Inaugural – Dissertation

zur

Erlangung der Doktorwürde

der

Gesamtfakultät für Mathematik, Ingenieur- und Naturwissenschaften

der

Ruprecht – Karls – Universität

Heidelberg

vorgelegt von

Meyer, Joseph Theo

aus London

Statistical Learning for Structured Models: Tree Based Methods and Neural Networks

Betreuer : Prof. Dr. Enno Mammen

Abstract

In this thesis, estimation in regression and classification problems which include low dimensional structures are considered. The underlying question is the following. *How well do statistical learning methods perform for models with low dimensional structures?* We approach this question using various algorithms in various settings. For our first main contribution, we prove optimal convergence rates in a classification setting using neural networks. While non-optimal rates existed for this problem, we are the first to prove optimal ones. Secondly, we introduce a new tree based algorithm we named random planted forest. It adapts particularly well to models which consist of low dimensional structures. We examine its performance in simulation studies and include some theoretical backing by proving optimal convergence rates in certain settings for a modification of the algorithm. Additionally, a generalized version of the algorithm is included, which can be used in classification settings. In a further contribution, we prove optimal convergence rates for the local linear smooth backfitting algorithm. While such rates have already been established, we bring a new simpler perspective to the problem which leads to better understanding and easier interpretation. Additionally, given an estimator in a regression setting, we propose a constraint which leads to a unique decomposition. This decomposition is useful for visualising and interpreting the estimator, in particular if it consits of low dimenional structures.

Zusammenfassung

In dieser Arbeit betrachten wir Regressions- sowie Klassifikationsprobleme die auf niedrigdimensionalen Strukturen beruhen. Wir interessieren uns für folgende Frage. *Wie gut schneiden Methoden des statistischen Lernens in Modellen mit niedrigdimensionalen Strukturen ab?* Um an diese Frage heranzutreten, untersuchen wir verschiedene Algorithmen in unterschiedlichen statistischen Modellen. Als erstes Hauptresultat zeigen wir optimale Konvergenzraten in einem Klassifikationsmodell für einen Schätzer der auf Neuronalen Netzen basiert. Suboptimale Raten existieren bereits für dieses Problem. Wir sind hingegen die ersten, die optimale Raten in diesem Problem für einen Schätzer, der auf Neuronalen Netzen beruht, beweisen. Ein weiterer wesentlicher Beitrag besteht in der Einführung eines neuen Baum-basierten Alorithmus den wir "Random Planted Forest" getauft haben. Dieser ist insbesondere für Modelle, die aus niedrigdimensionalen Strukturen bestehen, konzipiert. Wir evaluieren die Schätzungen, die der Algorithmus hervorbringt, anhand von Simulationsstudien und schaffen eine theoretische Grundlage für eine leicht abgeänderte Version des Algorithmus. Wir geben auch eine verallgemeinerte Version an, die unter anderem für Klassifikationsprobleme verwendet werden kann. In einem zusätzlichen Resultat beweisen wir optimale Konvergenzraten für den "local linear smooth backfitting" Schätzer. Solche Raten wurden für diesen Algorithmus bereits gezeigt. Wir betrachten das Problem aus einer neuen Perspektive, die eine neue Interpretation des Schätzers zulässt und die Beweise vereinfacht. Des weiteren geben wir für einen beliebigen Schätzer in einem Regressionsproblem eine Bedingung an, durch die man eine eindeutige Zerlegung des Schätzers erhält. Diese Zerlegung ist hilfreich um den Schätzer zu visualisieren und zu interpretieren, insbesondere wenn niedrigdimensionale Strukturen vorherrschen.

Acknowledgments

First and foremost, I would like to thank my supervisor Enno Mammen for signing me as a doctoral student and supporting me throughout my doctorate. Your calm attitude has been pleasant and with your help, I always felt I could freely pursue my goals and interests. I also thank the graduate college RTG 1953 for financing my projects as well as enabling interesting seminars and workshops. Next, I would like to thank Munir Hiabu, Marvin Wright, Lukas Burk and Ricardo Blum for the time we worked together on different projects. I thoroughly appreciate working with other people and was happy to find this small group of individuals. It has been a pleasure working with all of you. Moreover, I specially thank Munir Hiabu for many many zoom meetings and for lots of advice throughout my doctorate. I also thank Philipp Tepel and Maike Spankus for assisting in programming regarding projects I am involved in. Considering relations not directly tied to the university, special thanks goes to Dario Weißmann, Johann Franke, Kira Hülshoff and Lukas Gärttner who have supported me in many aspects throughout my time in the doctorate. I can not name everyone, but would like to thank all other friends just as much! You have helped me throughout my life. This includes having fun, cheering each other up and going on adventures together. It provides the energy and enthusiasm required for accomplishing a doctorate or anything else. Lastly, I would like to thank my family. In particular, my sister Helen Meyer, my father Paul Meyer and my mother Celia Brown. You always have my back and enable me to feel loved and appreciated no matter what I do. This gives me the confidence I require and the feeling, that I do not have to worry about anything.

Contents

1 Introduction **13**

1.1 Bounded Interaction Models 15

1.2 Algorithms 16

1.2.1 Backfitting 16

1.2.2 Neural Networks 18

1.2.3 Tree Based Algorithms 20

1.3 Outline 23

2 Local Linear Smoothing **25**

2.1 Introduction 25

2.2 Local Linear Smoothing in Additive Models 27

2.3 Existence and Uniqueness of the Estimator, Convergence of the Algorithm 31

2.4 Asymptotic Properties of the Estimator 38

3 Classification using Neural Networks **47**

3.1 Introduction 47

3.1.1 Contribution 49

3.1.2 Outline 49

3.1.3 Notation 49

3.2 General Convergence Results 50

3.2.1 Classification Setup 50

3.2.2 Consistency Results 51

3.3 Convergence Rates for Neural Networks 53

3.3.1 Definitions Regarding Neural Networks 54

3.3.2 Conditions on the Bayes-Rule 55

3.3.3 Main Theorems 57

3.3.4 Results for Regular Boundaries 58

3.4 Lower Bound 61

3.5 Concluding Remarks 62

4 Random Planted Forest: A Directly Interpretable Tree Ensemble **65**

4.1 Introduction 65

4.2 The Algorithm 70

4.2.1 Calculating a Split 72

4.2.2 Planted Tree Family 72

4.2.3 From a Tree Family to a Forest . . . 73
4.3 Discussion of Parameters . . . 74
4.3.1 t_try . . . 74
4.3.2 split_try . . . 75
4.3.3 Driving Parameters . . . 75
4.4 Simulations . . . 75
4.5 Theoretical Properties . . . 79
4.6 Generalized RPF . . . 84
4.6.1 Remarks . . . 85
4.6.2 Least Squares Regression . . . 87
4.6.3 L1 Loss . . . 88
4.6.4 Logit Link Function with Maximum Likelihood . . . 88
4.6.5 Logit Link Function with Exponential Loss . . . 89
4.6.6 General RPF Simulations . . . 91
4.7 Conclusion . . . 94

5 Decomposition of Low Dimensional Structures 95
5.1 Introduction . . . 95
5.1.1 Motivating Example . . . 97
5.1.2 Related Work . . . 98
5.2 Main Result . . . 99
5.3 Truly Global Interpretation . . . 99
5.3.1 Describing SHAP Values . . . 101
5.3.2 Describing Partial Dependence Plots . . . 102
5.3.3 A Causal Application . . . 102
5.4 Feature Importance . . . 103
5.5 Experiments . . . 103
5.5.1 Truly Global Explanations . . . 103
5.5.2 Feature Importance . . . 105
5.5.3 Post-Hoc Feature Removal . . . 105
5.6 Concluding Remarks and Limitations . . . 106

6 Additional Remarks 109
6.1 Random Planted Forest . . . 109
6.1.1 Additive Random Planted Forests . . . 109
6.1.2 Identification Constraint . . . 110
6.1.3 Additional Remarks . . . 111
6.2 Decomposition of Low Dimensional Structures . . . 111
6.2.1 SHAP Values . . . 111
6.2.2 Shapley Axioms . . . 112
6.2.3 Connecting a General Truly Global Expansion to SHAP Values . . . 112

6.2.4 Functional Decomposition of SHAP Values from Low-Dimensional Tree Structures . . . 113

7 Proofs 117

7.1 Local Linear Smoothing . . . 117

7.1.1 Projection Operators . . . 117

7.1.2 Proofs of Propositions 2.3.4 and 2.3.6 . . . 118

7.2 Classification using Neural Networks . . . 120

7.2.1 General Convergence Results . . . 120

7.2.2 Convergence Rates for Neural Networks . . . 127

7.2.3 Lower Bound . . . 141

7.3 Random Planted Forest . . . 148

7.4 Decomposition of Low Dimensional Structures . . . 157

7.4.1 Lemmata . . . 157

7.4.2 Proofs . . . 159

8 Simulation Results 161

8.1 Random Planted Forest . . . 161

8.1.1 Simulations Random Planted Forest . . . 161

8.1.2 Generalized RPF . . . 168

8.2 Decomposition of Low Dimensional Structures . . . 173

8.2.1 Motivating Example . . . 173

8.2.2 Truly Global Explanations . . . 173

8.2.3 Feature Importance . . . 175

8.2.4 Post-hoc Feature Removal . . . 175

1 Introduction

In this thesis, we contribute to the theoretical literature considering statistical learning algorithms in settings with high dimensional data with low dimensional structures. Additionally, we introduce and elaborate on a novel algorithm called random planted forests which is particularly efficient in such settings.
The interest in statistical learning theory has steadily increased in the last few decades. Due to technical advancements, computational capabilities have led to researchers dealing with huge data sets including high dimensional data. After defining a response variable, there are many different approaches for estimation within such data sets. Most prevalent, artificial neural networks and tree based estimators such as gradient boosting [37] or random forests [13] have shown to be particularly useful in practice. These algorithms are often referred to as black box methods for the following two reasons. First of all, there is a lack of rigorous mathematical explanations for why the methods are so successful. Second, resulting estimators are difficult to interpret and visualize.
Finding theoretical explanations for the success of black box algorithms is a wide field of research. Theoretically approaching them has turned out to be quite challenging. Among others, [6, 94, 127] have contributed by establishing convergence rates as well as asymptotic normality for modifications of the random forests algorithm. Similarly, [74] and [109] prove consistency results for neural networks in specific settings. While these findings have had considerable impact in the statistical literature, they do not use the respective algorithms which are commonly applied in practice. In both cases, this is due to technical issues which arise from non-standard ways in which the algorithms unfold and use the input data. In the case of neural networks, additional complications come from the fact that there are many variants and many hyperparameters involved. The approach from researchers to tackle these problems is as follows. One begins by proving results for a oversimplified version of the algorithms in a simple statistical setting. Iteratively, using former insights, one implements new ideas to prove theorems which correspond to more complicated settings using algorithms which resemble the ones used in practice more accurately. Using this strategy, one hopes to achieve the goal of truly proving or disproving the efficiency of the algorithms in question.
After estimating a function with an algorithm, many practitioners consider interpreting and visualising their findings. For estimators resulting from algorithms such as random forests and neural networks, it is not straight forward how to do so. In the literature, there have been some ideas. Considering neural networks [85] suggest a method pointing out the most relevant features for a decision made by the estimator. Similarly, variable importance measures are highly used for random forests, see [13, 47, 118]. Other methods include partial dependence plots [37]

and so called shapley values [111] which are motivated from game theory.
In this thesis, we consider models based on low dimensional structures, which are described in more detail in Section 1.1. These circumvent the curse of dimensionality. Additionally, if a model is based on structures of a maximum dimension of two, the results can be visualised and described by partial dependence plots. We are interested in the following question. *How well do statistical learning methods perform for models with low dimensional structures?* Our main contributions include the following.

- First of all, in Chapter 3 we prove optimal convergence rates up to a log term in a classification setting using neural networks. One of the main driving factors for the convergence rates we obtain are the assumptions on the boundaries of sets we consider for approximation. We point out that the main theorem (Theorem 3.3.4) is applicable to classes of sets with other types of boundaries, which can lead to other convergence rates. Additionally, we provide new insights on a specific statistical setting regarding an assumption sometimes referred to as the Tsybakov noise condition. These insights can be useful independent of the usage of neural networks.

- Secondly, in Chapter 4 we introduce a new tree based algorithm which we named random planted forests. It is designed to adapt particularly well to data which can be described mainly by low dimensional structures. In particular, the resulting estimator satisfies the low dimensionality condition (BI) from Section 1.1 if specified beforehand. Thus, the results are easily visualisable if we specify that the model is based on at most two dimensional structures. In simulation studies, we show that the algorithm performs well in regression as well as classification settings using artificial as well as real world data. Additionally, we prove optimal convergence rates for certain settings for a simplified version of the algorithm. We conjecture that these convergence rates also hold for the true algorithm.

Additional contributions include the following.

- In Chapter 2 we establish a new proof for optimal convergence rates for local linear smooth backfitting. It provides a new interpretation of the algorithm as a projection of the data onto a linear space. Additionally, it simplifies the arguments necessary significantly.

- Lastly, in Chapter 5 we introduce a marginal constraint which leads to a unique decomposition of a given estimator. The decomposition allows for simple calculation of partial dependence plots and provides a global interpretation of the function. While it is known as Harsanyi dividend in game theory and as Möbius inverse in combinatorics, we bring a new perspective to the decomposition and suggest using it in practice for interpreting results from low dimensional structures. In particular, we implemented an open source efficient algorithm for xgboost and random planted forest.

In the remainder of the introduction, we specify what we consider to be low dimensional structures by introducing a bounded interaction (BI) assumption. Additionally, we introduce some algorithms which are central to our work and provide a short outline of this thesis.

1.1 Bounded Interaction Models

We are handed i.i.d. data $(X_i, Y_i)_{i=1}^n$ with $X_i \in D \subseteq \mathbb{R}^d$, $Y_i \in \mathbb{R}$ and

$$Y_i = m(X_i) + \varepsilon_i,$$

with $\mathbb{E}[\varepsilon_i \,|X_i] = 0$ and $m : D \to \mathbb{R}$. The goal is to find an estimator $\widehat{m}$ which converges to m in some sense. In order to do so, we set some conditions for m. A typical approach is to impose smoothness assumptions such as Hölder continuity with smoothness parameter β on m. Under some moment conditions for ϵ, it is known that the optimal convergence rate with respect to the L_2-norm is $n^{-\frac{\beta}{2\beta+d}}$ [116]. This rate suffers from the so called curse of dimensionality, meaning the rate decreases exponentially for large input dimension d. There have been many suggestions for conditions on m which enable rates of convergence which do not depend on d. One approach is to impose a sparsity constraint. The idea is that the function m does not depend on all input coordinates x_i with $i \in \{1, \ldots, d\}$ but only on a small (unknown) subset with indices in $S \subseteq \{1, \ldots, d\}$ with $q := |S| << d$. This assumption is used for example in parametric models such as linear regression, where LASSO [124] and RIDGE [58] estimates are common approaches to the problem. In this thesis, we focus on settings where m satisfies the bounded interaction (BI) assumption.

Assumption (BI) The regression function m is of the form

$$m(x) = \sum_{S \subseteq \{1,\ldots,d\},\ |S| \leq q} m_S(x_S), \tag{1.1.1}$$

where $q \in \{1, \ldots, d\}$ is the maximum order of interaction and $m_S : \mathbb{R}^{|S|} \to \mathbb{R}$ are functions.

Thus, we assume that a maximum of q coordinates may interact in m. Note that given m the right hand side of (1.1.1) is not unique. Assumption (BI) is a generalisation of the sparsity assumption. Instead of saying m may only depend on all coordinates with indices in a subset $S \subseteq \{1, \ldots, d\}$ with $|S| = q$, we say that m can be a sum of such functions. In this thesis, we consider a variety of estimators and various variations of the (BI) assumption. While we consider additive models in Chapter 2 which corresponds to (BI) with $q = 1$, we consider higher order interactions ($q > 1$) in Chapter 5 as well as Chapter 4. Furthermore, we also work with classification settings in this thesis. There, we impose (BI) indirectly. In Chapter 3, we approach a binary classification setup ($Y \in \{0, 1\}$) by directly estimating the set

$$G^* = \Big\{ x \in D \;\Big|\; \mathbb{P}(Y = 1|X = x) \geq \frac{1}{2} \Big\}$$

instead of the function m. Intuitively, we assume that the boarder ∂G^* acts like a function satisfying a generalisation of (BI). Lastly, in Section 4.6, we simply assume there is a link function σ such that

$$\mathbb{P}(Y = 1|X = x) = \sigma(m(x))$$

and m satisfies (BI).

1.2 Algorithms

As discussed above, in non-parametric setups many different algorithms are considered for theoretical approaches as well as in practice. In this section, we describe three different algorithm classes which are used in this thesis. In Subsection 1.2.1 we introduce backfitting algorithms which are used mostly in additive models, i.e. if (BI) holds with $q = 1$. Later, we provide new insights on proving optimal convergence rates for a specific method called the local linear smooth backfitting estimator. Neural networks are considered in Subsection 1.2.2. The first of two major contributions of this thesis is proving optimal convergence rates in a classification setting where neural networks are used. Lastly, the other main contribution of this thesis is the introduction of random planted forests. This is a novel tree based estimation procedure. The basics of tree based algorithms are described in Subsection 1.2.3.

1.2.1 Backfitting

Backfitting was introduced in [15] for additive models. Theoretical results and variants followed in [87, 99, 100], among others. A generalisation is given in [134], where the true function to be estimated may include a link function to an additive model instead of being additive itself. Backfitting algorithms have shown to approximate additive functions well, even in high dimensions. Additionally, in contrast to e.g. random forests or neural networks, they typically provide estimators which are additive themselves, which allows for easy visualisation and interpretation.

We begin by describing the structure of a backfitting algorithm. Let $\mathcal{F}$ be a class of functions $f : \mathbb{R} \to \mathbb{R}$ and let

$$\widehat{f} : \mathbb{R}^{2n} \to \mathcal{F}$$

be an estimator. We assume (BI) holds for $q = 1$ and thus

$$m(x) = \sum_{k=1}^{p} m_k(x_k),$$

where we omit the term $m_\emptyset$. We then use the procedure given in Algorithm 1. The intuition why this estimator provides reasonable results is as follows, which is motivated by Section A.4 in [9]. Let H be a Hilbert space and $H_1, \dots, H_s \subseteq H$ be subspaces of H. Define $H_+ := H_1 + \cdots + H_s$ and let $P_+, P_1, \dots, P_s$ be the orthogonal projection operators onto $H_+, H_1, \dots, H_s$ respectively. Define an initial value $h \in H$. Regarding Algorithm 1, h typically corresponds to the data Y and $H_1, \dots, H_s$ are function spaces which include the one dimensional functions m_k. Thus, H_+ includes m. Assume $\widehat{f}((X_{ik}, Y_i)_{i=1}^n)$ corresponds to the projection operator P_k. A template of

Algorithm 1 Template Backfitting Algorithm

```
 1: Input: $(X_i, Y_i)_{i=1}^n$
 2: Start: $\widehat{m}_k \equiv 0$, $error = \infty$                ▷ $k = 1, \dots, s$
 3: while $error > tolerance$ do
 4:     $error \leftarrow 0$
 5:     for $k = 1, \dots, s$ do
 6:         $\widehat{m}_k^{\text{old}} \leftarrow \widehat{m}_k$
 7:         $\widehat{R}_i \leftarrow Y_i - \sum_{j \neq k} \widehat{m}_j(X_{ij})$
 8:         $\widehat{m}_i \leftarrow \widehat{f}((X_{ik}, R_i)_{i=1}^n)$
 9:         $error \leftarrow error + d(\widehat{m}_k, \widehat{m}_k^{\text{old}})$
10: return $\widehat{m} = \sum_{j=1}^s \widehat{m}_j$
```

the rewritten algorithm is given in Algorithm 2. Under certain assumptions, the backfitting algorithm approximates the orthogonal projection $P_+(h)$ using only the projections $P_1, \dots, P_s$. To see this, note that in the first step, we obtain $\widehat{h}_1 = P_1(h)$. We have $\widehat{h}_2 = P_2(h - P_1(h))$. Now, note that $h - P_1(h)$ is the orthogonal projection onto $H_1^\perp$. Next, $\widehat{h}_3 = P_3(h - P_1(h) - P_2(h - P_1(h)))$ holds. Observe that $h - P_1(h) - P_2(h - P_1(h))$ is the projection $h - P_1(h)$ onto $H_2^\perp$. By iteratively following these steps, we can see that in every iteration the algorithm projects the previous result onto $H_k^\perp$ for some $k = 1, \dots, s$. One can now show that $h - \sum_{j \neq k} \widehat{h}_j$ converges to the orthogonal Projection of h onto $H_1^\perp + \dots + H_s^\perp$. The assertion then follows by noting that $(H_1^\perp + \dots + H_s^\perp)^\perp = H_+$, if H_+ is a closed subset of H.

