783.

European Commission (2020). *White Paper on Artificial Intelligence : a European approach to excellence and trust*. `https://ec.europa.eu/info/sites/info/files/commission-white-paper-artificial-intelligence-feb2020$_$en.pdf`.

Fang, X., Liu Sheng, O., and Goes, P. (2013). When is the right time to refresh knowledge discovered from data? *Operations Research*, 61(1):32–44.

Fawagreh, K., Gaber, M. M., and Elyan, E. (2014). Random forests: from early developments to recent advancements. *Systems Science & Control Engineering*, 2(1):602–609.

Fayyad, U. and Irani, K. (1992). The attribute selection problem in decision tree generation. In *AAAI*, pages 104–110.

Febrero-Bande, M. and de la Fuente, M. O. (2012). Statistical Computing in Functional Data Analysis: The R Package fda.usc. *Journal of Statistical Software*, 51(4):1–28.

Febrero-Bande, M., González-Manteiga, W., and Oviedo De La Fuente, M. (2019). Variable selection in functional additive regression models. *Computational Statistics*, 34(2):469–487.

Fernández-Delgado, M., Cernadas, E., Barro, S., and Amorim, D. (2014). Do we need hundreds of classifiers to solve real world classification problems? *Journal of Machine Learning Research*, 15(1):3133–3181.

Fernández, R., Martín de Diego, I., Aceña, V., Fernández-Isabel, A., and Moguerza, J. (2020). Random forest explainability using counterfactual sets. *Information Fusion*, 63:196–207.

Ferraty, F., Hall, P., and Vieu, P. (2010). Most-predictive design points for functional data predictors. *Biometrika*, 97(4):807–824.

Firat, M., Crognier, G., Gabor, A., Hurkens, C., and Zhang, Y. (2019). Column generation based math-heuristic for classification trees. *Computers & Operations Research*, 116:104866.

Fountoulakis, K. and Gondzio, J. (2016). A second-order method for strongly convex ℓ_1-regularization problems. *Mathematical Programming*, 156(1):189–219.

Freitas, A. (2014). Comprehensible classification models: a position paper. *ACM SIGKDD Explorations Newsletter*, 15(1):1–10.

Friedman, J. (2001). Greedy function approximation: a gradient boosting machine. *Annals of Statistics*, pages 1189–1232.

Friedman, J. (2002). Stochastic gradient boosting. *Computational Statistics & Data Analysis*, 38(4):367–378.

Friedman, J., Hastie, T., and Tibshirani, R. (2010). Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software*, 33(1):1–22.

Fu, Z., Golden, B., Lele, S., Raghavan, S., and Wasil, E. (2003). A genetic algorithm-based approach for building accurate decision trees. *INFORMS Journal on Computing*, 15(1):3–22.

Gambella, C., Ghaddar, B., and Naoum-Sawaya, J. (2021). Optimization problems for machine learning: A survey. *European Journal of Operational Research*, 290(3):807–828.

Genuer, R., Poggi, J.-M., Tuleau-Malot, C., and Villa-Vialaneix, N. (2017). Random Forests for Big Data. *Big Data Research*, 9:28–46.

Georganos, S., Grippa, T., Gadiaga, A., Linard, C., Lennert, M., Vanhuysse, S., Mboga, N., Wolff, E., and Kalogirou, S. (2021). Geographical random forests: a spatial extension of the random forest algorithm to address spatial heterogeneity in remote sensing and population modelling. *Geocarto International*, 36(2):121–136.

Gevrey, M., Dimopoulos, I., and Lek, S. (2003). Review and comparison of methods to study the contribution of variables in artificial neural network models. *Ecological Modelling*, 160(3):249–264.

Ghorbani, A. and Zou, J. (2020). Neuron Shapley: Discovering the Responsible Neurons. *Advances in Neural Information Processing Systems*, 33:5922–5932.

Giloni, A., Simonoff, J. S., and Sengupta, B. (2006). Robust weighted lad regression. *Computational Statistics & Data Analysis*, 50(11):3124–3140.

Goia, A. and Vieu, P. (2016). An introduction to recent advances in high/infinite dimensional statistics. *Journal of Multivariate Analysis*, 146:1–6.

Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. MIT Press.

Goodman, B. and Flaxman, S. (2017). European Union regulations on algorithmic decision-making and a "right to explanation". *AI Magazine*, 38(3):50–57.