Algorithm 2 Template Backfitting Algorithm

```
1: Start: $h \in H$, $\widehat{h}_k \equiv 0$, $error = \infty$                ▷ $k = 1, \dots, s$
2: while $error > tolerance$ do
3:     $error \leftarrow 0$
4:     for $k = 1, \dots, s$ do
5:         $\widehat{h}_k^{old} \leftarrow \widehat{h}_k$
6:         $\widehat{h}_k \leftarrow P_k(h - \sum_{j \neq k} \widehat{h}_j)$
7:         $error \leftarrow error + |\widehat{h}_k - \widehat{h}_k^{old}|$
8: return $\widehat{h} = \sum_{j=1}^s \widehat{h}_j$
```

Advantages of backfitting are the following

- Given $\widehat{f}$, the algorithm is simple to explain and to implement.
- For additive models, the algorithm shows promising results.
- For many cases the algorithm has theoretical backing, meaning convergence results exist for the true algorithm implemented.

Problems with backfitting include the following:

- Backfitting algorithms are typically not applicable if (BI) is not satisfied for $q = 1$, but for $q > 1$.

- The algorithms do not adapt well to sparsity.

We use a variant of the backfitting algorithm in Chapter 2. Additionally, the theoretical results in Chapter 4 use ideas relating to backfitting.

1.2.2 Neural Networks

Artificial neural networks have shown astonishing results in various fields, which include disease detection [70, 79], speech recognition [57] and image recognition [40] among many others. While simulations suggest that neural networks outperform other algorithms in high dimensional settings, theoretical backing is lacking due to many technical complications as well as the variety of neural networks used for different tasks.
In this thesis, we consider feedforward deep neural networks and leave out the word "feedforward" when referring to them. These networks have been studied theoretically for a long time. They were first brought up in [93]. Universal approximation theorems where found in [29] and [61]. In order to find convergence rates, one typically requires more sophisticated approximation results, which specify the structures of neural networks required to approach certain classes of functions. The paper [104] establishes such results for classes we are interested in. Using similar findings, [74, 109] provide convergence rates for different statistical settings using different types of neural networks. Note that the results obtained in these papers do not include networks which are typically used in practice. Among others, the results above where improved upon in [75] which use slightly more natural constraints. We are interested in providing similar results in a different setting. This subsection is used to shortly define neural networks. We use the definition and explanation below from [96].

Definition 1.2.1. *Let $L, z_0, \dots, z_{L+1} \in \mathbb{N}$. For $i = 1, \dots, L$, let σ_i be a function*

$$\sigma_i : \mathbb{R} \to \mathbb{R}.$$

For $b = (b_1, \dots, b_{z_i}) \in \mathbb{R}^{z_i}$, define a shifted z_i-dimensional version of σ_i by

$$\sigma_{i,b} : \mathbb{R}^{z_i} \to \mathbb{R}^{z_i}, \ \sigma_{i,b}(y_1, \dots, y_{z_i}) = \big(\sigma_i(y_1 - b_1), \dots, \sigma_i(y_{z_i} - b_{z_i})\big).$$

A neural network with network architecture

$$\big(L, (z_0, \dots, z_{L+1}), (\sigma_1, \dots, \sigma_L)\big)$$

is a sequence

$$\Phi := \big(W_1, b_1, \dots, W_L, b_L, W_{L+1}\big)$$

where each $W_s \in \mathbb{R}^{z_s \times z_{s-1}}$ is a weight matrix and $b_s \in \mathbb{R}^{z_s}$ is a shift vector. The realization of a neural network Φ on a set $D \subseteq \mathbb{R}^{z_0}$ is the function

$$R(\Phi) : D \to \mathbb{R}^{z_{L+1}}, \ R(\Phi)(x) = W_{L+1}\sigma_{L,b_L} W_L \cdots W_2 \sigma_{1,b_1} W_1 x.$$

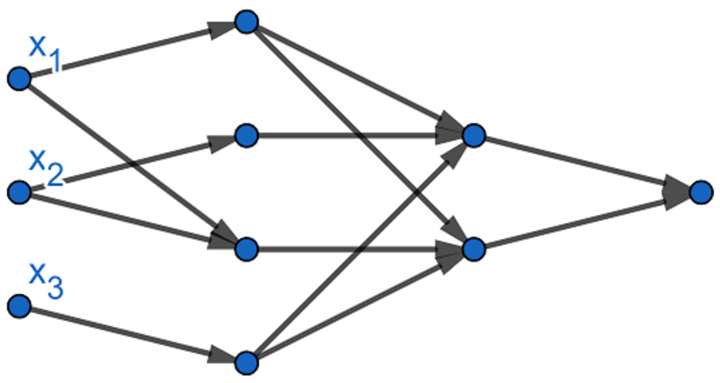


Figure 1.1: Illustration of a neural network with $L = 2$ hidden Layers. It represents a function $f : \mathbb{R}^3 \to \mathbb{R}$.

We denote by

$$\mathcal{N}_{L,z,\sigma} := \left\{ R(\Phi) \mid \Phi = \big(W_1, b_1, \ldots, W_{L+1}, b_{L+1}\ \big),\ W_s \in \mathbb{R}^{z_s \times z_{s-1}}, b_s \in \mathbb{R}^{z_s} \right\}$$

a set of realizations of neural networks where $z := (z_0, \ldots, z_{L+1}) \in \mathbb{N}^{L+2}$ *and* $\sigma := (\sigma_1, \ldots, \sigma_L)$. *We call* (L, z, σ) *the network architecture.*

Typically, for $i = 1, \ldots, L$ the function σ_i is called activation function and $\sigma_{i,b}$ is named shifted activation function. The constant L denotes the number of hidden Layers. The values $d := z_0$ and z_{L+1} are the input and output dimensions, respectively. The dimensions z_i are referred to as number of neurons in the i-th layer. For the sake of completeness, we note that a network with L=0 layers is of the form $\Phi := (W)$ for $W \in \mathbb{R}^{z_0 \times z_1}$ and has realization $R(\Phi)(x) = Wx$. Note that, in general, the weights of a neural network Φ, i.e. the entries of its shift vectors $(b_1, \ldots, b_{L+1})$ and weight matrices $(W_1, \ldots, W_{L+1})$ are not uniquely determined by its realization $R(\Phi)$. In the following, for brevity, we occasionally introduce a network by defining its realization. In such a case, it is clear from the presentation of the realization which precise neural network is considered. Figure 1.2.2 contains a typical illustration of a neural network. It can be interpreted as follows.

- Each node corresponds to a so called neuron of the neural network.
- The left most column corresponds to the input, the right to the output. Here, the input is 3 dimensional and the output 1 dimensional.
- The columns in the middle correspond to the hidden Layers. Here, we have 2 hidden Layers. The first has $z_1 = 4$, the second $z_2 = 2$ neurons.
- The arrows correspond to the weight matrices W_i. If there is no arrow between to neurons of adjacent columns, the corresponding weight is zero. Here e.g. $W_1 \in \mathbb{R}^{4\times 3}$ has 5 non-zero weights.

In order to use neural networks for statistical analysis, one typically defines an estimator $\widehat{m}$ of the regression function m by an "approximation" of

$$\arg\min_{f \in N'} V(f(X), Y), \tag{1.2.1}$$

where $N' \subseteq N_{L^n, z^n, \sigma}$ for sequences L^n, z^n and V is some loss function. This procedure has two main challenges.

- First of all, it is not clear how to define the sequences L^n, z^n as well as which additional conditions to impose on N'. As discussed above, practical approaches differ substantially from theoretical ones, which include sparsity constraints as well as bounding or discretizing the weights among others.
- For most choices of classes of neural networks N' and loss functions V, the term (1.2.1) cannot be reliably calculated since $V(f(X), Y)$ is highly non-convex in its parameters, which are typically very high dimensional. Thus, many different minimization algorithms are used to approximate the minimum such as gradient descent. However, it is not clear in which form these algorithms truly approximate this minimum or if they do something different such as getting stuck in local minima, see e.g. [30, 114, 121].

In Chapter 3 we define an estimator which is in the form (1.2.1). There, we do not discuss the problem of finding the minimum but assume it is given. Thus, our results are entirely theory based. Advantages of using neural networks for satistical problems are the following.

- They have provided extrodenary results in practice.
- They have a wide range of application.

Problems with neural networks include the following.

- Many different types of neural networks are used, such that a lot of fine tuning is required.
- For the algorithms used in practice, theoretical backing is lacking.
- The results are often hard to interpret.

1.2.3 Tree Based Algorithms

Tree based algorithms were popularised during the 1980s. On the one hand, there popularity comes from the fact, that the construction of a tree is usually very intuitive and easy to understand. On the other hand, algorithms such as gradient boosting [37] and random forests [13] have provided useful results in many research areas, such as genomic data analysis [22], network intrusion detection [68] and word recognition [62]. A tree in a tree based algorithm is typically of the form

$$\widehat{m}(x) = \sum_{j=1}^{p} a_j \mathbb{1}(x \in \mathcal{I}_j), \tag{1.2.2}$$

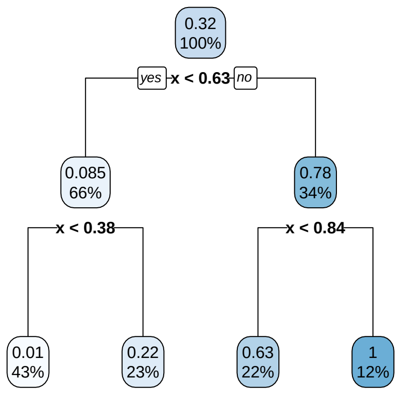


Figure 1.2: Illustration of a decision tree. Each node corresponds to a set. The root at the top corresponds to the set $\mathcal{I}^0 = D = [0, 1]$. The daughter cells of a node $\mathcal{I}$ form a partition of $\mathcal{I}$ obtained by splitting it at some point c. For example, the bottom left node corresponds to the set $\mathcal{I}_1^2 = [0, 0.38]$. The one next to it corresponds to $\mathcal{I}_2^2 = (0.38, 0.63]$.

where $a_j \in \mathbb{R}$ and $\mathcal{I}_j \subseteq \mathbb{R}^d$. The name "tree" comes from the idea, that the sets $\mathcal{I}_j$ are constructed by an iterative procedure, which can be visualized by a graph theoretical tree with a specified root node, see Figure 1.2 for an example. Algorithms differ in how the sets $\mathcal{I}_j$ and the values a_j are constructed. We explain the decision tree algorithm used in e.g. random forests as well as gradient boosting as it is the most common example. Here, the $\mathcal{I}_j$ are hypercubes which form a partition of the domain D of m. Define $\mathcal{I}_1^0 = D$. In step s, we go through a list of all sets $\mathcal{I}_1^{s-1}, \dots \mathcal{I}_{2^{s-1}}^{s-1}$ and obtain a list $\mathcal{I}_1^s, \dots \mathcal{I}_{2^s}^s$ as follows. Given $\mathcal{I}_j^{s-1}$ we define a coordinate $k \in \{1, \dots, d\}$ as well as a split point $c \in \mathbb{R}$ in order to set

$$\mathcal{I}_{2j-1}^s = \{x \in \mathcal{I}_j^{s-1} \mid x_k \leq c\}, \quad \mathcal{I}_{2j}^s = \{x \in \mathcal{I}_j^{s-1} \mid x_j > c\}.$$

This procedure is illustrated in Figure 1.2. The root node at the top corresponds to $I_1^0 = D$. Each row of nodes corresponds to a partition of D. Here, $s = 2$ and $I_1^s, \dots, I_{2^s}^s$ are represented by the nodes in the bottom row. The values a_{2j-1}^s and a_{2j}^s are given by

$$a_{2j-1}^s = \frac{\sum_{X_i \in I_{2j-1}^s} Y_i}{\sum_{X_i \in I_{2j-1}^s} 1}, \quad a_{2j}^s = \frac{\sum_{X_i \in I_{2j}^s} Y_i}{\sum_{X_i \in I_{2j}^s} 1}.$$

Figure 1.3 shows the development of the estimator corresponding to Figure 1.2. Observe that the number of subsets in the partition correspond to the number of nodes in the bottom row.

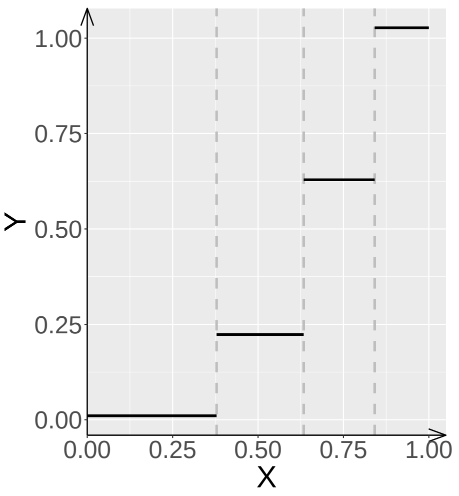


Figure 1.3: Resulting estimator $\widehat{m}$ of the tree algorithm which is represented in Figure 1.2.

The coordinate j and split point c are chosen by the so called CART criteria

$$(k,c) \in \arg\min_{k',c'} \sum_{X_i \in \mathcal{I}^s_{2j-1}} (a^s_{2j-1} - Y_i)^2 + \sum_{X_i \in \mathcal{I}^s_{2j}} (a^s_{2j} - Y_i)^2, \tag{1.2.3}$$

where $\mathcal{I}^s_{2j-1}, \mathcal{I}^s_{2j}, a_{2j-1}, a_{2j}$ on the left hand side are constructed from $\mathcal{I}^{s-1}_j$ using k', c'. There are many different conditions used for stopping the iteration. For simplicity, we just stay it stops after $S \in \mathbb{N}$ steps where S is given beforehand. Pseudo-code of this procedure is given in Algorithm 3. To complete a tree based algorithm, many different extensions are possible.

Algorithm 3 CART Decision Tree

```
1: **Input** $(Y_1, X_1), \dots, (Y_n, X_n)$
2: $\mathcal{I}_1 = D; \quad p = 1$
3: **for** $s = 1, \dots, S$ **do**
4:     **for** $j = p, \dots, 1$ **do**
5:         **Calculate** $k, c$ by (1.2.3)
6:         $\mathcal{I}_{2j-1} = \{x \in \mathcal{I}_j \mid x_k \le c\}; \quad \mathcal{I}_{2j} = \{x \in \mathcal{I}_j \mid x_j > c\}$
7: **for** $j = 1, \dots, p$ **do**
8:     $a_{2j-1} = \frac{\sum_{X_i \in I_{2j-1}} Y_i}{\sum_{X_i \in I_{2j-1}} 1}; \quad a_{2j} = \frac{\sum_{X_i \in I_{2j}} Y_i}{\sum_{X_i \in I_{2j}} 1}$
9: **Output** $\widehat{m}(\cdot) = \sum_{j=1}^p a_j \mathbb{1}(\cdot \in \mathcal{I}_j)$
```

For example, the random forest algorithm [13] creates B bootstrap samples and creates an estimator $\widehat{m}^b$ with a CART decision tree with some minor adjustments on each of these samples $b = 1, \dots, B$. The resulting estimator is given by $\widehat{m}(x) = \frac{1}{B}\sum_{b=1}^B \widehat{m}^b$. The gradient boosting algorithm [37] creates B estimators $\widehat{m}^b$ based on small trees (S is small) which are created by replacing Y_i with the current residual $R_i = Y_i - \sum_{b=1}^{b_0-1} \widehat{m}^b(X_i)$ for the tree with index b_0. The resulting estimator is given by $\widehat{m}(x) = \sum_{b=1}^B \widehat{m}^b(x)$.
While the algorithms are popular for their intuitive explanation, theoretically some technical

problems arise. We shortly explain a challenge in proving convergence rates as well as asymptotic normality for random forests, since it comes up in our analysis of random planted forests as well. The difficulty mentioned comes from the fact that the data is used for the creation of the leaves as well as the estimator given in each of the leaves at the end. This double use of data creates technical problems, which are not easy to circumvent. In [6, 94, 127], stylized versions of the forest algorithm in which the leaves are created without using the values Y_i of the data are used. In [127] an alternative procedure is suggested, in which one splits the data into two sets. One set is used to determine the leaves of a tree, the other is used to calculate the estimator. Recently, [24] was able to show consistency for the original random forests algorithm from [13]. Their analysis is complicated and the convergence rates are slow, but very interesting nonetheless. Generally, advantages of tree based algorithms are the following

- Trees are easy to implement and intuitive to explain.
- Tree based algorithms are widely applied in practice due to promising results in many areas.

Disadvantages include the following.

- For the algorithms used in practice, theoretical backing is lacking.

An additional disadvantage of most tree based algorithms is that they do not provide an estimator which satisfies (BI), even in the case where (BI) is satisfied. In Chapter 4 we describe a novel algorithm called random planted forests which is specifically designed to approximate low dimensional structures. By pre-defining q in (BI), the resulting estimator does in fact satisfy the assumption. Additionally, while the decomposition method introduced in Chapter 5 is theoretically applicable to any estimation procedure, it is particularly effective when the estimator to be decomposed comes from a tree based method which satisfies (BI) for small q. It has been implemented for xgboost [20] as well as random planted forests. It works well for these methods since it involves calculating many integrals which is particularly simple for estimators of the form (1.2.2).

1.3 Outline

We first introduce two novel theoretical results on topics regarding low dimensional structures. The first given in Chapter 2 shows optimal convergence rates in a regression setting for a backfitting algorithm when the underlying model is an additive model, i.e. (BI) is satisfied for $q = 1$. It follows the paper [55]. In Chapter 3, a classification setup is considered which is characterised by a restraint sometimes referred to as the Tsybakov noise condition. We show that almost optimal convergence rates can be achieved using deep neural networks. The intuition behind the approximation is to directly approximate optimal sets instead of approximating the corresponding regression function. A generalisation of (BI) is imposed on functions which represent a partition of the boundary of such optimal sets. The chapter follows the paper

[96]. In Chapter 4, we introduce a new algorithm named random planted forest. It is a tree based algorithm which adapts well to models with a low degree of interaction. In particular, in contrast to the usual random forest algorithm introduced in [13], one can specify the degree q beforehand. The resulting function then satisfies (BI) for q. The chapter follows the paper [54] and includes research on a generalisation of random planted forest which will be published in the future. Lastly, in Chapter 5, we provide a closed form representation for interaction terms of a function m which make the interaction terms in (BI) identifiable. For tree based procedures such as random forests [13], xgboost [20] and random planted forest introduced in Chapter 4, we can construct particularly simple algorithms to calculate these closed form representations. The algorithms are fast when the degree of interaction of m is low, which can be the case for xgboost as well as random planted forest. Additional remarks to further enhance the understanding of the projects are given in Chapter 6. We collected all proofs not included in the main text in Chapter 7 and some simulation results in Chapter 8.

2 Local Linear Smoothing in Additive Models as Data Projection

This chapter follows the paper [55]. We included some slight modifications in order to embed it into this thesis.

We discuss local linear smooth backfitting for additive nonparametric models. This procedure is well known for achieving optimal convergence rates under appropriate smoothness conditions. In particular, it allows for the estimation of each component of an additive model with the same asymptotic accuracy as if the other components were known. The asymptotic discussion of local linear smooth backfitting is rather complex because typically an overwhelming notation is required for a detailed discussion. We interpret the local linear smooth backfitting estimator as a projection of the data onto a linear space with a suitably chosen semi-norm. This approach simplifies both the mathematical discussion as well as the intuitive understanding of properties of this version of smooth backfitting.

2.1 Introduction

We consider local linear smoothing in an additive model

$$E[Y_i|X_i] = m_0 + m_1(X_{i1}) + \cdots + m_d(X_{id}), \tag{2.1.1}$$

where (Y_i, X_i) $(i = 1, \ldots, n)$ are i.i.d. observations with values in $\mathbb{R} \times \mathcal{X}$ for a bounded connected open subset $\mathcal{X} \subseteq \mathbb{R}^d$. Here, m_j $(j = 1, \ldots, d)$ are some smooth functions which we aim to estimate and $m_0 \in \mathbb{R}$. Below, we add norming conditions on $m_0, \ldots, m_d$ such that they are uniquely defined given the sum. In [87] a local linear smooth backfitting estimator based on smoothing kernels was proposed for the additive functions m_j. There, it was shown that their version of a local linear estimator $\widehat{m}_j$ of the function m_j has the same pointwise asymptotic variance and bias as a classical local linear estimator in the oracle model, where one observes i.i.d. observations (Y_i^*, X_{ij}) with

$$E[Y_i^*|X_{ij}] = m_j(X_{ij}), \quad Y_i^* = Y_i - \sum_{k \neq j} m_k(X_{ik}).$$

In this respect the local linear estimator differs from other smoothing methods where the asymptotic bias of the estimator of the function m_j depends on the shape of the functions m_k for $k \neq j$. An example for an estimator with this disadvantageous bias property is the local

constant smooth backfitting estimator which is based on a backfitting implementation of one-dimensional Nadaraya-Watson estimators. It is also the case for other smoothing estimators as regression splines, smoothing splines and orthogonal series estimators, where in addition also no closed form expression for the asymptotic bias is available. Asymptotic properties of local linear smoothing simplify the choice of bandwidths as well as the statistical interpretation of the estimators $\widehat{m}_j$. These aspects have made local linear smooth backfitting a preferred choice for estimation in additive models. Deriving asymptotic theory for local linear smooth backfitting is typically complicated by an overloaded notation that is required for detailed proofs. We use that the local linear smooth backfitting estimator has a nice geometric interpretation. This simplifies mathematical arguments and allows for a more intuitive derivation of asymptotic properties. In particular, we see that the estimator can be characterized as a solution of an empirical integral equation of the second kind as is the case for local constant smooth backfitting, see [92].
Our main point is that the local linear estimator can be seen as an orthogonal projection of the response vector $Y = (Y)_{i=1,\dots,n}$ onto a subspace of a suitably chosen linear space. A similar point of view is taken in [86] for a related construction where it was also shown that regression splines, smoothing splines and orthogonal series estimators can be interpreted as projection of the data in an appropriately chosen Hilbert space. Whereas this interpretation is rather straight forward for these classes of estimators it is not immediately clear that it also applies for kernel smoothing and local polynomial smoothing, see [86]. We introduce a new and simple view of local linear smoothing as data projection. In the next section we define the required spaces together with a corresponding semi-norm. We also introduce a new algorithm motivated by our interpretation of local linear smooth backfitting. The algorithm is discussed in Section 2.3. In Section 2.4 we see that our geometric point of view allows for simplified arguments for the asymptotic study of properties of the local linear smooth backfitting estimator.
The additive model (2.1.1) was first introduced in [38] and enjoys great popularity for two main reasons. The first is estimation performance. While not being as restrictive as a linear model, in contrast to a fully flexible model, it is not subject to the curse of dimensionality. Assuming that $E[Y_i|X_i = x]$ is twice continuously differentiable, the optimal rate of convergence of an estimator of $E[Y_i|X_i = x]$ is $n^{-2/(d+4)}$ if no further structural assumptions are made, see [116]. This means the rate deteriorates exponentially in the dimension of the covariates d. Under the additive model assumption (2.1.1) and assuming that each function m_j, $j = 1, \dots, d$ is twice continuously differentiable, the optimal rate of convergence is $n^{-2/5}$. The second reason is interpretability. In many applications it is desirable to understand the relationship between predictors and the response. Even if the goal is prediction only, understanding this relationship may help detect systematic biases in the estimator, so that out of sample performance can be improved or adjusted for. While it is almost impossible to grasp the global structure of a multivariate function m in general, the additive structure (2.1.1) allows for visualisation of each of the univariate functions, providing a comprehensible connection between predictors and the response.
Though the setting considered here is fairly simple, it can be seen as a baseline for more complicated settings. One main drawback is the additive structure which cannot account for interac-

tions between covariates. It is assuring however that even if the true model is not additive, the smooth backfitting estimator is still defined as the closest additive approximation. This is shown in the next section. If the true regression function is far away from an additive structure, then a more complex structure may be preferable. This could be done by adding higher-dimensional covariates, products of univariate functions or considering a generalized additive model. For testing procedures that compare such specifications, see also [48, 90]. Besides such structural assumptions, other directions the ideas developed here can be extended to are time-series data or high dimensional settings. Settings using more complicated responses like survival times, densities or other functional data may also be approached. Some of these cases have been considered, e.g., in [43, 45, 46, 53, 67, 88, 89, 90, 92, 134]. We hope that a better understanding of local linear estimation in this simple setting will help advance theory and methodology for more complicated settings in the future.

2.2 Local Linear Smoothing in Additive Models

The local linear smooth backfitting estimator

$$\widehat{m} = (\widehat{m}_0, \widehat{m}_1, \dots, \widehat{m}_d, \widehat{m}_1^{(1)}, \dots, \widehat{m}_d^{(1)})$$

is defined as the minimizer of the criterion

$$\begin{aligned} &S(f_0, \dots, f_d, f_1^{(1)}, \dots, f_d^{(1)}) \\ &\quad = n^{-1} \sum_{i=1}^n \int_{\mathcal{X}} \left\{ Y_i - f_0 - \sum_{j=1}^d f_j(x_j) - \sum_{j=1}^d f_j^{(1)}(x_j)(X_{ij} - x_j) \right\}^2 \\ &\quad \times K_h^{X_i}(X_i - x)\mathrm{d}x \end{aligned}$$

under the constraint

$$\sum_{i=1}^n \int_{\mathcal{X}} f_j(x_j) K_h^{X_i}(X_i - x)\mathrm{d}x = 0 \tag{2.2.1}$$

for $j = 1, \dots, d$. The minimization runs over all values $f_0 \in \mathbb{R}$ and all functions $f_j, f_j^{(1)} : \mathcal{X}_j \to \mathbb{R}$ with $\mathcal{X}_j = \{u \in \mathbb{R} : \text{there exists an } x \in \mathcal{X} \text{ with } x_j = u\}$. Under the constraint (2.2.1) and some conditions introduced in Section 2.3, the minimizer is unique. For $j = 1, \dots, d$ the local linear estimator of m_j is defined by $\widehat{m}_j$.

In the definition of S the function $K_h^u(\cdot)$ is a boundary corrected product kernel, i.e.,

$$K_h^u(u - x) = \frac{\prod_{j=1}^d \kappa\left(\frac{u_j - x_j}{h_j}\right)}{\int_{\mathcal{X}} \prod_{j=1}^d \kappa\left(\frac{u_j - v_j}{h_j}\right) \mathrm{d}v_j}.$$

Here, $h = (h_1, \dots, h_d)$ is a bandwidth vector with $h_1, \dots, h_d > 0$ and $\kappa : \mathbb{R} \to \mathbb{R}$ is some given univariate density function, i.e., $\kappa(t) \geq 0$ and $\int \kappa(t)\mathrm{d}t = 1$. We use the variable u twice in the notation because away from the boundary of $\mathcal{X}$, the kernel $K_h^u(u-x)$ only depends on $u-x$. It is worth emphasizing that the empirical minimization criterion S depends on a choice of a kernel κ and a smoothing bandwidth h. While the choice of κ is not of great importance, see e.g. Section 3.3.2. in [112], the quality of estimation heavily depends on an appropriate choice of the smoothing parameter h. We do not discuss the choice of a (data-driven) bandwidth, but note that the asymptotic properties of the local linear smoothing estimator do simplify the choice of bandwidths compared to other estimators. The reason is that the asymptotic bias of one additive component does not depend on the shape of the other components and on the bandwidths used for the other components.

We now argue that the local linear smooth backfitting estimator can be interpreted as an empirical projection of the data onto a space of additive functions. We introduce the linear space

$$\mathcal{H} = \left\{(f^{i,j})_{i=1,\ldots,n;\ j=0,\ldots,d} \middle|\ f^{i,j} : \mathcal{X} \mapsto \mathbb{R},\ \|f\|_n < \infty\right\}$$

with inner product

$$\begin{aligned}\langle f, g\rangle_n &= n^{-1}\sum_{i=1}^{n}\int_{\mathcal{X}}\left\{f^{i,0}(x) + \sum_{j=1}^{d} f^{i,j}(x)(X_{ij}-x_j)\right\} \\ &\qquad \times\left\{g^{i,0}(x) + \sum_{k=1}^{d} g^{i,k}(x)(X_{ij}-x_j)\right\} K_h^{X_i}(X_i - x)\mathrm{d}x\end{aligned}$$

and norm $\|f\|_n = \sqrt{\langle f, f\rangle_n}$. We identify the response $Y = (Y_i)_{i=1,\ldots,n}$ as an element of $\mathcal{H}$ via $Y^{i,0} \equiv Y_i$ and $Y^{i,j} \equiv 0$ for $j \geq 1$. We later assume that the functions m_j are differentiable. We identify the regression function

$$m : \mathcal{X} \to \mathbb{R},\ m(x) = m_0 + m_1(x_1) + \cdots + m_d(x_d)$$

as an element of $\mathcal{H}$ via $m^{i,0}(x) = m_0 + \sum_j m_j(x_j)$ and $m^{i,j} = \partial m_j(x_j)/\partial x_j$ for $j \geq 1$. Note that the components of $m \in \mathcal{H}$ do not depend on i. We define the following subspaces of $\mathcal{H}$:

$$\begin{aligned}\mathcal{H}_{full} &= \{f \in \mathcal{H}|\ \text{the components of } f \text{ do not depend on } i\}, \\ \mathcal{H}_{add} &= \Big\{f \in \mathcal{H}_{full}|\ f^{i,0}(x) = f_0 + f_1(x_1) + \cdots + f_d(x_d), f^{i,j}(x) = f_j^{(1)}(x_j) \\ &\qquad \text{for some } f_0 \in \mathbb{R} \text{ and some univariate functions } f_j, f_j^{(1)} : \mathcal{X}_j \to \mathbb{R}, \\ &\qquad j = 1, \ldots, d \text{ with } \sum_{i=1}^{n}\int_{\mathcal{X}} f_j(x_j)K_h^{X_i}(X_i - x)\mathrm{d}x = 0\Big\}.\end{aligned}$$

For a function $f \in \mathcal{H}_{add}$ we write $f_0 \in \mathbb{R}$ and $f_j, f_j^{(1)} : \mathcal{X}_j \to \mathbb{R}$ for $j = 1, \ldots, d$ for the constant and functions that define f. In the next section we state conditions under which the constant f_0 and functions $f_j, f_j^{(1)}$ are unique given any $f \in \mathcal{H}_{add}$. By a slight abuse of notation we also write f_j for the element of $\mathcal{H}$ given by $f^{i,0}(x) = f_j(x_j)$ and $f^{i,k}(x) \equiv 0$ for $k = 1, \ldots, d$. We also write $f_j^{(1)}$ for the element of $\mathcal{H}$ with $f^{i,k}(x) \equiv 0$ for $k \neq j$ and $f^{i,j}(x) = f_j^{(1)}(x_j)$. Furthermore, we define $f_{j+d} := f_j^{(1)}$ for $j = 1, \dots, d$ for both interpretations. Thus, for $f \in \mathcal{H}_{add}$ we have

$$f = f_0 + \cdots + f_{2d}. \tag{2.2.2}$$

Recall that the linear smooth backfitting estimator

$$\widehat{m} = (\widehat{m}_0, \widehat{m}_1, \dots, \widehat{m}_d, \widehat{m}_1^{(1)}, \dots, \widehat{m}_d^{(1)})$$

is defined as the minimizer of the criterion S under the constraint (2.2.1). By setting $\widehat{m}^{i,0}(x) = \widehat{m}_0 + \sum_{j=1}^{d} \widehat{m}_j(x_j)$ and $m^{i,j}(x) = \widehat{m}_j^{(1)}(x_j)$ for $j \geq 1$ it can easily be seen that

$$\widehat{m} = \operatorname*{arg\,min}_{f \in \mathcal{H}_{add}} \|Y - f\|_n. \tag{2.2.3}$$

In the next section we state conditions under which the minimization has a unique solution. Equation (2.2.3) provides a geometric interpretation of local linear smooth backfitting. The local linear smooth backfitting estimator is an orthogonal projection of the response vector Y onto the linear subspace $\mathcal{H}_{add} \subseteq \mathcal{H}$. We make repeated use of this fact in this Chapter.
We now introduce the following subspaces of $\mathcal{H}$:

$$\begin{aligned}
\mathcal{H}_0 &= \left\{ f \in \mathcal{H} | f^{i,0}(x) \equiv c \text{ for some } c \in \mathbb{R}, f^{i,j}(x) \equiv 0 \ \text{ for } \ j \neq 0 \right\}, \\
\mathcal{H}_k &= \Big\{ f \in \mathcal{H} | \ f^{i,j}(x) \equiv 0 \ \text{ for } \ j \neq 0, \text{ and } f^{i,0}(x) = f_k(x_k) \text{ for some} \\
&\qquad \text{function } f_k : \mathcal{X}_k \to \mathbb{R} \text{ with } \sum_{i=1}^{n} \int_{\mathcal{X}} f_k(x_k) K_h^{X_i}(X_i - x) \mathrm{d}x = 0 \Big\}, \\
\mathcal{H}_{k'} &= \Big\{ f \in \mathcal{H} | \ f^{i,j}(x) \equiv 0 \ \text{ for } \ j \neq k, f^{i,k}(x) = f_k^{(1)}(x_k) \text{ for some} \\
&\qquad \text{function } f_k^{(1)} : \mathcal{X}_k \to \mathbb{R} \Big\}
\end{aligned}$$

for $k = 1, \dots, d$ and $k' := k + d$. Using these definitions we have $\mathcal{H}_{add} = \sum_{j=0}^{2d} \mathcal{H}_j$ with $\mathcal{H}_j \cap \mathcal{H}_k = \{0\}, j \neq k$. In particular, the functions f_j in (2.2.2) are unique elements in $\mathcal{H}_j, j = 0, \dots, 2d$. For $k = 0, \dots, 2d$ we denote the orthogonal projection of $\mathcal{H}$ onto the space $\mathcal{H}_k$ by $\mathcal{P}_k$. Note that for $k = 0, \dots, d$ the operators $\mathcal{P}_k$ set all components of an element $f = (f^{i,j})_{i=1,\dots,n;\ j=0,\dots,d} \in \mathcal{H}$ to zero except the components with indices $(i, 0), i = 1, \dots, n$. Furthermore, for $k = d + 1, \dots, 2d$, only components with index $(i, k - d)$ are not set to zero. Because $\mathcal{H}_0$ is orthogonal to $\mathcal{H}_k$ for $k = 1, \dots, d$, the orthogonal projection onto the space $\mathcal{H}_k$ is given by $\mathcal{P}_k = P_k - \mathcal{P}_0$ where P_k is the projection onto $\mathcal{H}_0 + \mathcal{H}_k$. In Subsection 7.1.1 we state explicit formulas for the orthogonal projection operators.
The operators $\mathcal{P}_k$ can be used to define an iterative algorithm for the approximation of $\widehat{m}$. For an explanation observe that $\widehat{m}$ is the projection of Y onto $\mathcal{H}_{add}$ and $\mathcal{H}_k$ is a linear subspace of

$\mathcal{H}_{add}$. Thus $\mathcal{P}_k(Y) = \mathcal{P}_k(\widehat{m})$ holds for $k = 0, \ldots, 2d$. This gives

$$\mathcal{P}_k(Y) = \mathcal{P}_k(\widehat{m}) = \mathcal{P}_k\left(\sum_{j=0}^{2d} \widehat{m}_j\right) = \widehat{m}_k + \mathcal{P}_k\left(\sum_{j\neq k} \widehat{m}_j\right) \tag{2.2.4}$$

or, equivalently,

$$\widehat{m}_k = \mathcal{P}_k(Y) - \sum_{j\neq k} \mathcal{P}_k(\widehat{m}_j) = P_k(Y) - \bar{Y} - \sum_{j\neq k} \mathcal{P}_k(\widehat{m}_j),$$

where $\bar{Y} = \mathcal{P}_0(Y) = P_0(Y)$ is the element of $\mathcal{H}$ with $(\bar{Y})^{i,0} \equiv \frac{1}{n}\sum_{i=1}^n Y^i$, $(\bar{Y})^{i,j} \equiv 0$ for $j \geq 1$. This equation hyperrectanglenspires an iterative algorithm where in each step approximations $\widehat{m}_k^{old}$ of $\widehat{m}_k$ are updated by

$$\widehat{m}_k^{new} = P_k(Y) - \bar{Y} - \sum_{j\neq k} \mathcal{P}_k(\widehat{m}_j^{old}).$$

Algorithm 4 provides a compact definition of our algorithm for the approximation of $\widehat{m}$. In

Algorithm 4 Smooth Backfitting Algorithm

```
1: Start: $\widehat{m}_k(x_k) \equiv 0, \widetilde{m}_k = \mathcal{P}_k(Y), error = \infty$        ▷ $k = 0, \ldots, 2d$
2: while $error > tolerance$ do
3:     $error \leftarrow 0$
4:     for $k = 0, \ldots, 2d$ do
5:         $\widehat{m}_k^{old} \leftarrow \widehat{m}_k$
6:         $\widehat{m}_k \leftarrow \widetilde{m}_k - \sum_{j\neq k} \mathcal{P}_k(\widehat{m}_j)$
7:         $error \leftarrow error + |\widehat{m}_k - \widehat{m}_k^{old}|$
8: return $\widehat{m} = (\widehat{m}_0, \widehat{m}_1, \ldots, \widehat{m}_{2d})$
```

each iteration step, either $\widehat{m}_j$ or $\widehat{m}_j^{(1)}$ is updated for some $j = 1, \ldots, d$. This is different from the algorithm proposed in [87] where in each step a function tuple $(\widehat{m}_j, \widehat{m}_j^{(1)})$ is updated. For the orthogonal projections of functions $m \in \mathcal{H}_{add}$ one can use simplified formulas. They are given in Subsection 7.1.1. Note that $\tilde{m}_k, k = 0, \ldots, 2d$ only needs to be calculated once at the beginning. Also the marginals $p_k(x_k)$, $p_k^*(x_k)$, $p_k^{**}(x_k)$, $p_{jk}(x_j, x_k)$, $p_{jk}^*(x_j, x_k)$ and $p_{jk}^{**}(x_j, x_k)$ which are needed in the evaluation of $\mathcal{P}_k$ only need to be calculated once at the beginning. Precise definitions of these marginals can be found in the following sections. In each iteration of the for-loop in line 4 of Algorithm 4, $O(d \times n \times gs)$ calculations are performed. Hence for a full cycle, the algorithm needs $O(d^2 \times n \times gs \times \log(1/\text{tolerance}))$ calculations. Here grid.size is the number of evaluation points for each coordinate x_k.