Grubinger, T., Zeileis, A., and Pfeiffer, K.-P. (2014). evtree: Evolutionary Learning of Globally Optimal Classification and Regression Trees in R. *Journal of Statistical Software*, 61(1):1–29.

Günlük, O., Kalagnanam, J., Li, M., Menickelly, M., and Scheinberg, K. (2021). Optimal Decision Trees for Categorical Data via Integer Programming. *Journal of Global Optimization*, 81:233–260.

Gunning, D. and Aha, D. (2019). DARPA's Explainable Artificial Intelligence Program. *AI Magazine*, 40(2):44–58.

Guyon, I. and Elisseeff, A. (2003). An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3:1157–1182.

Harrison Jr, D. and Rubinfeld, D. L. (1978). Hedonic housing prices and the demand for clean air. *Journal of Environmental Economics and Management*, 5(1):81–102.

Hart, W., Laird, C., Watson, J.-P., Woodruff, D., Hackebeil, G., Nicholson, B., and Siirola, J. (2017). *Pyomo–Optimization Modeling in Python*, volume 67. Springer Science & Business Media, second edition.

Hart, W., Watson, J.-P., and Woodruff, D. (2011). Pyomo: modeling and solving mathematical programs in Python. *Mathematical Programming Computation*, 3(3):219–260.

Hastie, T., Tibshirani, R., and Friedman, J. (2009). *The Elements of Statistical Learning*. Springer, New York, 2nd edition.

Hastie, T., Tibshirani, R., and Wainwright, M. (2015). *Statistical learning with sparsity: the lasso and generalizations*. CRC press.

Holter, S., Gomez, O., and Bertini, E. (2018). *FICO Explainable Machine Learning Challenge*. `https://community.fico.com/s/explainable-machine-learning-challenge`.

Höppner, S., Stripling, E., Baesens, B., vanden Broucke, S., and Verdonck, T. (2020). Profit driven decision trees for churn prediction. *European Journal of Operational Research*, 284(3):920–933.

Hu, X., Rudin, C., and Seltzer, M. (2019). Optimal sparse decision trees. In *Advances in Neural Information Processing Systems 32*, pages 7265–7273.

Hyafil, L. and Rivest, R. (1976). Constructing optimal binary decision trees is NP-complete. *Information Processing Letters*, 5(1):15–17.

Iosifidis, V. and Ntoutsi, E. (2019). Adafair: Cumulative fairness adaptive boosting. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, CIKM '19, page 781–790, New York, NY, USA. Association for Computing Machinery.

Irsoy, O., Yıldız, O., and Alpaydın, E. (2012). Soft decision trees. In *Proceedings of the 21st International Conference on Pattern Recognition (ICPR2012)*, pages 1819–1822.

Jakaitiene, A., Sangiovanni, M., Guarracino, M., and Pardalos, P. (2016). *Multidimensional Scaling for Genomic Data*, pages 129–139. Springer International Publishing, Cham.

Kao, H.-P. and Tang, K. (2014). Cost-sensitive decision tree induction with label-dependent late constraints. *INFORMS Journal on Computing*, 26(2):238–252.

Karimi, A.-H., Barthe, G., Schölkopf, B., and Valera, I. (2020). A survey of algorithmic recourse: definitions, formulations, solutions, and prospects. *arXiv preprint arXiv:2010.04050*.

Kass, G. (1980). An exploratory technique for investigating large quantities of categorical data. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 29(2):119–127.

Katuwal, R., Suganthan, P., and Zhang, L. (2020). Heterogeneous oblique random forest. *Pattern Recognition*, 99:107078.

Khalil, E. B., Bodic, P. L., Song, L., Nemhauser, G., and Dilkina, B. (2016). Learning to branch in mixed integer programming. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, pages 724–731.

Kim, H. and Loh, W.-Y. (2001). Classification trees with unbiased multiway splits. *Journal of the American Statistical Association*, 96(454):589–604.

Kleinberg, J., Lakkaraju, H., Leskovec, J., Ludwig, J., and Mullainathan, S. (2018). Human decisions and machine predictions. *The Quarterly Journal of Economics*, 133(1):237–293.

Koenker, R. and Hallock, K. (2001). Quantile regression. *Journal of Economic Perspectives*, 15(4):143–156.