Existence and uniqueness of the local linear smooth backfitting estimator are discussed in the next section. Additionally, convergence of the proposed iterative algorithm is shown.

2.3 Existence and Uniqueness of the Estimator, Convergence of the Algorithm

In this section we establish conditions for existence and uniqueness of the local linear smooth backfitting estimator $\widehat{m}$. Afterwards we discuss convergence of the iterative algorithm provided in Algorithm 4. Note that convergence is shown for arbitrary starting values, i.e., we can set $\widehat{m}_k(x_k)$ to values other than zero in step 1 of Algorithm 4. For these statements we require the following weak condition on the kernel.

(A1) The kernel k has support $[-1,1]$. Furthermore, k is strictly positive on $(-1,1)$ and continuous on $\mathbb{R}$.

For $k=1,\dots,d$ and $x\in\mathbb{R}^d$ we write

$$x_{-k} := (x_1,\dots,x_{k-1},x_{k+1},\dots,x_d).$$

In the following, we show that our claims hold on the following event:

$$\begin{aligned}\mathcal{E} = \Bigg\{ &\text{For } k=1,\dots,d \text{ and } x_k\in\overline{\mathcal{X}}_k \text{ there exist } i_1,i_2\in\{1,\dots,n\}\\ &\text{such that } X_{i_1,k}\neq X_{i_2,k}\ ,\ |X_{i,k}-x_k|<h_k, (x_k,X_{i,-k})\in\overline{\mathcal{X}} \text{ for } i=i_1,i_2.\\ &\text{There exist no } b_0,\dots,b_d\in\mathbb{R} \text{ with } b_0+\sum_{j=1}^d b_jX_{ij}=0\ \forall i=1,\dots,n\Bigg\},\end{aligned}$$

where $\overline{\mathcal{X}}_k$ is the closure of $\mathcal{X}_k$ and by a slight abuse of notation

$$(x_k,X_{i,-k}) := (X_{i,1},\dots,,X_{i,k-1},x_k,X_{i,k+1},\dots,,X_{i,d}).$$

We require the following definitions.

$$\begin{aligned}\widehat{p}_k(x_k) &= \frac{1}{n}\sum_{i=1}^n\int_{\mathcal{X}_{-k}(x_k)} K_h^{X_i}(X_i-x)\mathrm{d}x_{-k},\\ \widehat{p}_k^*(x_k) &= \frac{1}{n}\sum_{i=1}^n\int_{\mathcal{X}_{-k}(x_k)} (X_{ik}-x_k)K_h^{X_i}(X_i-x)\mathrm{d}x_{-k},\\ \widehat{p}_k^{**}(x_k) &= \frac{1}{n}\sum_{i=1}^n\int_{\mathcal{X}_{-k}(x_k)} (X_{ik}-x_k)^2K_h^{X_i}(X_i-x)\mathrm{d}x_{-k},\end{aligned}$$

where $\mathcal{X}_{-k}(x_k)=\{u_{-k}\mid (x_k,u_{-k})\in\mathcal{X}\}$.

Lemma 2.3.1. *Make Assumption (A1). Then, on the event $\mathcal{E}$ it holds that $\|f\|_n=0$ implies $f_0=0$ as well as $f_j\equiv 0$ almost everywhere for $j=1\dots 2d$ and all $f\in\mathcal{H}_{add}$.*

One can easily see that the lemma implies the following. On the event $\mathcal{E}$, if a minimizer $\widehat{m}=\widehat{m}_0+\dots+\widehat{m}_{2d}$ of $\|Y-f\|_n$ over $f\in\mathcal{H}_{add}$ exists, the components $\widehat{m}_0,\dots,\widehat{m}_{2d}$ are

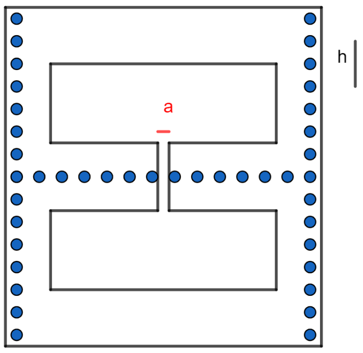


Figure 2.1: An example of a possible data set $\mathcal{X} \subseteq \mathbb{R}^2$ including data points where the conditions of the event $\mathcal{E}$ are not satisfied and where the components of functions $f \in \mathcal{H}_{add}$ are not identified. The data is visualized by blue dots. The size of the parameter $h = h_1 = h_2$ is showcased on the right hand side. For explanatory reasons, the interval a is included.

uniquely determined: Suppose there exists another minimizer $\tilde{m} \in \mathcal{H}_{add}$. Then it holds that $\langle Y - \widehat{m}, \widehat{m} - \tilde{m}\rangle_n = 0$ and $\langle Y - \tilde{m}, \widehat{m} - \tilde{m}\rangle_n = 0$ which gives $\|\widehat{m} - \tilde{m}\|_n = 0$. An application of the lemma yields uniqueness of the components $\widehat{m}_0, \dots, \widehat{m}_{2d}$.

Remark 2.3.2. *In Figure 2.1 we give an example where a set $\mathcal{X}$ and data points X_i do not belong to the event $\mathcal{E}$ and where the components of the function $f \in \mathcal{H}_{add}$ are not identified. Note that in this example for all $k = 1, 2$ and $x_k \in \overline{\mathcal{X}_k}$ there exist $i_1, i_2 \in \{1, \dots, n\}$ such that $X_{i_1} \neq X_{i_2}$ and $|X_{i,k} - x_k| < h$, for $i = i_1, i_2$. However, for $x_1 \in a$ the condition $(x_1, X_{i,2}) \in \overline{\mathcal{X}}$ is not fulfilled for any $i = 1, \dots, n$ with $|X_{i,1} - x_1| < h$. Therefore, $K_h^{X_i}(X_i - x) = 0$ for all $x \in \overline{\mathcal{X}}$ with $x_1 \in a$. Thus, any function satisfying $f \in \mathcal{H}_1$ with $f_1(x) = 0$ for $x \in \mathcal{X}_1 \backslash \{a\}$ has the property $\|f\|_n = 0$.*

Proof of Lemma 2.3.1. First, for each pair $i_1, i_2 = 1, \dots, n$ define the set

$$M_{i_1,i_2} := \{x_1 \in \mathcal{X}_1 \mid |X_{i,1} - x_1| < h, (x_1, X_{i,-1}) \in \mathcal{X} \text{ for } i = i_1, i_2\}$$

if $X_{i_1,1} \neq X_{i_2,1}$ and $M_{i_1,i_2} = \emptyset$ otherwise. It is easy to see that M_{i_1,i_2} is open as an intersection of open sets. Note that on the event $\mathcal{E}$ we have

$$\bigcup_{i_1,i_2} M_{i_1,i_2} = \mathcal{X}_1. \tag{2.3.1}$$

Now, suppose that for some $f \in \mathcal{H}_{add}$ we have $\|f\|_n = 0$. We want to show that $f_0 = 0$ and that $f_j \equiv 0$ for $j = 1, \dots, 2d$. From $\|f\|_n = 0$ we obtain

$$\left\{f_0 + \sum_{j=1}^{d} f_j(x_j) + \sum_{j=1}^{d} f_{j'}(x_j)(X_{ij} - x_j)\right\}^2 K_h^{X_i}(X_i - x) = 0$$

for $i = 1, \dots, n$ and almost all $x \in \mathcal{X}$. Let $i_1, i_2 \in \{1, \dots, n\}$. Then

$$f_0 + f_1(x_1) + \sum_{j=2}^{d} f_j(X_{ij}) + f_{d+1}(x_1)(X_{i1} - x_1) = 0 \tag{2.3.2}$$

holds for all $i = i_1, i_2$ and $x_1 \in M_{i_1,i_2}$ almost surely. By subtraction of Equation (2.3.2) for $i = i_1$ and $i = i_2$ we receive

$$f_{d+1}(x_1) = v_1$$

with constant $v_1 = -\sum_{j=2}^{d}(f_j(X_{i_1,j}) - f_j(X_{i_2,j}))/(X_{i_1,1} - X_{i_2,1})$ for $x_1 \in M_{i_1,i_2}$. Furthermore, by using (2.3.2) once again we obtain

$$f_1(x_1) = u_1 + v_1 x_1$$

with another constant $u_1 \in \mathbb{R}$. Following (2.3.1), since $\mathcal{X}_1$ is connected and the sets M_{i_1,i_2} are open we can conclude

$$f_{d+1}(x_1) = v_1 \text{ and } f_1(x_1) = u_1 + v_1 x_1$$

for almost all $x_1 \in \mathcal{X}_1$ since the sets must overlap. Similarly one shows

$$f_{j'}(x_j) = v_j \text{ and } f_j(x_j) = u_j + v_j x_j$$

for $j = 2, \dots, d$ and almost all $x_j \in \mathcal{X}_j$. We conclude that

$$\begin{aligned} 0 &= \|f\|_n^2 \\ &= \frac{1}{n} \int \sum_{i=1}^{n} \left\{ f_0 + \sum_{j=1}^{d} f_j(x_j) + \sum_{j=1}^{d} f_{j'}(x_j)(X_{ij} - x_j) \right\}^2 K_h^{X_i}(X_i - x)\mathrm{d}x \\ &= \frac{1}{n} \int \sum_{i=1}^{n} \left\{ f_0 + \sum_{j=1}^{d} u_j + \sum_{j=1}^{d} v_j X_{ij} \right\}^2 K_h^{X_i}(X_i - x)\mathrm{d}x \\ &= \left\{ f_0 + \sum_{j=1}^{d} u_j + \sum_{j=1}^{d} v_j X_{ij} \right\}^2. \end{aligned}$$

On the event $\mathcal{E}$ the covariates X_i do not lie in a linear subspace of $\mathbb{R}^d$. This shows $v_j = 0$ for $1 \le j \le d$. Thus $f_j \equiv 0$ for $d + 1 \le j \le 2d$ and $f_j = u_j$ for $1 \le j \le d$.

Now, $\int f_j(x_j)\widehat{p}_j(x_j)\mathrm{d}x_j = 0$ implies that $f_j \equiv 0$ for $1 \le j \le d$ and $f_0 = 0$. This concludes the proof of the lemma. □

Existence and uniqueness of $\widehat{m}$ on the event $\mathcal{E}$ under Assumption (A1) follows immediately from the following lemma.

Lemma 2.3.3. *Make Assumption (A1). Then, on the event $\mathcal{E}$, for every $D \subseteq \{0, \dots, 2d\}$ the linear space $\sum_{k \in D} \mathcal{H}_k$ is a closed subset of $\mathcal{H}$. In particular, $\mathcal{H}_{add}$ is closed.*

For the proof of this lemma we make use of some propositions introduced below. In the following, we consider sums $L = L_1 + L_2$ of closed subspaces L_1 and L_2 of a Hilbert space with $L_1 \cap L_2 = \{0\}$. In this setup, an element $g \in L$ has a unique decomposition $g = g_1 + g_2$ with $g_1 \in L_1$ and $g_2 \in L_2$. Thus, the projection operator from L onto L_1 along L_2 given by

$$\Pi_1(L_2) : L \to L_1,\ \Pi_1(L_2)(g) = g_1$$

is well defined.

Proposition 2.3.4. *For the sum $L = L_1 + L_2$ of two closed subspaces L_1 and L_2 of a Hilbert space with $L_1 \cap L_2 = \{0\}$, the following conditions are equivalent*

(i) *L is closed.*

(ii) *There exists a constant $c > 0$ such that for every $g = g_1 + g_2 \in L$ with $g_1 \in L_1$ and $g_2 \in L_2$ we have*

$$\|g\| \geq c \max\{\|g_1\|, \|g_2\|\}. \tag{2.3.3}$$

(iii) *The projection operator $\Pi_1(L_2)$ from L onto L_1 along L_2 is bounded.*

(iv) *The gap from L_1 to L_2 is greater than zero, i.e.,*

$$\gamma(L_1, L_2) := \inf_{g_1 \in L_1} \frac{dist(g_1, L_2)}{\|g_1\|} > 0,$$

where $\operatorname{dist}(f, V) := \inf_{h \in V} \|f - h\|$ *with the convention* $0/0 = 1$.

Remark 2.3.5. *A version of Proposition 2.3.4 is also true if $L_1 \cap L_2 \neq \{0\}$. In this case, the quantities involved need to be identified as objects in the quotient space $L/(L_1 \cap L_2)$.*

Proposition 2.3.6. *The sum $L = L_1 + L_2$ of two closed subspaces L_1 and L_2 of a Hilbert space with $L_1 \cap L_2 = \{0\}$ is closed if the orthogonal projection of L_2 on L_1 is compact.*

The proofs of Propositions 2.3.4 and 2.3.6 can be reconstructed from A.4 Proposition 2 in [9], Chapter 4 Theorem 4.2 in [69] and [72]. For completeness, we have added proofs of the propositions in Subsection 7.1.2. We now come to the proof of Lemma 2.3.3.

Proof of Lemma 2.3.3. First note that the spaces $\mathcal{H}_k$ are closed for $k = 0, \dots, 2d$. We show that $\mathcal{H}_k + \mathcal{H}_{k'}$ is closed for $1 \leq k \leq d$. Consider $R = \min M$ where

$$M := \{r \geq 0 \mid (\widehat{p}_k^*(x_k))^2 \leq r \widehat{p}_k(x_k) \widehat{p}_k^{**}(x_k) \text{ for all } x_k \in \overline{\mathcal{X}}_k \text{ and } 1 \leq k \leq d\}$$

and

$$I_{x_k} := \left\{ i \in \{1, \dots, n\} \;\middle|\; \int_{u \in \mathcal{X}_{-k}(x_k)} K_h^{X_i}(X_i - u) \mathrm{d}u_{-k} > 0 \right\}$$

for $x_k \in \overline{\mathcal{X}}_k$. By the Cauchy-Schwarz inequality we have

$$(\widehat{p}_k^*(x_k))^2 \leq \widehat{p}_k(x_k) \widehat{p}_k^{**}(x_k)$$

for $x_k \in \overline{\mathcal{X}}_k$ and $1 \leq k \leq d$. This implies $R \leq 1$. Now, equality in the inequality only holds if $X_{ik} - x_k$ does not depend on $i \in I_{x_k}$. On the event $\mathcal{E}$ for $x_k \in \overline{\mathcal{X}}_k$ there exist $1 \leq i_1, i_2 \leq n$ with $|x_k - X_{i,k}| < h$ for $i = i_1, i_2$ and $X_{i_1,k} \neq X_{i_2,k}$. Thus, $X_{ik} - x_k$ depends on i for $i \in I_{x_k}$ and the strict inequality holds for all x_k. Furthermore, because the kernel function k is continuous, we have that $\widehat{p}_k$, $\widehat{p}_k^*$ and $\widehat{p}_k^{**}$ are continuous. Thogether with the compactness of $\overline{\mathcal{X}}_k$ this implies that $R < 1$ on the event $\mathcal{E}$. Now let $f \in \mathcal{H}_k$ and $g \in \mathcal{H}_{k'}$ for some $1 \leq k \leq d$. We show

$$\|f + g\|_n^2 \geq (1 - R)(\|f\|_n^2 + \|g\|_n^2). \tag{2.3.4}$$

By application of Proposition 2.3.4 this immediately implies that $\mathcal{H}_k + \mathcal{H}_{k'}$ is closed. For a proof of (2.3.4) note that

$$\begin{aligned}
&\|f + g\|_n^2 \\
= \;& n^{-1} \sum_{i=1}^{n} (f_k(x_k) + g_0 + (X_{ik} - x_k) g_{k'}(x_k))^2 K_h^{X_i}(X_i - x) \mathrm{d}x \\
= \;& \int (f_k(x_k) + g_0)^2 \widehat{p}_k(x_k) \mathrm{d}x_k + 2 \int (f_k(x_k) + g_0) g_{k'}(x_k) \widehat{p}_k^*(x_k) \mathrm{d}x_k \\
& \quad + \int g_{k'}(x_k)^2 \widehat{p}_k^{**}(x_k) \mathrm{d}x_k \\
\geq \;& \int (f_k(x_k) + g_0)^2 \widehat{p}_k(x_k) \mathrm{d}x_k + \int g_{k'}(x_k)^2 \widehat{p}_k^{**}(x_k) \mathrm{d}x_k \\
& \quad -2R \int |f_k(x_k) + g_0| |g_{k'}(x_k)| (\widehat{p}_k(x_k) \widehat{p}_k^{**}(x_k))^{1/2} \mathrm{d}x_k
\end{aligned}$$

and thus

$$\begin{aligned}
&\|f + g\|_n^2 \\
\geq \;& \int (f_k(x_k) + g_0)^2 \widehat{p}_k(x_k) \mathrm{d}x_k + \int g_{k'}(x_k)^2 \widehat{p}_k^{**}(x_k) \mathrm{d}x_k \\
& -2R \left(\int (f_k(x_k) + g_0)^2 \widehat{p}_k(x_k) \mathrm{d}x_k \right)^{1/2} \left(\int g_{k'}(x_k)^2 \widehat{p}_k^{**}(x_k) \mathrm{d}x_k \right)^{1/2} \\
\geq \;& (1 - R) \int (f_k(x_k) + g_0)^2 \widehat{p}_k(x_k) \mathrm{d}x_k + (1 - R) \int g_{k'}(x_k)^2 \widehat{p}_k^{**}(x_k) \mathrm{d}x_k \\
= \;& (1 - R) \left(\int f_k(x_k)^2 \widehat{p}_k(x_k) \mathrm{d}x_k + g_0^2 + \int g_{k'}(x_k)^2 \widehat{p}_k^{**}(x_k) \mathrm{d}x_k \right) \\
\geq \;& (1 - R)(\|f\|_n^2 + \|g\|_n^2),
\end{aligned}$$

where in the second to last row, we used that $f \in \mathcal{H}_k$. This concludes the proof of (2.3.4). Note that the statement of the lemma is equivalent to the following statement: For $D_1, D_2 \subseteq \{1, \ldots, d\}$ and $\delta \in \{0, 1\}$ the space $\delta\mathcal{H}_0 + \sum_{k \in D_1} \mathcal{H}_k + \sum_{k \in D_2} \mathcal{H}_{k'}$ is closed. We show this inductively over the number of elements $s = |D_2 \cap D_1|$ of $D_1 \cap D_2$.

For the case $s = 0$, note that for $D_1 \cap D_2 = \emptyset$, the space $\delta\mathcal{H}_0 + \sum_{k \in D_1} \mathcal{H}_k + \sum_{k \in D_2} \mathcal{H}_{k'}$ is closed, which can be shown with similar but simpler arguments than the ones used below.