Kong, D., Xue, K., Yao, F., and Zhang, H. H. (2016). Partially functional linear regression in high dimensions. *Biometrika*, 103(1):147–159.

Kraft, D. (1988). A software package for sequential quadratic programming. Technical report, DFVLR-FB 88-28, DLR German Aerospace Center — Institute for Flight Mechanics, Köln, Germany.

Kriegler, B. and Berk, R. (2010). Small area estimation of the homeless in Los Angeles: An application of cost-sensitive stochastic gradient boosting. *The Annals of Applied Statistics*, pages 1234–1255.

Kruber, M., Lübbecke, M. E., and Parmentier, A. (2017). Learning when to use a decomposition. In *International Conference on AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems*, pages 202–210.

Lee, C.-Y. and Cai, J.-Y. (2020). LASSO variable selection in data envelopment analysis with small datasets. *Omega*, 91:102019.

Leng, X. and Müller, H.-G. (2005). Classification using functional data analysis for temporal gene expression data. *Bioinformatics*, 22(1):68–76.

Li, X.-B., Sweigart, J., Teng, J., Donohue, J., Thombs, L., and Wang, S. (2003). Multivariate decision trees using linear discriminants and tabu search. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, 33(2):194–205.

Liaw, A. and Wiener, M. (2002). Classification and Regression by randomForest. *R News*, 2(3):18–22.

Liberti, L. (2020). Distance geometry and data science. *TOP*, 28:271–339.

Lichman, M. (2013). UCI Machine Learning Repository. http://archive.ics.uci.edu/ml. University of California, Irvine, School of Information and Computer Sciences.

Lin, J., Zhong, C., Hu, D., Rudin, C., and Seltzer, M. (2020). Generalized and scalable optimal sparse decision trees. In *International Conference on Machine Learning*, pages 6150–6160.

Liu, H., Hussain, F., Tan, C., and Dash, M. (2002). Discretization: An enabling technique. *Data Mining and Knowledge Discovery*, 6(4):393–423.

Liu, Y., Wen, K., Gao, Q., Gao, X., and Nie, F. (2018). SVM based multi-label learning with missing labels for image annotation. *Pattern Recognition*, 78:307–317.

Lodi, A. and Zarpellon, G. (2017). On learning and branching: a survey. *TOP*, 25(2):207–236.

Loh, W.-Y. (2014). Fifty years of classification and regression trees. *International Statistical Review*, 82(3):329–348.

Loh, W.-Y. and Shih, Y.-S. (1997). Split selection methods for classification trees. *Statistica Sinica*, 7(4):815–840.

Louppe, G., Wehenkel, L., Sutera, A., and Geurts, P. (2013). Understanding variable importances in forests of randomized trees. In *Advances in Neural Information Processing Systems*, pages 431–439.

Lucic, A., Oosterhuis, H., Haned, H., and de Rijke, M. (2022). Focus: Flexible optimizable counterfactual explanations for tree ensembles. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 5313–5322.

Lundberg, S., Erion, G., Chen, H., DeGrave, A., Prutkin, J., Nair, B., Katz, R., Himmelfarb, J., Bansal, N., and Lee, S.-I. (2020). From local explanations to global understanding with explainable AI for trees. *Nature Machine Intelligence*, 2(1):2522–5839.

Lundberg, S. and Lee, S.-I. (2017). A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems*, pages 4765–4774.

Martens, D., Baesens, B., Gestel, T., and Vanthienen, J. (2007). Comprehensible credit scoring models using rule extraction from support vector machines. *European Journal of Operational Research*, 183(3):1466–1476.

Martens, D. and Provost, F. (2014). Explaining data-driven document classifications. *MIS Quarterly*, 38(1):73–99.

Martínez Torres, J., Iglesias Comesaña, C., and García-Nieto, P. (2019). Machine learning techniques applied to cybersecurity. *International Journal of Machine Learning and Cybernetics*, 10(10):2823–2836.

Meinshausen, N. (2010). Node harvest. *The Annals of Applied Statistics*, 4(4):2049–2072.

Menze, B., Kelm, B., Splitthoff, D., Koethe, U., and Hamprecht, F. (2011). On oblique random forests. In Gunopulos, D., Hofmann, T., Malerba, D., and Vazirgiannis, M., eds., *Machine Learning and Knowledge Discovery in Databases*, pages 453–469.