Now let $s \geq 1$, $\delta \in \{0,1\}$, $D_1, D_2 \subseteq \{1,\dots,d\}$ with $|D_2 \cap D_1| = s-1$ and assume $L_2 = \delta\mathcal{H}_0 + \sum_{j\in D_1}\mathcal{H}_j + \sum_{j\in D_2}\mathcal{H}_{j'}$ is closed. Without loss of generality, let $k \in \{1,\dots,d\}\backslash(D_1\cup D_2)$. We argue that on the event $\mathcal{E}$ the orthogonal projection of L_2 on $L_1 = \mathcal{H}_k + \mathcal{H}_{k'}$ is Hilbert-Schmidt, noting that a Hilbert-Schmidt operator is compact. Using Proposition 2.3.6 since L_1 and L_2 are closed, this implies that $L = \delta\mathcal{H}_0 + \sum_{j\in D_1\cup\{k\}}\mathcal{H}_j + \sum_{j\in D_2\cup\{k\}}\mathcal{H}_{j'}$ is closed which completes the inductive argument.
For an element $f \in L_2$ with decomposition $f = f_0 + \sum_{j\in D_1} f_j + \sum_{j\in D_2} f_{j'}$ the projection onto $L_1 + \mathcal{H}_0$ is given by univariate functions $g_k, g_{k'}$ and $g_0 \in \mathbb{R}$ which satisfy

$$\begin{aligned} 0 \;=\; & \sum_{i=1}^{n}\int\Big(f_0 + \sum_{j\in D_1} f_j(x_j) + \sum_{j\in D_2} f_{j'}(x_j)(X_{ij}-x_j) - g_0 - g_k(x_k) \\ & -g_{k'}(x_k)(X_{ik}-x_k)\Big)\begin{pmatrix} 1 \\ X_{ik}-x_k \end{pmatrix} K^{X_i}(X_i - x)\mathrm{d}x_{-k}. \end{aligned}$$

Note that $f_0 = 0$ if $\delta = 0$. This implies

$$\begin{aligned} \begin{pmatrix} g_0 + g_k(x_k) \\ g_{k'}(x_k) \end{pmatrix} &= \frac{1}{(\widehat{p}_k\widehat{p}_k^{**} - (\widehat{p}_k^{*})^2)(x_k)}\begin{pmatrix} \widehat{p}_k^{**} & -\widehat{p}_k^{*} \\ -\widehat{p}_k^{*} & \widehat{p}_k \end{pmatrix}(x_k) \\ &\times\Bigg\{ f_0\begin{pmatrix} \widehat{p}_k \\ \widehat{p}_k^{*} \end{pmatrix}(x_k) + \sum_{j\in D_1}\int f_j(x_j)\begin{pmatrix} \widehat{p}_{jk}(x_j,x_k) \\ \widehat{p}_{kj}^{*}(x_k,x_j) \end{pmatrix}\mathrm{d}x_j \\ &+ \sum_{j\in D_2}\int f_{j'}(x_j)\begin{pmatrix} \widehat{p}_{jk}^{*}(x_j,x_k) \\ \widehat{p}_{jk}^{**}(x_j,x_k) \end{pmatrix}\mathrm{d}x_j\Bigg\}, \end{aligned}$$

where g_k and g_0 are chosen such that $\int g_k(x_k)\widehat{p}_k(x_k)\mathrm{d}x_k = 0$. We now use that the projection of g_0 onto $L_1 = \mathcal{H}_k + \mathcal{H}_{k'}$ is equal to

$$\begin{pmatrix} r_k(x_k) \\ r_{k'}(x_k) \end{pmatrix} = \begin{pmatrix} g_0(1 - c_k s_k(x_k)\widehat{p}_k^{**}(x_k)) \\ g_0 c_k s_k(x_k)\widehat{p}_k^{*}(x_k) \end{pmatrix},$$

where $s_k(x_k) = \widehat{p}_k(x_k)/(\widehat{p}_k(x_k)\widehat{p}_k^{**}(x_k) - (\widehat{p}_k^{*})^2(x_k))$ and $c_k = (\int s_k(x_k)\widehat{p}_k^{**}(x_k)\ \widehat{p}_k(x_k)\mathrm{d}x_k)^{-1}$. Thus, the projection of f onto L_1 is defined by $(g_k(x_k) + r_k(x_k),\ g_{k'}(x_k) + r_{k'}(x_k))^\intercal$. Under our settings on the event $\mathcal{E}$ this is a Hilbert-Schmidt operator. This concludes the proof. □

We now come to a short discussion of the convergence of Algorithm 4. The algorithm is used to approximate $\widehat{m}$. In the lemma we denote by $\widehat{m}^{[r]}$ the outcome of the algorithm after r iterations of the while loop (see Algorithm 4).

We prove the algorithm for arbitrary starting values, i.e. we can set the $\widehat{m}_k(x_k)$ to values other than zero in step 1 of Algorithm 4. The vector of starting values of the algorithm is denoted by $\widehat{m}^{[0]} \in \mathcal{H}_{add}$.

Lemma 2.3.7. *Make Assumption (A1). Then, on the event $\mathcal{E}$, for Algorithm 4 and all choices* *of starting values $\widehat{m}^{[0]} \in \mathcal{H}_{add}$ we have*

$$\|\widehat{m}^{[r]} - \widehat{m}\|_n \leq V^r \|\widehat{m}^{[0]} - \widehat{m}\|_n,$$

where $0 \leq V = 1 - \prod_{k=0}^{2d-1} \gamma^2(\mathcal{H}_k, \mathcal{H}_{k+1} + \cdots + \mathcal{H}_{2d}) < 1$ is a random variable depending on the observations.

Remark 2.3.8. *On the event $\mathcal{E}$, the algorithm converges with a geometric rate where in every iteration step the distance to the limiting value, $\widehat{m}$, is reduced by a factor smaller or equal to V. If the columns of the design matrix X are orthogonal, V is close to zero and if they are highly correlated, V is close to 1. The variable V depends on n and is random. Under additional assumptions, as stated in the next section, one can show that with probability tending to one, V is bounded by a constant smaller than 1.*

Proof of Lemma 2.3.7. For a subspace $\mathcal{V} \subseteq \mathcal{H}_{add}$ we denote by $\mathcal{P}_{\mathcal{V}}$ the orthogonal projection onto $\mathcal{V}$. For $k = 0, \dots, 2d$ let $\mathcal{Q}_k := \mathcal{P}_{\mathcal{H}_k^\perp} = 1 - \mathcal{P}_k$ be the projection onto the orthogonal complement $\mathcal{H}_k^\perp$ of $\mathcal{H}_k$. The idea is to show the following statements.

(i) $Y - \widehat{m}^{[r]} = (\mathcal{Q}_{2d} \dots \mathcal{Q}_0)^r (Y - \widehat{m}^{[0]})$,

(ii) $(\mathcal{Q}_{2d} \dots \mathcal{Q}_0)^r (Y - \widehat{m}) = Y - \widehat{m}$,

This then implies

$$\|\widehat{m}^{[r]} - \widehat{m}\|_n = \|\widehat{m}^{[r]} - Y + Y - \widehat{m}\|_n = \|(\mathcal{Q}_{2d} \dots \mathcal{Q}_0)^r (\widehat{m}^{[0]} - \widehat{m})\|_n.$$

The proof is concluded by showing

$$\|\mathcal{Q}_{2d} \dots \mathcal{Q}_0 g\|_n^2 \leq \left(1 - \prod_{k=0}^{2d-1} \gamma^2(\mathcal{H}_k, \mathcal{H}_{k+1} + \cdots + \mathcal{H}_{2d})\right) \|g\|_n^2 \tag{2.3.5}$$

for all $g \in \mathcal{H}_{add}$. Note that $0 \leq V := 1 - \prod_{k=0}^{2d-1} \gamma^2(\mathcal{H}_k, \mathcal{H}_{k+1} + \cdots + \mathcal{H}_{2d}) < 1$ by Lemma 2.3.3 and Proposition 2.3.4.

For (i), observe that for all $r \geq 1$ and $k = 0, \dots, 2d$ we have

$$\begin{aligned}
&Y - \widehat{m}_0^{[r-1]} - \cdots - \widehat{m}_{k-1}^{[r-1]} - \widehat{m}_k^{[r]} - \cdots - \widehat{m}_{2d}^{[r]} \\
&= (1 - \mathcal{P}_k)(Y - \widehat{m}_0^{[r-1]} - \cdots - \widehat{m}_{k-1}^{[r-1]} - \widehat{m}_{k+1}^{[r]} - \cdots - \widehat{m}_{2d}^{[r]}) \\
&= (1 - \mathcal{P}_k)(Y - \widehat{m}_0^{[r-1]} - \cdots - \widehat{m}_{k-1}^{[r-1]} - \widehat{m}_k^{[r-1]} - \widehat{m}_{k+1}^{[r]} - \cdots - \widehat{m}_{2d}^{[r]}) \\
&= \mathcal{Q}_k(Y - \widehat{m}_0^{[r-1]} - \cdots - \widehat{m}_k^{[r-1]} - \widehat{m}_{k+1}^{[r]} - \cdots - \widehat{m}_{2d}^{[r]}).
\end{aligned}$$

The statement follows inductively by beginning with the case $r = 1, k = 0$. Secondly, (ii) follows from

$$\mathcal{Q}_r \dots \mathcal{Q}_0 (Y - \widehat{m}) = \mathcal{Q}_r \dots \mathcal{Q}_0 \mathcal{P}_{\mathcal{H}_0^\perp \cap \dots \cap \mathcal{H}_{2d}^\perp}(Y) = \mathcal{P}_{\mathcal{H}_0^\perp \cap \dots \cap \mathcal{H}_{2d}^\perp}(Y) = Y - \widehat{m}.$$

It remains to show the inequality in (2.3.5).

For $0 \leq k \leq 2d$ define $\mathcal{N}_k := \mathcal{H}_k + \cdots + \mathcal{H}_{2d}$. We prove $\|\mathcal{Q}_{2d} \dots \mathcal{Q}_j g\|_n^2 \leq (1 - \prod_{k=j}^{2d-1} \gamma^2(\mathcal{H}_k, \mathcal{H}_{k+1} + \cdots + \mathcal{H}_{2d}))\|g\|_n^2$ for all $g \in \mathcal{H}_{add}$ and $0 \leq j \leq 2d$ using an inductive argument.
The case $j = 2d$ is trivial. For $0 \leq j < 2d$ and any $g \in \mathcal{H}_{add}$ let $g_j^{\perp} := \mathcal{Q}_j g = g' + g''$ with $g' := \mathcal{P}_{\mathcal{N}_{j+1}^{\perp}}(g)$ and $g'' := \mathcal{P}_{\mathcal{N}_{j+1}}(g)$. Then, by orthogonality, we have

$$\|\mathcal{Q}_{2d} \dots \mathcal{Q}_{j+1} g_j^{\perp}\|_n^2 = \|g' + \mathcal{Q}_{2d} \dots \mathcal{Q}_{j+1} g''\|_n^2 = \|g'\|_n^2 + \|\mathcal{Q}_{2d} \dots \mathcal{Q}_{j+1} g''\|_n^2.$$

Induction gives

$$\|\mathcal{Q}_{2d} \dots \mathcal{Q}_{j+1} g''\|_n^2 \leq \left(1 - \prod_{k=j+1}^{2d-1} \gamma^2(\mathcal{H}_k, \mathcal{H}_{k+1} + \cdots + \mathcal{H}_{2d})\right)(\|g_j^{\perp}\|_n^2 - \|g'\|_n^2)$$

which implies

$$\begin{aligned}\|\mathcal{Q}_{2d} \dots \mathcal{Q}_{j+1} g_j^{\perp}\|_n^2 \leq & \left(1 - \prod_{k=j+1}^{2d-1} \gamma^2(\mathcal{H}_k, \mathcal{H}_{k+1} + \cdots + \mathcal{H}_{2d})\right)\|g_j^{\perp}\|_n^2 \\ & + \prod_{k=j+1}^{2d-1} \gamma^2(\mathcal{H}_k, \mathcal{H}_{k+1} + \cdots + \mathcal{H}_{2d})\|g'\|_n^2.\end{aligned}$$

By Lemma 2.3.3 and Lemma 7.1.1 we have

$$\|g'\|_n^2 \leq \|\mathcal{P}_{\mathcal{N}_{j+1}^{\perp}} \mathcal{Q}_1\|_n^2 = \|\mathcal{P}_{N_{j+1}} \mathcal{P}_{\mathcal{H}_j}\|_n^2 = 1 - \gamma^2(\mathcal{H}_j, \mathcal{H}_{j+1} + \cdots + \mathcal{H}_{2d}).$$

This concludes the proof by noting that $\|g_j^{\perp}\|_n \leq \|g\|_n$. □

2.4 Asymptotic Properties of the Estimator

In this section we discuss asymptotic properties of the local linear smooth backfitting estimator. For simplicity we consider only the case that $\mathcal{X}$ is a product of intervals $\mathcal{X}_j = (a_j, b_j) \subset \mathbb{R}$. We make the following additional assumptions:

(A2) The observations (Y_i, X_i) are i.i.d. and the covariates X_i have one-dimansional marginal densities p_j which are strictly positive on $[a_k, b_k]$. The two-dimensional marginal densities p_{jk} of $(X_{i,j}, X_{i,k})$ are continuous on their support $[a_j, b_j] \times [a_k, b_k]$.

(A3) It holds

$$Y_i = m_0 + m_1(X_{i1}) + \cdots + m_d(X_{id}) + \varepsilon_i, \tag{2.4.1}$$

for twice continuously differentiable functions $m_j : \mathcal{X}_j \to \mathbb{R}$ with $\int m_j(x_j)\, p_j(x_j) \mathrm{d}x_j = 0$. The error variables ε_i satisfy $\mathbb{E}[\varepsilon_i | X_i] = 0$ and

$$\sup_{x \in \mathcal{X}} \mathbb{E}[|\varepsilon_i|^{5/2} | X_i = x] < \infty.$$

(A4) There exist constants $c_1, \ldots, c_d > 0$ with $n^{1/5} h_j \to c_j$ for $n \to \infty$. To simplify notation we assume that $h_1 = \cdots = h_d$. In abuse of notation we write h for h_j and c_h for c_j.

From now on we write $\widehat{m}^n = (\widehat{m}_0^n, \widehat{m}_1^n, \ldots, \widehat{m}_{2d}^n)$ for the estimator $\widehat{m}$ to indicate its dependence on the sample size n. The following theorem states an asymptotic expansion for the components $\widehat{m}_1^n, \ldots, \widehat{m}_d^n$. Later in this section we state some lemmas which are used to prove the result.

Theorem 2.4.1. *Make assumptions (A1) ,. . . , (A4). Then*

$$\begin{aligned}&\left|\widehat{m}_j^n(x_j) - m_j(x_j) - \left(\beta_j(x_j) - \int \beta_j(u_j) p_j(u_j) \mathrm{d}u_j\right) - v_j(x_j)\right| \\ &= o_P(h^2 + \{nh\}^{-1/2}) = o_P(n^{-2/5}),\end{aligned}$$

holds uniformly over $1 \leq j \leq d$ and $a_j \leq x_j \leq b_j$, where v_j is a stochastic variance term

$$v_j(x_j) = \frac{\frac{1}{n}\sum_{i=1}^n h^{-1} k(h^{-1}(X_{ij} - x_j))\varepsilon_i}{\frac{1}{n}\sum_{i=1}^n h^{-1} k(h^{-1}(X_{ij} - x_j))} = O_p(\{nh\}^{-1/2})$$

and β_j is a deterministic bias term

$$\beta_j(x_j) = \frac{1}{2} h^2 m_j^{''}(x_j) \frac{b_{j,2}(x_j)^2 - b_{j,1}(x_j) b_{j,3}(x_j)}{b_{j,0}(x_j) b_{j,2}(x_j) - b_{j,1}(x_j)^2} = O(h^2),$$

with $b_{j,l}(x_j) = \int_{\mathcal{X}_j} k(h^{-1}(u_j - x_j))(u_j - x_j)^l h^{-l-1} b_j(u_j)^{-1} \mathrm{d}u_j$ and $b_j(x_j) = \int_{\mathcal{X}_j} k(h^{-1}(x_j - w_j)) h^{-1} \mathrm{d}w_j$ for $0 \leq l \leq 2$.

The expansion for $\widehat{m}_j^n$ stated in the theorem neither depends on d nor on functions m_k $(k \neq j)$. In particular, this shows that the same expansion holds for the local linear estimator $\widetilde{m}_j^n$ in the oracle model where the functions m_k $(k \neq j)$ are known. More precisely, in the oracle model one observes i.i.d. observations (Y_i^*, X_{ij}) with

$$Y_i^* = m_j(X_{ij}) + \varepsilon_i, \quad Y_i^* = Y_i - \sum_{k \neq j} m_k(X_{ik}), \tag{2.4.2}$$

and the local linear estimator $\widetilde{m}_j^n$ is defined as the second component that minimises the criterion

$$\begin{aligned}\widetilde{S}(f_0, f_j, f_j^{(1)}) &= \sum_{i=1}^n \int_{\mathcal{X}} \left\{Y_i^* - f_0 - f_j(x_j) - f_j^{(1)}(x_j)(X_{ij} - x_j)\right\}^2 \\ &\quad \times \kappa_h^{X_{ij}}(X_{ij} - x) \mathrm{d}x_j\end{aligned}$$

with boundary corrected kernel

$$k_h^u(u - x) = \frac{\kappa\left(\frac{u-x}{h}\right)}{\int_{\mathcal{X}_j} \kappa\left(\frac{u-v}{h}\right) \mathrm{d}v}.$$

We conclude that the local linear smooth backfitting estimator $\widehat{m}_j$ is asymptotically equivalent to the local linear estimator $\widetilde{m}_j^n$ in the oracle model. We formulate this asymptotic equivalence as a first corollary of Theorem 2.4.1. In particular, it implies that the estimators have the same first order asymptotic properties.

Corollary 2.4.2. *Make assumptions (A1) ,..., (A4). Then it holds uniformly over* $1 \le k \le d$ *and* $a_j \le x_j \le b_j$ *that*

$$\left| \widehat{m}_j^n(x_j) - \widetilde{m}_j^n(x_j) \right| = o_P(h^2).$$

For $x_j \in (a_j + 2h, b_j - 2h)$ the bias term β_j simplifies and we have that

$$\beta_j(x_j) = h^2 \frac{1}{2} m_j''(x_j) \int k(v) v^2 \mathrm{d}v.$$

This implies the following corollary of Theorem 2.4.1.

Corollary 2.4.3. *Make assumptions (A1) ,..., (A4). Then it holds uniformly over* $1 \le k \le d$ *and* $a_j + 2h \le x_j \le b_j - 2h$ *that*

$$\Bigg| \widehat{m}_j^n(x_j) - m_j(x_j) - \frac{1}{2}\left(m_j''(x_j) - \int m_j''(u_j) p_j(u_j) \mathrm{d}u_j \right) h^2 \int k(v) v^2 \mathrm{d}v \\ -v_j(x_j) \Bigg| = o_P(h^2).$$

Corollary 2.4.3 can be used to derive the asymptotic distribution of $\widehat{m}_j^n(x_j)$ for an $x_j \in (a_j, b_j)$. Under the additional assumption that $\sigma_j^2(u) = \mathbb{E}[\varepsilon_i^2 | X_{ij} = u]$ is continuous in $u = x_j$ we get under (A1),..., (A4) that $n^{2/5}(\widehat{m}_j^n(x_j) - m_j(x_j))$ has an asymptotic normal distibution with mean

$$c_h \frac{1}{2} \left(m_j''(x_j) - \int m_j''(u_j) p_j(u_j) \mathrm{d}u_j \right) \int k(v) v^2 \mathrm{d}v$$

and variance

$$c_h^{-1} \sigma_j^2(x_j) p_j^{-1}(x_j) \int k(v)^2 \mathrm{d}v.$$

This is equal to the asymptotic limit distribution of the classical local linear estimator in the oracle model in accordance with Corollary 2.4.2. Now, we come to the proof of Theorem 2.4.1.

First, we define the operator $\mathcal{S}_n = (\mathcal{S}_{n,0}, \mathcal{S}_{n,1}, \ldots, \mathcal{S}_{n,2d}) : \mathcal{G}^n \to \mathcal{G}^n$ with

$$\mathcal{G}^n = \{(g_0, \ldots, g_{2d}) | g_0 \in \mathbb{R}, g_l, g_{l'} \in \mathrm{L}_2(p_l) \text{ with } P_0(g_l) = 0 \text{ for } l = 1, \ldots, d\},$$

where $\mathcal{S}_{n,k}$ maps $g = (g_0, \ldots, g_{2d}) \in \mathcal{G}^n$ to f_k with

$$f_0 = \mathcal{P}_0 \left(\sum_{1 \le l \le 2d} g_l \right) = \mathcal{P}_0 \left(\sum_{d+1 \le l \le 2d} g_l \right) \in \mathbb{R},$$

$$f_k(x_k) = \mathcal{P}_k \left(\sum_{0 \le l \le 2d, l \ne k} g_l \right)(x),$$

for $1 \le k \le 2d$. With this notation we can rewrite the backfitting equation (2.2.4) as

$$\bar{m}^n(Y) = \widehat{m}^n + \mathcal{S}_n \widehat{m}^n, \tag{2.4.3}$$

where for $z \in \mathbb{R}^n$ we define $\bar{m}^n_0(z) = \bar{z} = \frac{1}{n}\sum_{i=1}^n z_i$ and for $1 \le j \le d$,

$$\begin{aligned}\bar{m}^n_j(z)(x_j) &= \widehat{p}_j(x_j)^{-1}\frac{1}{n}\sum_{i=1}^n (z_i - \bar{z}) \int K_h^{X_i}(X_i - x)\mathrm{d}x_{-j}, \\ \bar{m}^n_{j'}(z)(x_j) &= \widehat{p}^{**}_j(x_j)^{-1}\frac{1}{n}\sum_{i=1}^n (X_{ij} - x_j) z_i \int K_h^{X_i}(X_i - x)\mathrm{d}x_{-j}.\end{aligned}$$

The following lemma shows that $I + \mathcal{S}_n$ is invertible on the event $\mathcal{E}$. Here we denote the identity operator by I.

Lemma 2.4.4. *On the event $\mathcal{E}$ the operator $I + \mathcal{S}_n : \mathcal{G}^n \to \mathcal{G}^n$ is invertible.*

Proof. Suppose that for some $g \in \mathcal{G}^n$ it holds that $(I + \mathcal{S}_n)(g) = 0$. We have to show that this implies $g = 0$.
For the proof of this claim note that $g_k + \mathcal{S}_{n,k}(g)$ is the orthogonal projection of $\sum_{j=0}^{2d} g_j$ onto $\mathcal{H}_k$. Furthermore, we have that g_k is an element of $\mathcal{H}_k$. This gives that

$$\left\langle g_k, \sum_{j=0}^{2d} g_j \right\rangle_n = 0.$$

Summing over k gives

$$\left\langle \sum_{j=0}^{2d} g_j, \sum_{j=0}^{2d} g_j \right\rangle_n = 0.$$

According to Lemma 2.3.1 on the event $\mathcal{E}$ we have $g_0 = 0$ and $g_j \equiv 0$ for $j = 1, \dots, 2d$. This concludes the proof of the lemma. □

One can show that under conditions (A1) ,..., (A4) the probability of the event $\mathcal{E}$ converges to one. Note that we have assumed that $\mathcal{X} = \prod_{j=1}^d \mathcal{X}_j$. We conclude that under (A1) ,..., (A4) $I + \mathcal{S}_n$ is invertible with probability tending to one. Thus we have that with probability tending to one

$$\begin{aligned}&\widehat{m}^n - m - \bar{m}^n(\varepsilon) - \beta_n + \Delta_n m + \Delta_n \beta_n \\ &\quad = (I + \mathcal{S}_n)^{-1}(I + \mathcal{S}_n)(\widehat{m}^n - m - \bar{m}^n(\varepsilon) - \beta_n + \Delta_n m + \Delta_n \beta_n),\end{aligned} \tag{2.4.4}$$

where m has components $m_0, \dots, m_{2d}$ with $m_0, \dots, m_d$ as in (2.4.1) and with $m_{j'} = m'_j$ for $1 \le j \le d$. Furthermore, $\beta(x)$ has components $\beta_0 = 0$, $\beta_j(x_j)$ and

$$\beta_{j'}(x_j) = \frac{1}{2} m''_j(x_j)\, \frac{b_{j,0}(x_j) b_{j,3}(x_j) - b_{j,1}(x_j) b_{j,2}(x_j)}{b_{j,0}(x_j) b_{j,2}(x_j) - b_{j,1}(x_j)^2} h$$

for $j = 1, \dots, d$ with $b_{j,l}(x_j)$ defined above. Additionally, the norming constants are given by

$$\begin{aligned}
(\Delta_n \beta)_j &= \int \beta_j(x_j)\widehat{p}_j(x_j)\mathrm{d}x_j,\\
(\Delta_n m)_j &= \int m_j(x_j)\widehat{p}_j(x_j)\mathrm{d}x_j,\\
(\Delta_n \beta)_{j'} &= (\Delta_n m)_{j'} = 0 \quad \text{for } j = 1, \dots, d,\\
(\Delta_n \beta)_0 &= \sum_{j=1}^{d} \int \beta_{j'}(x_j)\widehat{p}_j^*(x_j)\mathrm{d}x_j,\\
(\Delta_n m)_0 &= \sum_{j=1}^{d} \int m_{j'}(x_j)\widehat{p}_j^*(x_j)\mathrm{d}x_j.
\end{aligned}$$

One can verify that for $a_j + 2h_j \leq x_j \leq b_j - 2h_j$ one has $\beta_{j'}(x_j) = o_P(h)$. We have already seen that $\beta_j(x_j) = \frac{1}{2} m_j''(x_j) \int k(v)v^2 \mathrm{d}v + o_P(h^2)$ holds for such x_j.
For the statement of Theorem 2.4.1 we have to show that for $1 \leq j \leq d$ the j-th component on the left hand side of equation (2.4.4) is of order $o_P(h^2)$ uniformly for $a_j + 2h \leq x_j \leq b_j - 2h$. For a proof of this claim we first analyze the term

$$\begin{aligned}
D_n &= (I + \mathcal{S}_n)(\widehat{m}^n - m - \bar{m}^n(\varepsilon) - \beta_n + \Delta_n m + \Delta_n \beta_n) \qquad (2.4.5)\\
&= \bar{m}^n(Y) - (I + \mathcal{S}_n)(m + \bar{m}^n(\varepsilon) + \beta_n - \Delta_n m - \Delta_n \beta_n).
\end{aligned}$$

For this sake we split the term $\bar{m}^n(Y)$ into the sum of a stochastic variance term and a deterministic expectation term:

$$\bar{m}^n(Y) = \bar{m}^n(\varepsilon) + \sum_{j=0}^{d} \bar{m}^n(\mu_{n,j}), \qquad (2.4.6)$$

where

$$\begin{aligned}
\varepsilon &= Y - \sum_{j=0}^{d} \mu_{n,j},\\
\mu_{n,j} &= (m_j(X_{ij}))_{i=1,..,n} \quad \text{for } j = 1, \dots, d,\\
\mu_{n,0} &= (m_0)_{i=1,..,n}.
\end{aligned}$$

We write $D_n = D_n^\beta + D_n^\varepsilon$, with

$$\begin{aligned}
D_n^\beta &= \sum_{j=0}^{d} \bar{m}^n(\mu_{n,j}) - (I + \mathcal{S}_n)(m + \beta_n - \Delta_n m - \Delta_n \beta_n),\\
D_n^\varepsilon &= \mathcal{S}_n(\bar{m}^n(\varepsilon)).
\end{aligned}$$

The following lemma treats the conditional expectation term D^β.

Lemma 2.4.5. *Assume (A1) ,..., (A4). It holds* $D^{\beta}_{n,0} = o_p(h^2)$ *and*

$$\sup_{x_k \in \mathcal{X}_k} \left| D^{\beta}_{n,k}(x_k) \right| = \begin{cases} o_p(h^2) & \text{for } 1 \le k \le d, \\ o_p(h) & \text{for } d+1 \le k \le 2d. \end{cases}$$

Proof. The lemma follows by application of lengthy calculations using second order Taylor expansions for $m_j(X_{ij})$ and by application of laws of large numbers. □

We now turn to the variance term.

Lemma 2.4.6. *Assume (A1) ,..., (A4). It holds* $D^{\varepsilon}_{n,0} = o_p(h^2)$ *and*

$$\sup_{x_k \in \mathcal{X}_k} \left| D^{\varepsilon}_{n,k}(x_k) \right| = \begin{cases} o_p(h^2) & \text{for } 1 \le k \le d, \\ o_p(h) & \text{for } d+1 \le k \le 2d. \end{cases}$$

Proof. One can easily check that $D^{\varepsilon}_{n,k}(x_k)$ consists of weighted sums of ε_i where the weights are of the same order for all $1 \le i \le n$. For fixed x_k the sums are of order $O_P(n^{-1/2})$ for $1 \le k \le d$ and of order $O_P(h^{-1}n^{-1/2})$ for $d+1 \le k \le 2d$. Using the conditional moment conditions on ε_i in Assumption (A3) we get the uniform rates stated in the lemma. □

It remains to study the behaviour of $(I + \mathcal{S}_n)^{-1} D^{\varepsilon}_n$ and $(I + \mathcal{S}_n)^{-1} D^{\beta}_n$. We use a small transformation of $\mathcal{S}_n$ here which is better suitable for an inversion. Define the following 2×2 matrix $A_{n,k}(x)$ by

$$A_{n,k}(x) = \frac{1}{\widehat{p}_k \widehat{p}^{**}_k - (\widehat{p}^{*}_k)^2} \begin{pmatrix} \widehat{p}^{**}_k \widehat{p}_k & \widehat{p}^{**}_k \widehat{p}^{*}_k \\ \widehat{p}^{*}_k \widehat{p}_k & \widehat{p}_k \widehat{p}^{**}_k \end{pmatrix} (x_k).$$

Furthermore, define the $2d \times 2d$ matrix $A_n(x)$ where the elements with indices $(k,k), (k,k')$, $(k',k), (k',k')$ are equal to the elements of $A_{n,k}(x)$ with indices $(1,1), (1,2), (2,1), (2,2)$. We now define $\tilde{\mathcal{S}}_n$ by the equation $I + \tilde{\mathcal{S}}_n = A_n(I + \mathcal{S}_n)$. Below we make use of the fact that $\tilde{\mathcal{S}}_n$ is of the form

$$\tilde{\mathcal{S}}_{n,k} m(x) = \sum_{l \notin \{k,k'\}} \int q_{k,l}(x_k, u) m_l(u) \mathrm{d}u + \sum_{l \in \{k,k'\}} \int q_l(u) m_l(u) \mathrm{d}u, \tag{2.4.7}$$

$$\tilde{\mathcal{S}}_{n,k'} m(x) = \sum_{l \notin \{k,k'\}} \int q_{k',l}(x_k, u) m_l(u) \mathrm{d}u + \sum_{l \in \{k,k'\}} \int q_l(u) m_l(u) \mathrm{d}u \tag{2.4.8}$$

for $1 \le k \le d$ with some random functions $q_{k,l}, q_l$ which fulfill

$$\int q_{k,l}(x_k, u)^2 \mathrm{d}u, \int q_k(u)^2 \mathrm{d}u = O_P(1)$$

uniformly over $1 \le k, l \le 2d$ and x_k. Note that we need $\tilde{\mathcal{S}}_n$ because $\mathcal{S}_n$ can not be written in the form of (2.4.7) and (2.4.8). The operator $\tilde{\mathcal{S}}_n$ differs from $\mathcal{S}_n$ in the h-neighbourhood of the boundary by terms of order h^2. Otherwise the difference is of order $o_p(h^2)$. Outside of the h-neighbourhood of the boundary, for $n \to \infty$, the matrix $A_n(x)$ converges to the identity matrix.

Thus $\tilde{\mathcal{S}}_n$ is a second order modification of $\mathcal{S}_n$ with the advantage of having (2.4.7)-(2.4.8).
For our further discussion we now introduce the space $\mathcal{G}^0$ of tuples $f = (f_0, f_1, \ldots, f_{2d})$ with $f_0 = 0$ and $f_k, f_{k'} : \mathcal{X}_k \to \mathbb{R}$ with $\int f_k(x_k)p_k(x_k)\mathrm{d}x_k = 0$ and endow it with the norm $\|f\|^2 = \sum_{k=1}^d (f_k(x_k)^2 + f_{k'}(x_k)^2)p_k(x_k)\mathrm{d}x_k$. The next lemma shows that the norm of $H_n(I+\mathcal{S}_n)^{-1}D_n^\varepsilon$ and $H_n(I+\mathcal{S}_n)^{-1}D_n^\beta$ is of order $o_P(h^2)$. Here H_n is a diagonal matrix where the first $d+1$ diagonal elements equal 1. The remaining elements are equal to h.

Lemma 2.4.7. *Assume (A1) ,..., (A4). Then it holds that* $\|H_n(I+\mathcal{S}_n)^{-1}D_n^*\| = \|H_n(I+\tilde{\mathcal{Q}}_n)^{-1}A_nD_n^*\| = o_P(h^2)$ *for* $D_n^* = D_n^\varepsilon$ *and* $D_n^* = D_n^\beta$.

Proof. Define $\bar{D}_n^\varepsilon$ and $\bar{D}_n^\beta$ by $\bar{D}_{n_k}^\varepsilon(x_k) = D_{n,k}^\varepsilon(x_k) - \int D_{n,k}^\varepsilon(u_k)p_k(u_k)\mathrm{d}u_k$ and $\bar{D}_{n,k}^\beta(x_k) = D_{n,k}^\beta(x_k) - \int D_{n,k}^\beta(u_k)p_k(u_k)\mathrm{d}u_k$ for $1 \le k \le d$ and $\bar{D}_{n,k}^\varepsilon = D_{n,k}^\varepsilon$ and $\bar{D}_{n,k}^\beta = D_{n,k}^\beta$, otherwise. It can be checked that it suffices to prove the lemma with D_n^ε and D_n^β replaced by $\bar{D}_n^\varepsilon$ and $\bar{D}_n^\beta$. Note that $\bar{D}_n^\varepsilon$ and $\bar{D}_n^\beta$ are elements of $\mathcal{G}^0$. For the proof of this claim we compare the operator $\tilde{\mathcal{S}}_n$ with the operator $\mathcal{S}_0$ defined by $\mathcal{S}_{0,0}(g) = 0$, $\mathcal{S}_{0,k'}(g)(x_k) = 0$ and

$$\mathcal{S}_{0,k}(g)(x_k) = \sum_{j \neq k} \int_{\mathcal{X}_j} g_j(u_j)\frac{p_{j,k}(u_j,x_k)}{p_k(x_k)}\mathrm{d}u_j$$

for $1 \le k \le d$. By standard kernel smoothing theory one can show that

$$\sup_{g \in \mathcal{G}^0, \|g\| \le 1} \|(\mathcal{S}_0 - H_n\tilde{\mathcal{S}}_nH_n^{-1})g\| = o_P(1).$$

For the proof of this claim one makes use of the fact that non-vanishing differences in the h-neighbourhood of the boundary are asymptotically negligible in the calculation of the norm because the size of the neighbourhood converges to zero.
In the next lemma we show that $I + \mathcal{S}_0$ has a bounded inverse. This implies the statement of the lemma by applying the following expansion:

$$\begin{aligned}
&(I + H_n\tilde{\mathcal{S}}_nH_n^{-1})^{-1} - (I+\mathcal{S}_0)^{-1} \\
&\quad= (I+\mathcal{S}_0)^{-1}((I+\mathcal{S}_0)(I+H_n\tilde{\mathcal{S}}_nH_n^{-1})^{-1} - I) \\
&\quad= (I+\mathcal{S}_0)^{-1}(((I+H_n\tilde{\mathcal{S}}_nH_n^{-1})(I+\mathcal{S}_0)^{-1})^{-1} - I) \\
&\quad= (I+\mathcal{S}_0)^{-1}((I+(H_n\tilde{\mathcal{S}}_nH_n^{-1} - \mathcal{S}_0)(I+\mathcal{S}_0)^{-1})^{-1} - I) \\
&\quad= (I+\mathcal{S}_0)^{-1}\sum_{j=1}^{\infty}(I+(-1)^j(H_n\tilde{\mathcal{S}}_nH_n^{-1} - \mathcal{S}_0)(I+\mathcal{S}_0)^{-1})^j.
\end{aligned}$$

This shows the lemma because of

$$H_n(I+\tilde{\mathcal{S}}_n)^{-1}A_nD_n^* = H_n(I+\tilde{\mathcal{Q}}_n)^{-1}H_n^{-1}H_nA_nD_n^* = (I+H_n\tilde{\mathcal{S}}_nH_n^{-1})^{-1}H_nA_nD_n^*$$

for $D_n^* = D_n^\varepsilon$ and $D_n^* = D_n^\beta$. □

Lemma 2.4.8. *Assume (A1) ,..., (A4). The operator* $I + \mathcal{S}_0 : \mathcal{G}^0 \to \mathcal{G}^0$ *is bijective and has a bounded inverse.*

Proof. For a proof of this claim it suffices to show that the operator $I+\mathcal{S}_* : \mathcal{G}^* \to \mathcal{G}^*$ is bijective and has a bounded inverse where $\mathcal{G}^*$ is the space of tuples $f = (f_1, \dots, f_d)$ where $f_k : \mathcal{X}_j \to \mathbb{R}$ with $\int f_k(x_k)\mathrm{d}x_k = 0$ with norm $\|f\|^2 = \sum_{k=1}^d f_k(x_k)^2 p_k(x_k)\mathrm{d}x_k$ and

$$\mathcal{S}_{*,k}(g)(x_k) = \sum_{j\neq k} \int_{\mathcal{X}_j} g_j(u_j)\frac{p_{j,k}(u_j,x_k)}{p_k(x_k)}\mathrm{d}u_j$$

for $1 \le k \le d$. We apply the bounded inverse theorem. For an application of this theorem we have to show that $I + \mathcal{S}_*$ is bounded and bijective. It can easily be seen that the operator is bounded. It remains to show that it is surjective. We show that

(i) $(I + \mathcal{S}_*)g^n \to 0$ for a sequence $g^n \in \mathcal{G}^*$ implies that $g^n \to 0$.

(ii) $\int g_k(I + \mathcal{S}_*)_k r(x_k)p_k(x_k)\mathrm{d}x_k = 0$ for all $g \in \mathcal{G}^*$ implies that $r = 0$.

Note that (i) implies that $\mathcal{G}^{**} = \{(I+\mathcal{S}_*)g : g \in \mathcal{G}^*\}$ is a closed subset of $\mathcal{G}^*$. To see this suppose that $(I + \mathcal{S}_*)g^n \to g$ for $g, g^n \in \mathcal{G}^*$. Then (i) implies that g^n is a Cauchy sequence and thus g^n has a limit in $\mathcal{G}^*$ which implies that $(I + \mathcal{S}_*)g^n$ has a limit in $\mathcal{G}^{**}$. Thus $\mathcal{G}^{**}$ is closed.
From (ii) we conclude that the orthogonal complement of $\mathcal{G}^{**}$ is equal to $\{0\}$. Thus the closure of $\mathcal{G}^{**}$ is equal to $\mathcal{G}^*$. This shows that $\mathcal{G}^* = \mathcal{G}^{**}$ because $\mathcal{G}^{**}$ is closed. We conclude that $(I+\mathcal{S}_*)$ is surjective.
It remains to show (i) and (ii). Fo a proof of (i) note that $(I + \mathcal{S}_*)g^n \to 0$ implies that

$$\int g_k^n(x_k)(I + \mathcal{S}_*)_k g^n(x_k)p_k(x_k)\mathrm{d}x_k \to 0$$

which shows

$$\sum_{k=1}^d \int g_k^n(x_k)^2 p_k(x_k)\mathrm{d}x_k + \sum_{k\neq j} g_k^n(x_k)p_{kj}(x_k,x_j)g_j(x_j)\mathrm{d}x_k\mathrm{d}x_j \to 0.$$

Thus we have

$$\mathbb{E}[(\sum_{k=1}^d g_k(X_{ik}))^2] \to 0.$$

By application of Proposition 2.3.4 (ii) we get that $\max_{1\le k\le d}\mathbb{E}[g_k(X_{ik})^2] \to 0$, which shows (i).
Claim (ii) can be seen by a similar argument. Note that

$$\int g_k(I + \mathcal{S}_*)_k r p_k(x_k)\mathrm{d}x_k = 0$$

for all $g \in \mathcal{G}^*$ implies that $\int r_k(I + \mathcal{S}_*)_k r(x_k)p_k(x_k)\mathrm{d}x_k = 0$. □

We now apply the results stated in the lemma for the final proof of Theorem 2.4.1.