Miller, T. (2019). Explanation in artificial intelligence: Insights from the social sciences. *Artificial Intelligence*, 267:1–38.

Miron, M., Tolan, S., Gómez, E., and Castillo, C. (2020). Addressing multiple metrics of group fairness in data-driven decision making. *arXiv preprint arXiv:2003.04794*.

Mišić, V. V. (2020). Optimization of tree ensembles. *Operations Research*, 68(5):1605–1624.

Möller, A., Tutz, G., and Gertheiss, J. (2016). Random forests for functional covariates. *Journal of Chemometrics*, 30(12):715–725.

Molnar, C., Casalicchio, G., and Bischl, B. (2018). iml: An R package for interpretable machine learning. *Journal of Open Source Software*, 3(26):786.

Molnar, C., Casalicchio, G., and Bischl, B. (2020). Interpretable machine learning – a brief history, state-of-the-art and challenges. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 417–431.

Mothilal, R., Sharma, A., and Tan, C. (2020). Explaining machine learning classifiers through diverse counterfactual explanations. In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency*, pages 607–617.

Murthy, S., Kasif, S., and Salzberg, S. (1994). A system for induction of oblique decision trees. *Journal of Artificial Intelligence Research*, 2:1–32.

Narodytska, N., Ignatiev, A., Pereira, F., and Marques-Silva, J. (2018). Learning Optimal Decision Trees with SAT. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence (IJCAI-18)*, pages 1362–1368.

Nijssen, S. and Fromont, E. (2010). Optimal constraint-based decision tree induction from itemset lattices. *Data Mining and Knowledge Discovery*, 21(1):9–51.

Norouzi, M., Collins, M., Johnson, M., Fleet, D., and Kohli, P. (2015). Efficient non-greedy optimization of decision trees. In *Advances in Neural Information Processing Systems*, pages 1729–1737.

Obermeyer, Z., Powers, B., Vogeli, C., and Mullainathan, S. (2019). Dissecting racial bias in an algorithm used to manage the health of populations. *Science*, 366(6464):447–453.

Olafsson, S., Li, X., and Wu, S. (2008). Operations research and data mining. *European Journal of Operational Research*, 187(3):1429–1448.

Orsenigo, C. and Vercellis, C. (2003). Multivariate classification trees based on minimum features discrete support vector machines. *IMA Journal of Management Mathematics*, 14(3):221–234.

Óskarsdóttir, M., Ahmed, W., Antonio, K., Baesens, B., Dendievel, R., Donas, T., and Reynkens, T. (2022). Social network analytics for supervised fraud detection in insurance. *Risk Analysis*, 42(8):1872–1890.

Palagi, L. (2019). Global optimization issues in deep network regression: an overview. *Journal of Global Optimization*, 73(2):239–277.

Pande, A., Li, L., Rajeswaran, J., Ehrlinger, J., Kogalur, U., Blackstone, E., and Ishwaran, H. (2017). Boosted multivariate trees for longitudinal data. *Machine Learning*, 106(2):277–305.

Pangilinan, J. and Janssens, G. (2011). Pareto-optimality of oblique decision trees from evolutionary algorithms. *Journal of Global Optimization*, 51(2):301–311.

Pardalos, P., Boginski, V., and Vazacopoulos, A., eds. (2007). *Data Mining in Biomedicine*. Springer Optimization and Its Applications. Springer.

Pfetsch, M. and Pokutta, S. (2020). Ipboost–non-convex boosting via integer programming. In *International Conference on Machine Learning*, pages 7663–7672.

Piccialli, V. and Sciandrone, M. (2018). Nonlinear optimization and support vector machines. *4OR*, 16(2):111–149.

Picheny, V., Servien, R., and Villa-Vialaneix, N. (2019). Interpretable sparse SIR for functional data. *Statistics and Computing*, 29(2):255–267.

Pospisil, T. and Lee, A. (2019). `(f)RFCDE`: Random forests for conditional density estimation and functional data. *arXiv preprint arXiv:1906.07177*.

Python Core Team (2015). Python: A dynamic, open source programming language. Python Software Foundation. `https://www.python.org`.

Qing, Y., Zeng, Y., Li, Y., and Huang, G.-B. (2020). Deep and wide feature based extreme learning machine for image classification. *Neurocomputing*, 412:426–436.

Quinlan, J. (1993). *C4.5: Programs for Machine Learning*. Morgan Kaufmann, San Mateo, CA.

Rahman, R., Dhruba, S., Ghosh, S., and Pal, R. (2019). Functional random forest with applications in dose-response predictions. *Scientific Reports*, 9(1):1–14.

Ramon, Y., Martens, D., Provost, F., and Evgeniou, T. (2020). A comparison of instance-level counterfactual explanation algorithms for behavioral and textual data: Sedc, lime-c and shap-c. *Advances in Data Analysis and Classification*, 14(4):801–819.

Ribeiro, M., Singh, S., and Guestrin, C. (2016). "Why Should I Trust You?": Explaining the Predictions of Any Classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1135–1144.

Ridgeway, G. (2013). The pitfalls of prediction. *National Institute of Justice Journal*, 271:34–40.

Romei, A. and Ruggieri, S. (2014). A multidisciplinary survey on discrimination analysis. *The Knowledge Engineering Review*, 29(5):582–638.

Rudin, C. (2019). Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, 1(5):206–215.

Rudin, C., Chen, C., Chen, Z., Huang, H., Semenova, L., and Zhong, C. (2022). Interpretable machine learning: Fundamental principles and 10 grand challenges. *Statistics Surveys*, 16:1–85.

Rudin, C. and Ertekin, Ş. (2018). Learning customized and optimized lists of rules with mathematical programming. *Mathematical Programming Computation*, 10(4):659–702.

Ruggieri, S. (2019). Complete search for feature selection in decision trees. *Journal of Machine Learning Research*, 20(104):1–34.

Saha, A., Basu, S., and Datta, A. (2021). Random forests for spatially dependent data. *Journal of the American Statistical Association*, pages 1–19.

Savickỳ, P., Klaschka, J., and Antoch, J. (2000). Optimal classification trees. In *COMPSTAT*, pages 427–432.

Scornet, E. (2016). On the asymptotics of random forests. *Journal of Multivariate Analysis*, 146:72–83.

Scornet, E., Biau, G., and Vert, J.-P. (2015). Consistency of random forests. *The Annals of Statistics*, 43(4):1716–1741.

Shapiro, A., Dentcheva, D., and Ruszczyński, A. (2009). *Lectures on stochastic programming: modeling and theory*. SIAM.

Sherali, H., Hobeika, A., and Jeenanunta, C. (2009). An optimal constrained pruning strategy for decision trees. *INFORMS Journal on Computing*, 21(1):49–61.

Sokol, K. and Flach, P. (2019). Counterfactual explanations of machine learning predictions: opportunities and challenges for AI safety. In *SafeAI @ AAAI*.

Souillard-Mandar, W., Davis, R., Rudin, C., Au, R., Libon, D., Swenson, R., Price, C., Lamar, M., and Penney, D. (2016). Learning classification models of cognitive conditions from subtle behaviors in the digital clock drawing test. *Machine Learning*, 102(3):393–441.

Street, W. (2005). Oblique multicategory decision trees using nonlinear programming. *INFORMS Journal on Computing*, 17(1):25–31.

Strobl, C., Boulesteix, A.-L., Kneib, T., Augustin, T., and Zeileis, A. (2008). Conditional variable importance for random forests. *BMC Bioinformatics*, 9(1):307.

Strzalkowska-Kominiak, E. and Romo, J. (2021). Censored functional data for incomplete follow-up studies. *Statistics in Medicine*, 40(12):2821–2838.

Su, X., Wang, M., and Fan, J. (2004). Maximum likelihood regression trees. *Journal of Computational and Graphical Statistics*, 13(3):586–598.

Suárez, A. and Lutsko, J. F. (1999). Globally optimal fuzzy decision trees for classification and regression. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(12):1297–1311.

Therneau, T., Atkinson, B., and Ripley, B. (2015). *rpart: Recursive Partitioning and Regression Trees*. R package version 4.1-10.

Truong, A. (2009). *Fast growing and interpretable oblique trees via logistic regression models*. Ph.D. thesis, University of Oxford, UK.

Turney, P. (1995). Cost-sensitive classification: Empirical evaluation of a hybrid genetic decision tree induction algorithm. *Journal of Artificial Intelligence Research*, 2:369–409.

Ustun, B. and Rudin, C. (2016). Supersparse linear integer models for optimized medical scoring systems. *Machine Learning*, 102(3):349–391.