Proof of Theorem 2.4.1. From (2.4.4) and Lemma 2.4.7 we know that the L_2 norm of $\widehat{m}^n - m - \bar{m}^n(\varepsilon) - \beta_n + \Delta_n m + \Delta_n\beta_n = H_n(I + \mathcal{S}_n)^{-1}(D_n^\varepsilon + D_n^\beta) = H_n(I + \tilde{\mathcal{S}}_n)^{-1}A_n(D_n^\varepsilon + D_n^\beta)$ is of order $o_P(h^2)$. Note that $H_n(I+\tilde{\mathcal{S}}_n)^{-1} = H_n - H_n\tilde{\mathcal{S}}_n(I+\tilde{\mathcal{S}}_n)^{-1}$. We already know that the sup norm of all components in $H_nA_n(D_n^\varepsilon + D_n^\beta)$ are of order $o_P(h^2)$. Thus, it remains to check that the sup norm of the components of $H_n\tilde{\mathcal{S}}_n(I+\tilde{\mathcal{S}}_n)^{-1}A_n(D_n^\varepsilon + D_n^\beta)$ is of order $o_P(h^2)$. But this follows by application of the just mentioned bound on the L_2 norm of $H_n(I+\mathcal{S}_n)^{-1}(D_n^\varepsilon + D_n^\beta)$, by equations (2.4.7), (2.4.8), and the bounds for the random functions $q_{k,l}$ and q_l mentioned after the statement of the equations. One gets a bound for the sup norms by application of the Cauchy Schwarz inequality. □

3 Optimal Convergence Rates of Deep Neural Networks in a Classification Setting

This chapter follows the paper [96]. We included some slight modifications in order to embed it into this thesis.

We establish optimal convergence rates up to a log-factor for a class of deep neural networks in a classification setting under a restraint sometimes referred to as the Tsybakov noise condition. We construct classifiers in a general setting where the boundary of the bayes-rule can be approximated well by neural networks. Corresponding rates of convergence are proven with respect to the misclassification error. It is then shown that these rates are optimal in the minimax sense if the boundary satisfies a smoothness condition. Non-optimal convergence rates already exist for this setting. Our main contribution lies in improving existing rates and showing optimality, which was an open problem. Furthermore, we show almost optimal rates under some additional restraints which circumvent the curse of dimensionality. For our analysis we require a condition which gives new insight on the restraint used. In a sense it acts as a requirement for the "correct noise exponent" for a class of functions.

3.1 Introduction

We consider i.i.d. data $(Y_i, X_i)_{i=1}^n$ with $Y_i \in \{0,1\}$ and $X_i \in \mathbb{R}^d$. Our goal is to provide an estimator of the form $\widehat{Y} = \mathbb{1}(X \in \widehat{G})$ where $\widehat{G}$ is constructed with a neural network which approximates Y well with respect to the misclassification error. We show optimal convergence rates under the following two conditions. First, the underlying distribution $\mathbb{Q}$ satisfies a noise condition as in [126] described below. Second, the boundary of the set

$$G^*_{\mathbb{Q}} := \left\{ x \;\middle|\; f_{\mathbb{Q}}(x) \geq \frac{1}{2} \right\}$$

with $f_{\mathbb{Q}}(x) := \mathbb{Q}(Y = 1|X = x)$ satisfies certain regularity conditions.

Neural Networks have shown outstanding results in many classification tasks such as image recognition [51], language recognition [26], cancer recognition [70], and other disease detection [79]. Our work follows current approaches in the statistical literature to explain the success of neural networks, e.g. the impactful contributions [73, 109]. The objective is to fill a gap in the literature by proving optimal convergence rates in a specific setting which was also considered in [71]. We focus on deep feedforward neural networks with ReLU-activation functions. Deep networks have been considered in many theoretical articles [73, 74, 104, 105] and have proven useful in many applications [78, 108]. Intuitively, we wish to approximate the set $G^*_{\mathbb{Q}}$ directly instead of approximating the regression function $f_{\mathbb{Q}}$. The classification setting we consider is similar to the setting given in [91, 126]. In particular, we assume that $\mathbb{Q}$ satisfies a noise condition which can be described as follows. For $\mathbb{Q}$-measurable sets G_1, G_2 define

$$\begin{aligned} d_{f_{\mathbb{Q}}}(G_1, G_2) &:= \int_{G_1 \Delta G_2} |2f_{\mathbb{Q}}(x) - 1| \; \mathbb{Q}_X(\mathrm{d}x), \\ d_{\Delta}(G_1, G_2) &:= \mathbb{Q}_X(G_1 \Delta G_2). \end{aligned}$$

The condition then states that there exists a constant $\kappa \geq 1$ such that

$$d_{f_{\mathbb{Q}}}(G, G^*_{\mathbb{Q}}) \geq c_1 d^{\kappa}_{\Delta}(G, G^*_{\mathbb{Q}}) \tag{3.1.1}$$

for some constant $c_1 > 0$ and all G. This requirement is sometimes referred to as the Tsybakov noice condition. It can be interpreted as a restraint on the probability distribution regarding regions close to the boundary where $f_{\mathbb{Q}}(x) = \frac{1}{2}$. Roughly speaking, it forces the mass to decay at a certain rate when one approaches this boundary. Using this, one can achieve rates approaching n^{-1} for small κ, i.e. if there is not much mass in the region around $f_{\mathbb{Q}}(x) = \frac{1}{2}$. The condition has been used in many statistical articles considering classification such as [3] and [131] who analyse support vector machines. Similarly to [91], we show optimal convergence rates in the case where the boundary of $G^*_{\mathbb{Q}}$ satisfies certain regularity conditions, i.e. is similar to an element of a Dudley class [32]. More precisely, we consider sets which are slightly more general then the sets given in [104]. While many other approximation results using neural networks exist, see [29] using sigmoid activation functions or [132] using piecewise linear functions among others, the methods used in [104] inspired us to obtain the results for our setting. The sets they consider have been used in many articles such as [71, 105]. As an estimator, we use a risk minimizer of the empirical version of the misclassification error. Precisely calculating this estimator involves finding a global minimum of a highly non-convex loss with respect to the parameters of a neural network. Typically, such calculations are not feasible in practice. Thus, the results we provide are theoretical in nature and do not have direct useful applications, as is typical for results of this kind [74, 109]. From our point of view, the main value of current contributions is to show results such as consistency in situations which are typical for statisticians using relatively simple classes of neural networks. In time, the techniques developed may be used to show claims in cases which are closer to those encountered in reality and using classes of neural networks which are closer to those used in practise.

A lot of work has been done regarding consistency of feedforward deep neural networks. [109] prove optimal convergence rates with respect to the uniform norm in a regression setting. Among others, similar results were given by [64] for non-continuous regression functions with respect to the L_2-norm, [75] who did not use a sparsity constraint, and [5]. Regarding results for classification, [105] show convergence rates considering the misclassification error in a noiseless setting. Consistency results which include condition (3.1.1) in the assumptions are given by

[12, 63, 74]. In contrast to our approach, the previously mentioned articles attempt to estimate the regression function $f_{\mathbb{Q}}$ instead of directly estimating the set $G^*_{\mathbb{Q}}$. Additionally, while some obtain optimal convergence rates, the settings do not correspond to the setting given in [126]. In particular, the (optimal) convergence rates differ from ours in these papers. A very interesting contribution was made by [71] who consider an almost identical situation to ours, while their estimators differ. However, the rates they obtain are not optimal in the minimax sense.

3.1.1 Contribution

Our contribution includes the following.

- First and foremost, Theorem 3.4.1 together with Corollary 3.3.6 prove optimal convergence rates in the minimax sense for the setting described above. To the best of our knowledge, we are the first to obtain optimal convergence rates using neural networks corresponding to the setting given in [126] and thus close this gap in the literature.
- Theorem 3.3.4 establishes convergence rates in a general setting, where the boundary of the set $G^*_{\mathbb{Q}}$ can be well approximated by neural networks. This enables us to prove rates in a variety of settings. We use this theorem to prove optimal convergence rates under an additional constraint, which circumvents the curse of dimensionality, in the sense that the rates do not decrease exponentially in the dimension d.
- In order to prove the results stated here, we require a condition which together with condition (3.1.1) forces κ to be the "correct parameter" for the distribution $\mathbb{Q}$. We believe that this condition may bring new insights to condition (3.1.1) and may be helpful in other situations where (3.1.1) is required.

3.1.2 Outline

After introducing some notation, we rigorously introduce the problem at hand in Section 3.2. Here, we also provide some convergence results considering empirical risk minimizers with respect to arbitrary sets. These results are then used to prove our main consistency theorems regarding neural networks in Section 3.3. Section 3.4 includes the corresponding lower bounds followed by some concluding remarks in Section 3.5.

3.1.3 Notation

We introduce some general notation which is used throughout this article.

For $x \in \mathbb{R}$, let $\lfloor x \rfloor := \max\{k \in \mathbb{Z} \mid k \leq x\}$ and $\lceil x \rceil := \min\{k \in \mathbb{Z} \mid k \geq x\}$. Let λ be the Lebesgue measure. For a function $g : \Omega \subseteq \mathbb{R}^s \to \mathbb{R}$ and $k \in \mathbb{N}$ denote by

$$\|g\|_\infty := \sup_{x \in \Omega} |g(x)|, \quad \|g\|_{L^k} := \left(\int |g|^k \, \lambda(\mathrm{d}x) \right)^{\frac{1}{k}}$$

the uniform-norm and the L^k-norm, respectively. Note that we omit the dependence on Ω in the notation. For $x \in \mathbb{R}^s$, let $\|x\|_2$ and $\|x\|_\infty$ be the euclidean-norm and the uniform-norm, respectively. For $j \in \{1, \dots, s\}$ let

$$x_{-j} := (x_1, \dots, x_{j-1}, x_{j+1}, \dots, x_s).$$

Additionally, let

$$\mathcal{B}_r(x) := \{y \in \mathbb{R}^s \mid \|x - y\|_\infty \leq r\},$$
$$\mathcal{B}_r^\circ(x) := \{y \in \mathbb{R}^s \mid \|x - y\|_\infty < r\}.$$

For $a \in \mathbb{N}^s$ let $|a| := \sum_{i=1}^s a_i$.

Now, let $\beta \in (0, \infty)$. Define $m := \max\{k \in \mathbb{N} \mid k < \beta\}$ and $\omega := \beta - m > 0$. For $f \in \mathcal{C}\big([0,1]^s, \mathbb{R}\big)$ let

$$\|f\|_{\mathcal{C}^\beta} := \sum_{|\alpha| \leq m} \|\partial^\alpha f\|_\infty + \sum_{|\alpha| = m} \sup_{x \neq y} \frac{|\partial^\alpha f(x) - \partial^\alpha f(y)|}{|x - y|_\infty^\omega}$$

be the Hoelder-norm. For $B > 0$, define the class of Hoelder-continuous functions by

$$\mathcal{F}_{\beta,B,s} := \Big\{ f \in \mathcal{C}\big([0,1]^s, \mathbb{R}\big) \;\Big|\; \|f\|_{\mathcal{C}^\beta} \leq B \Big\}.$$

Let $G_1, G_2 \subseteq \Omega$ be two subsets. We write

$$G_1 \Delta G_2 := (G_1 \backslash G_2) \cup (G_2 \backslash G_1)$$

for their symmetric difference and

$$\mathbb{1}(x \in G_1) := \begin{cases} 1, & \text{for } x \in G_1, \\ 0, & \text{otherwise} \end{cases}$$

for the indicator function corresponding to G_1.

3.2 General Convergence Results

In this section, we state our results in a relatively general setting. The results on neural networks in the next section only consider the case where $\mathbb{Q}_X$ has a bounded density with respect to the Lebesgue measure. Our setup is similar to the binary classification setup of [126].

3.2.1 Classification Setup

Let $(X_i, Y_i)_{i=1}^n$ be *i.i.d.* observations distributed according to some probability measure $\mathbb{Q}$, where $X_i \in \mathbb{R}^d$ and $Y_i \in \{0,1\}$. Denote by $\mathbb{Q}_X$ the marginal probability distribution with respect to $X \in \mathbb{R}^d$. The goal is to predict $Y \in \{0,1\}$ when observing $X \in \mathbb{R}^d$, where (X, Y) is distributed according to $\mathbb{Q}$ independently of $(X_i, Y_i)_{i=1}^n$ using classifiers of the form

$$\widehat{Y} := \mathbb{1}(X \in \widehat{G})$$

for some $\mathbb{Q}$-measurable set $\widehat{G} \subseteq \mathbb{R}^d$. Note that a classifier is uniquely determined by $\widehat{G}$. Performance is measured by the misclassification error

$$R(\widehat{G}) := \mathbb{P}\big(Y \neq \widehat{Y}\big) = \mathbb{E}\Big[\big(Y - 1(X \in \widehat{G})\big)^2\Big].$$

For $f_{\mathbb{Q}}(x) := \mathbb{E}[Y|X = x] = \mathbb{Q}(Y = 1|X = x)$ the set

$$G^*_{\mathbb{Q}} := \Big\{x \;\Big|\; f_{\mathbb{Q}}(x) \geq \frac{1}{2}\Big\}$$

is a so called bayes rule and thus minimizes the misclassification error. Classification can equivalently be seen as estimation of $G^*_{\mathbb{Q}}$ by the set $\widehat{G}$, which is therefore equally referred to as classifier. For a $\mathbb{Q}$-measureable set $G \subseteq \mathbb{R}^d$ let

$$R_n(G) = \frac{1}{n}\sum_{i=1}^n 1\big(Y_i \neq 1(X_i \in G)\big) = \frac{1}{n}\sum_{i=1}^n \big(Y_i - 1(X_i \in G)\big)^2$$

be the empirical version of the misclassification error $R(G)$. We consider empirical risk minimization classifiers defined by

$$\widehat{G}_n := \underset{G \in \mathcal{N}_n}{\arg\min}\, R_n(G)$$

where $\mathcal{N}_n$ is some finite collection of $\mathbb{Q}$-measurable sets for all $n \in \mathbb{N}$.

3.2.2 Consistency Results

Proposition 3.2.1 establishes convergences rates for estimating $G^*_{\mathbb{Q}}$ using $\widehat{G}_n$ under certain conditions on $\mathcal{N}_n$ and $\mathbb{Q}$. For the loss function, we consider a slight generalization of the misclassification error

$$\mathbb{E}\big[\big(R(\widehat{G}_n) - R(G^*_{\mathbb{Q}})\big)^p\big] = \mathbb{E}\big[d^p_{f_{\mathbb{Q}}}(G_n, G^*_{\mathbb{Q}})\big]$$

for $p \geq 1$. The proposition is somewhat similar to Theorem 2 from [91]. In contrast to our approach, they consider the discrimination of two probability distributions with underlying distribution functions and do not allow for non-optimal convergence rates. The proposition is an important component for the proof of our main theorem given in Section 3.3. The proofs of this section can be found in Subsection 7.2.1.

Proposition 3.2.1. *Let $\tau_n > 0$ be a monotonically increasing sequence. Let $\mathfrak{Q}$ be a class of potential joint distributions $\mathbb{Q}$ of (X, Y) and $\mathcal{N}_n$ be a collection of subsets of $\mathbb{R}^d$ for all $n \in \mathbb{N}$ such that the following conditions hold.*

(i) For all $\mathbb{Q} \in \mathfrak{Q}$ all sets in $\bigcup_{n\in\mathbb{N}} \mathcal{N}_n$ and $G^*_{\mathbb{Q}}$ are $\mathbb{Q}$-measurable.

(ii) There exists a constant $\kappa \geq 1$ such that

$$d_{f_{\mathbb{Q}}}(G, G^*_{\mathbb{Q}}) \geq c_1 d^{\kappa}_{\Delta}(G, G^*_{\mathbb{Q}})$$

for some constant $c_1 > 0$, all $G \in \bigcup_{n\in\mathbb{N}} \mathcal{N}_n$ and all $\mathbb{Q} \in \mathfrak{Q}$.

Additionally, we assume that there is a constant $N_0 \in \mathbb{N}$ such that for all $n \geq N_0$ the following holds.

(iii) There is a constant $c_2 > 0$ such that for all $\mathbb{Q} \in \mathfrak{Q}$ there is a $G \in \mathcal{N}_n$ with

$$d_{f_{\mathbb{Q}}}(G, G^*_{\mathbb{Q}}) \leq c_2 \tau_n^{-\kappa}.$$

(iv) There exist constants $c_3, \rho > 0$ such that

$$\log\left(|\mathcal{N}_n|\right) \leq c_3 n^{\frac{\rho}{\rho+2\kappa-1}}.$$

Then for all $p \geq 1$ we have

$$\limsup_{n\to\infty} \sup_{\mathbb{Q}\in\mathfrak{Q}} \tilde{\tau}_n^{\kappa p}\, \mathbb{E}\big[d^p_{f_{\mathbb{Q}}}(\widehat{G}_n, G^*_{\mathbb{Q}})\big] < \infty,$$
$$\limsup_{n\to\infty} \sup_{\mathbb{Q}\in\mathfrak{Q}} \tilde{\tau}_n^{p}\, \mathbb{E}\big[d^p_{\Delta}(\widehat{G}_n, G^*_{\mathbb{Q}})\big] < \infty,$$

where

$$\tilde{\tau}_n := \min\{\tau_n, n^{\frac{1}{\rho+2\kappa-1}}\}$$

for all $n \in \mathbb{N}$.

Condition (i) is needed for all terms to be well defined. Condition (iii) states that the set in question must be well approximated by elements of $\mathcal{N}_n$. A sufficient assumption is that $\mathcal{N}_n$ is an ϵ-net of $\{G^*_{\mathbb{Q}} \mid \mathbb{Q} \in \mathfrak{Q}\}$, where $\epsilon := c_1\tau_n^{-\kappa}$. Together with (iv), this indirectly bounds the complexity of $\mathfrak{Q}$. If the class of sets $\{G^*_{\mathbb{Q}} \mid \mathbb{Q} \in \mathfrak{Q}\}$ is to large, one will not be able to find sets $\mathcal{N}_n$ that satisfy (iii) and (iv) at the same time. It is clear that the best rates are achieved with $\tau_n = n^{\frac{1}{\rho+2\kappa-1}}$. We do not use the same sequences in conditions (iii) and (iv) since one can prove non-optimal convergence rates using this version of the proposition.
The second condition is the noise condition described in the introduction. Note that following [126], for $\kappa > 1$ condition (ii) holds if

$$\mathbb{P}\Big(\big|2f_{\mathbb{Q}}(X) - 1\big| \leq t\Big) \leq ct^{\frac{1}{\kappa-1}}$$

for all $t > 0$ and some $c > 0$. Roughly speaking, this forces the mass to decay at a certain rate when one approaches the boundary of $G^*_{\mathbb{Q}}$. Note that we use (ii) instead of this assumption since it is slightly more general and includes the case $\kappa = 1$. Additionally, it appears more naturally in the proofs. Observing the alternative assumption, $\kappa = 1$ corresponds to the case where there is no mass close to the boundary of $G^*_{\mathbb{Q}}$, meaning that $f_{\mathbb{Q}}$ does not take on values close to $\frac{1}{2}$.

Using Proposition 3.2.1 one can achieve rates approaching n^{-1} for small κ, ρ i.e. if there is not much mass in the region around $f_{\mathbb{Q}}(x) = \frac{1}{2}$ and the complexity of $\mathcal{N}_n$, and consequently $\mathfrak{Q}$, is moderate. In contrast, the following proposition provides convergence rates if condition (ii) is not satisfied.