Václavík, R., Novák, A., Sucha, P., and Hanzálek, Z. (2018). Accelerating the branch-and-price algorithm using machine learning. *European Journal of Operational Research*, 271(3):1055–1069.

Van Vlasselaer, V., Eliassi-Rad, T., Akoglu, L., Snoeck, M., and Baesens, B. (2017). GOTCHA! Network-based fraud detection for social security fraud. *Management Science*, 63(9):3090–3110.

Verhaeghe, H., Nijssen, S., Pesant, G., Quimper, C.-G., and Schaus, P. (2020). Learning optimal decision trees using constraint programming. *Constraints*, 25(3):226–250.

Verma, S., Dickerson, J., and Hines, K. (2020). Counterfactual explanations for machine learning: A review. *arXiv preprint arXiv:2010.10596*.

Verwer, S. and Zhang, Y. (2017). Learning decision trees with flexible constraints and objectives using integer optimization. In Salvagnin, D. and Lombardi, M., eds., *Integration of AI and OR Techniques in Constraint Programming: 14th International Conference, CPAIOR 2017, Padua, Italy, June 5-8, 2017, Proceedings*, pages 94–103.

Verwer, S., Zhang, Y., and Ye, Q. (2017). Auction optimization using regression trees and linear models as integer programs. *Artificial Intelligence*, 244:368–395.

Verwer, S., Zhang, Y., and Ye, Q. (2019). Learning optimal classification trees using a binary linear program formulation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 1625–1632.

Vidal, T. and Schiffer, M. (2020). Born-again tree ensembles. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119, pages 9743–9753.

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S. J., Brett, M., Wilson, J., Millman, K. J., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., Carey, C. J., Polat, İ., Feng, Y., Moore, E. W., VanderPlas, J., Laxalde, D., Perktold, J., Cimrman, R., Henriksen, I., Quintero, E. A., Harris, C. R., Archibald, A. M., Ribeiro, A. H., Pedregosa, F., van Mulbregt, P., and SciPy 1.0 Contributors (2020). SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272.

Wächter, A. and Biegler, L. T. (2006). On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical Programming*, 106(1):25–57.

Wager, S. and Athey, S. (2018). Estimation and inference of heterogeneous treatment effects using random forests. *Journal of the American Statistical Association*, 113(523):1228–1242.

Wang, J.-L., Chiou, J.-M., and Müller, H.-G. (2016). Functional data analysis. *Annual Review of Statistics and Its Application*, 3:257–295.

Wang, P., Fan, E., and Wang, P. (2021). Comparative analysis of image classification algorithms based on traditional machine learning and deep learning. *Pattern Recognition Letters*, 141:61–67.

Weston, J., Elisseeff, A., Schölkopf, B., and Tipping, M. (2003). Use of the zero norm with linear models and kernel methods. *Journal of Machine Learning Research*, 3:1439–1461.

Wickramarachchi, D., Robertson, B., Reale, M., Price, C., and Brown, J. (2016). HHCART: An oblique decision tree. *Computational Statistics & Data Analysis*, 96:12–23.

Yang, L., Liu, S., Tsoka, S., and Papageorgiou, L. (2017). A regression tree approach using mathematical programming. *Expert Systems with Applications*, 78:347–357.

Yang, Y., Morillo, I. G., and Hospedales, T. (2018). Deep neural decision trees. *arXiv preprint arXiv:1806.06988*.

Yu, J., Ignatiev, A., Stuckey, P., and Le Bodic, P. (2020). Computing Optimal Decision Sets with SAT. *arXiv preprint arXiv:2007.15140*.

Zafar, M., Valera, I., Gomez Rodriguez, M., and Gummadi, K. (2017). Fairness constraints: Mechanisms for fair classification. In *Artificial Intelligence and Statistics*, pages 962–970. PMLR.

Zantedeschi, V., Kusner, M., and Niculae, V. (2020). Learning binary trees via sparse relaxation. *arXiv preprint arXiv:2010.04627*.

Zeng, J., Ustun, B., and Rudin, C. (2017). Interpretable classification models for recidivism prediction. *Journal of the Royal Statistical Society: Series A*, 180(3):689–722.

Zhu, H., Murali, P., Phan, D., Nguyen, L., and Kalagnanam, J. (2020). A scalable MIP-based method for learning optimal multivariate decision trees. *Advances in Neural Information Processing Systems*, 33:1771–1781.