Proposition 3.2.2. *Let $\tau_n > 0$ be a monotonically increasing sequence. Let $\mathfrak{Q}$ be a class of potential joint distributions $\mathbb{Q}$ of (X, Y) and $\mathcal{N}_n$ be a collection of subsets of $\mathbb{R}^d$ for all $n \in \mathbb{N}$ such that the following conditions hold.*

(i) For all $\mathbb{Q} \in \mathfrak{Q}$ all sets in $\bigcup_{n\in\mathbb{N}} \mathcal{N}_n$ and $G^*_{\mathbb{Q}}$ are $\mathbb{Q}$-measurable.

Additionally, we assume that there is a constant $N_0 \in \mathbb{N}$ such that for all $n \geq N_0$ the following holds.

(ii) There is a constant $c_2 > 0$ such that for all $n \in \mathbb{N}$ and $\mathbb{Q} \in \mathfrak{Q}$ there is a $G \in \mathcal{N}_n$ with

$$d_{f_{\mathbb{Q}}}(G, G^*_{\mathbb{Q}}) \leq c_2 \tau_n^{-1}.$$

(iii) There exist $c_3, \rho > 0$ such that

$$\log\left(|\mathcal{N}_n|\right) \leq c_3 n^{\frac{\rho}{\rho+2}}.$$

Then for all $p \geq 1$ we have

$$\lim_{n\to\infty} \sup \sup_{\mathbb{Q}\in\mathfrak{Q}} \tilde{\tau}_n^p \, \mathbb{E}\left[d^p_{f_{\mathbb{Q}}}(\widehat{G}_n, G^*_{\mathbb{Q}})\right] < \infty,$$

where

$$\tilde{\tau}_n := \min\{\tau_n, n^{\frac{\rho}{\rho+2}}\}$$

for all $n \in \mathbb{N}$.

Note that the requirement in conditon (ii) of Proposition 3.2.2 corresponds to requirement (iii) of 3.2.1 with $\kappa = 1$. However, for $p = 1$ the best rate achievable is of order $n^{-\frac{1}{\rho+2}}$, which is always slower $n^{-\frac{1}{2}}$. Proposition 3.2.2 provides rates in absence of condition (ii) of Proposition 3.2.1. We do not claim optimality for these rates.

3.3 Convergence Rates for Neural Networks

We begin by shortly introducing neural networks. The idea is to use Proposition 3.2.1 to obtain optimal convergence rates up to a log factor. Neural networks are used to define a suitable class of sets $\mathcal{N}_n$ for every $n \in \mathcal{N}$.

3.3.1 Definitions Regarding Neural Networks

In this article, we are interested in the case where for $i = 1, \dots, L$ the activation function in the i-th layer is the rectifier linear unit (ReLU)

$$\sigma_i(x) := \max\{x, 0\}.$$

Additionally, if not further specified, we consider a compact domain $D := [0,1]^d$ and a one dimensional output $z_{L+1} = 1$.

As a first step, we wish to introduce a suitable finite class of sets parameterized by neural networks and count the number of elements. We define these sets as $R(\Phi)^{-1}(1)$ where Φ is a realization of a neural network. Equivalently, we could have considered neural networks with a binary step function in the output layer or find a neural network $\tilde{\Phi}$ and define the approximating set $R(\tilde{\Phi})^{-1}((0.5, 1])$, which is closer to the idea that the realization of the neural network represents some sort of probability. Since this is not the idea of our approximation results, we stick to the version above. In order to obtain a finite class, we need to reduce the number of considered elements of $\mathcal{N}_{L,z,\sigma}$ while maintaining reasonable approximating capabilities. A typical approach in the theoretical literature is to use a sparsity constraint. For $s > 1$ we therefore only consider realizations of neural nets which have at most s nonzero weights. If s is the total number of nonzero weights, we say that the network has sparsity s. Additionally, we assume all weights to be elements of the set

$$\mathcal{W}_c := \left\{ k2^{-c} \;\middle|\; c \in \mathbb{N},\ k \in \{-2^c, -2^c+1, \dots, 2^c-1, 2^c\} \right\}.$$

Thus, we only consider weights $|w| \leq 1$. Concluding, we use the following notation to describe the collection of sets we are interested in.

Definition 3.3.1. *Let $L_0, c \in \mathbb{N}$ and $s_0 > 1$ be fixed. Denote by $\tilde{\mathcal{N}}_{L_0,s_0,c}$ the set of realizations of neural networks with d dimensional input, one dimensional output, at most L_0 layers, ReLU activation functions and sparsity at most s_0, where all weights are elements of $\mathcal{W}_c$. The class of corresponding sets given by neural networks is then*

$$\mathcal{N}_{L_0,s_0,c} := \left\{ R(\Phi)^{-1}(1) \subseteq [0,1]^d, \;\middle|\; \Phi \in \tilde{\mathcal{N}}_{L_0,s_0,c} \right\}.$$

Note that the requirements from Definition 3.3.1 allow for realizations of neural networks with arbitrary hidden layer dimensions $(z_1, \dots, z_L) \in \mathbb{N}^L$. However, it is easy to see that every element of $\tilde{\mathcal{N}}_{L_0,s_0,c}$ is a realization of a neural net which satisfies the properties described in the Definition and $z_i \leq s_0$ for all $i \in \{1, \dots, L_0\}$. Using this, we receive an upper bound on the number of elements of $\mathcal{N}_{L_0,s_0,c}$ by counting the number of corresponding neural networks. Thus, the following bound is independent of the choice of activation functions σ.

Lemma 3.3.2. *For $s_0 > 1$ and $L_0, c \in \mathbb{N}$ let $\mathcal{N}_{L_0,s_0,c}$ be the class of sets introduced in Definition*

3.3.1. We have an upper bound on the number of elements given by

$$\left|\mathcal{N}_{L_0,s_0,c}\right| \leq \left((ds_0 + \min\{s_0, L_0\}(s_0+1)^2)2^{c+2}\right)^{s_0}.$$

Proof. First of all, if $s_0 \leq L_0$, clearly only the last s_0 layers have influence on the realization of a neural network. Thus, an upper bound is given by counting the number of neural nets with at most $\min\{s_0, L_0\}$ layers, at most sparsity s_0, weights in $\mathcal{W}_c$ and $m_i \leq s_0$ for all $i \in \{1, \dots, L\}$. Each weight can take on $|\mathcal{W}_c| = 2^{c+1} + 1$ different values. The total number of weights can be bounded by

$$\begin{aligned} z_0 z_1 + \sum_{i=1}^{\min\{s_0,L\}} (z_i+1) z_{i+1} &\leq ds_0 + \sum_{i=1}^{\min\{s_0,L_0\}} (s_0+1)s_0 \\ &\leq ds_0 + \min\{s_0, L_0\}(s_0+1)^2 \\ &=: V. \end{aligned}$$

Note that if $s_0 \leq L_0$, the input dimension does not influence the outcome. Therefore, there are at most

$$\binom{V}{s_0} \leq V^{s_0}$$

possible combinations to pick s_0 (possibly) nonzero weights. Thus

$$\left|\mathcal{N}_{L_0,s_0,c}\right| \leq V^{s_0}\left(2^{c+1}+1\right)^{s_0} \leq \left(V 2^{c+2}\right)^{s_0}.$$

□

3.3.2 Conditions on the Bayes-Rule

In order to define the set of probability distributions we consider for approximation, we restrict the possible bayes rules. We then add a smoothness condition to the function $f_{\mathbb{Q}}$ near the boundary of the respective bayes rule. Intuitively, the boundary should satisfy some kind of smoothness condition so that it can be approximated by neural networks. Additionally, the set must be discretizable in some sense. When using $\mathcal{F} = \mathcal{F}_{\beta,B,d-1}$ the class of sets we use is similar to a class defined in [104]. Note, that the class used here is larger. This version depends on a set $\mathcal{F}$ which represents a class of boundary functions. The idea is that we can obtain different convergence rates for different classes using the same procedure.

Definition 3.3.3. *Let $\beta \geq 0$, $B > 0$ and $d \in \mathbb{N}$ with $d \geq 2$. Additionally, let $r \in \mathbb{N}$, $\epsilon_1, \epsilon_2 > 0$, $\mathbb{Q}$ be a probability measure on $[0,1]^d \times \{0,1\}$ and $\mathcal{F}$ be a class of functions*

$$\gamma : [0,1]^{d-1} \to \mathbb{R}.$$

Define

$$\mathcal{I} := \{1,\dots,d\} \times \{-1,1\},$$
$$\mathcal{D} := \left\{ \prod_{i=1}^{d} [a_i, b_i] \;\middle|\; 0 \le a_i < b_i \le 1 \right\}.$$

and

$$\mathcal{H}_{\beta,B} := \left\{ g \in \mathcal{F}_{\beta,B,1} \;\middle|\; \forall \alpha < \beta : \partial^\alpha g(0) = 0 \right\}$$

if $\beta > 0$. Let $\mathcal{K}^{\mathcal{F}}_{\mathbb{Q},\beta,B,\epsilon_1,\epsilon_2,r,d}$ be the class of all subsets $H = H_1 \cup \dots \cup H_u \subseteq [0,1]^d$ for $u \in \{0,\dots,r\}$ such that for $\nu = 1,\dots,u$ there exist $(j_\nu, \iota_\nu) \in \mathcal{I}$, $\gamma_\nu \in \mathcal{F}$, $D_\nu = \prod_{i=1}^{d} [a_i^\nu, b_i^\nu] \in \mathcal{D}$ with the following properties.

1. *For all $\nu = 1,\dots,u$ we have*

$$H_\nu = D_\nu \cap \{x \in [0,1]^d \mid \iota_\nu x_{j_\nu} \le \gamma_\nu(x_{-j_\nu})\}.$$

2. *$\lambda(D_{\nu_1} \cap D_{\nu_2}) = 0$ for $\nu_1 \ne \nu_2$.*

3. *If $\beta > 0$, the following holds. For $\nu = 1,\dots,u$ and all $x \in D_\nu \cap \partial H$ there exists $g_{\nu,x} \in \mathcal{H}_{\beta,B}$ such that for $y_{j_\nu} \in [\max\{0, x_{j_\nu} - \epsilon_1\}, \min\{1, x_{j_\nu} + \epsilon_1\}]$ we have*

$$|2f_{\mathbb{Q}}(y) - 1| \le g_{\nu,x}(x_{j_\nu} - y_{j_\nu}) \quad \text{for } x_{j_\nu} - \epsilon_1 \le y_{j_\nu} \le x_{j_\nu},$$
$$|2f_{\mathbb{Q}}(y) - 1| \le g_{\nu,x}(y_{j_\nu} - x_{j_\nu}) \quad \text{for } x_{j_\nu} \le y_{j_\nu} \le x_{j_\nu} + \epsilon_1,$$

   where $y = (x_1,\dots,x_{j_\nu - 1}, y_{j_\nu}, x_{j_\nu + 1},\dots,x_d)$.

4. *$b_{j_\nu}^\nu - a_{j_\nu}^\nu \ge \epsilon_2$ for all $\nu = 1,\dots,u$.*

Note that $g \in \mathcal{H}_{\beta,B}$ implies $g(0) = \partial^0 g(0) = 0$. The idea is to use sets defined by realizations of neural networks to approximate $G^*_{\mathbb{Q}} \in \mathcal{K}^{\mathcal{F}}_{\mathbb{Q},\beta,B,\epsilon_1,\epsilon_2,r,d}$ from Definition 3.3.3 for a suitable class $\mathcal{F}$ in order to apply Proposition 3.2.1. Figure 3.1 shows an example for an element of $\mathcal{K}^{\mathcal{F}}_{\mathbb{Q},\beta,B,\epsilon_1,\epsilon_2,12,2}$, where $\mathcal{F}$ is the set of piecewise linear functions. The definition contains an additional condition on the function $f_{\mathbb{Q}}$ close to the boundary of $G^*_{\mathbb{Q}}$. Following the intuition mentioned in [91], for $\beta > 0$ condition (ii) in Theorem 2.1 means that $f_{\mathbb{Q}}$ acts like x^β close to the boundary of $G^*_{\mathbb{Q}}$, where $\kappa = 1 + \beta$. More precisely, condition (ii) requires that $f_{\mathbb{Q}}$ does not increase slower than x^β. In order to prove the combination of conditions (iii) and (iv), we require that β is the correct rate, meaning that $f_{\mathbb{Q}}$ does not increase faster than x^β. In Section 3.4 we prove that this condition does not lower the complexity of the problem. Thus, the rates obtained by [91] are still optimal.

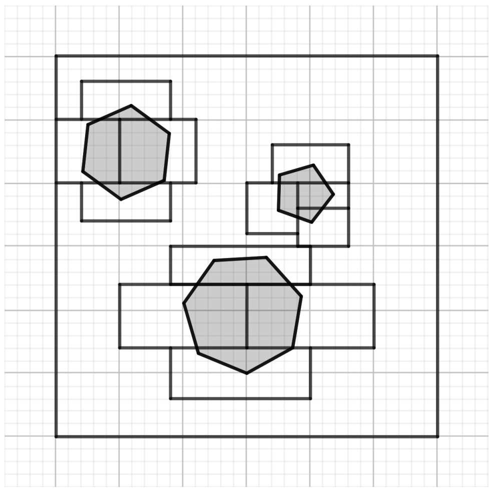


Figure 3.1: Example for an element of $\mathcal{K}^{\mathcal{F}}_{\mathbb{Q},\beta,B,\epsilon_1,\epsilon_2,12,2}$, where $\mathcal{F}$ is the set of piecewise linear functions. The grey objects represent the set. The small boxes represent a possible choice for $D_1, \ldots, D_{12}$.

3.3.3 Main Theorems

We begin by stating the central result of this article. We then use this result to show consistency results for more specific cases. The rates we obtain in Theorem 3.3.4 are optimal up to a log factor. In the following, all proofs of this section are given in Subsection 7.2.2.

Theorem 3.3.4. *Let $\beta \geq 0$, $B, \rho > 0$ and $d \in \mathbb{N}$ with $d \geq 2$. Let $\mathcal{F}$ be a set of functions*

$$\gamma : [0,1]^{d-1} \to \mathbb{R}$$

such that the following holds. There exist $\epsilon_0, C_1, C_2 > 0$ and $C_3, C_4 \in \mathbb{N}$ such that for any $\gamma \in \mathcal{F}$ and any $\epsilon \in (0, \epsilon_0)$ there is a neural network Φ with $L \leq L_0(\epsilon) := C_1 \lceil \log(\epsilon^{-1}) \rceil$ layers, sparsity $s \leq s_0(\epsilon) := C_2 \epsilon^{-\rho} \log(\epsilon^{-1})$ and weights in $\mathcal{W}_c$ with $c = c_0(\epsilon) := C_3 + C_4 \lceil \log(\epsilon^{-1}) \rceil$ such that

$$\|R(\Phi)(x) - \gamma\|_\infty \leq \epsilon.$$

Define $\kappa := 1 + \beta$ and let $\mathfrak{Q}$ be a class of potential joint distributions $\mathbb{Q}$ of (X, Y) such that the following conditions hold.

(a) There is a constant $M > 1$ such that for all $\mathbb{Q} \in \mathfrak{Q}$ the marginal distribution of $\mathbb{Q}_X$ has a Lebesgue density bounded by M.

(b) There are constants $r \in \mathbb{N}$ and $\epsilon_1, \epsilon_2 > 0$ such that for all $\mathbb{Q} \in \mathfrak{Q}$ the bayes rule satisfies $G^*_{\mathbb{Q}} \in \mathcal{K}^{\mathcal{F}}_{\mathbb{Q},\beta,B,\epsilon_1,\epsilon_2,r,d}$.

(c) We have

$$d_{f_{\mathbb{Q}}}(G, G^*_{\mathbb{Q}}) \geq c_1 d^{\kappa}_{\Delta}(G, G^*_{\mathbb{Q}})$$

for some constant $c_1 > 0$, all $G \in \mathcal{N}$ and all $\mathbb{Q} \in \mathfrak{Q}$, where

$$\mathcal{N} := \bigcup_{n \in \mathbb{N}} \mathcal{N}_{n,n,n}$$

is the class of sets corresponding to any neural network.

Let

$$\tau_n := \frac{n^{\frac{1}{2\kappa+\rho-1}}}{\log^{\frac{2}{\rho}}(n)}.$$

Then there exist constants $C_1', C_2' > 0$ and $C_3' \in \mathbb{N}$ such that for all $p \geq 1$ we have

$$\limsup_{n\to\infty} \sup_{\mathbb{Q}\in\mathfrak{Q}} \tau_n^{p\kappa}\, \mathbb{E}\big[d^p_{f_\mathbb{Q}}(\widehat{G}_n, G^*_\mathbb{Q})\big] < \infty,$$
$$\limsup_{n\to\infty} \sup_{\mathbb{Q}\in\mathfrak{Q}} \tau_n^{p}\, \mathbb{E}\big[d^p_{\Delta}(\widehat{G}_n, G^*_\mathbb{Q})\big] < \infty,$$

where

$$\widehat{G}_n := \arg\min_{G\in\mathcal{N}_n} R_n(G).$$

with $\mathcal{N}_n = \mathcal{N}_{C_1' L_0(\tau_n^{-1}), C_2' s_0(\tau_n^{-1}), C_3' c_0(\tau_n^{-1})}$.

3.3.4 Results for Regular Boundaries

We can now prove results for specific classes of sets $\mathcal{F}$ to obtain convergence results. A first important example is the class $\mathcal{F}_{\beta,B,d}$. The following Lemma is a consequence of Theorem 5 in [109].

Lemma 3.3.5. *Let $\beta, B > 0$ and $d \in \mathbb{N}$. Then there exist $\epsilon_0, c_1, c_2 > 0, c_3, c_4 \in \mathbb{N}$ such that the following holds. For any function $\gamma \in \mathcal{F}_{\beta,B,d}$ and any $\epsilon \in (0, \epsilon_0)$, there exists a neural network Φ with $L \leq L_0(\epsilon) := c_1\lceil\log(\epsilon^{-1})\rceil$ layers, sparsity $s \leq s_0(\epsilon) := c_2\epsilon^{-\frac{d}{\beta}}\log(\epsilon^{-1})$ and weights in $\mathcal{W}_c$ with $c = c_0(\epsilon) := c_3 + c_4\lceil\log(\epsilon^{-1})\rceil$ such that*

$$\|R(\Phi)(x) - \gamma\|_\infty \leq \epsilon.$$

Corollary 3.3.6. *Let $\beta_1 \geq 0$, $B_1, \beta_2, B_2 > 0$ and $d \in \mathbb{N}$ with $d \geq 2$. Let $\mathfrak{Q}$ be a class of potential joint distributions $\mathbb{Q}$ of (X, Y). Assume (a),(b),(c) from Theorem 3.3.4 hold with $\rho := \frac{d-1}{\beta_2}$, $\beta := \beta_1$, $B := B_1$ as well as $\mathcal{F} := \mathcal{F}_{\beta_2,B_2,d-1}$. Let*

$$\tau_n := \frac{n^{\frac{1}{2\kappa+\rho-1}}}{\log^{\frac{2}{\rho}}(n)}.$$

Then there exist constants $C_1', C_2' > 0$ and $C_3' \in \mathbb{N}$ such that for all $p \geq 1$ we have

$$\limsup_{n\to\infty} \sup_{\mathbb{Q}\in\mathfrak{Q}} \tau_n^{p\kappa}\, \mathbb{E}\big[d_{f_{\mathbb{Q}}}^p(\widehat{G}_n, G_{\mathbb{Q}}^*)\big] < \infty,$$
$$\limsup_{n\to\infty} \sup_{\mathbb{Q}\in\mathfrak{Q}} \tau_n^{p}\, \mathbb{E}\big[d_{\Delta}^p(\widehat{G}_n, G_{\mathbb{Q}}^*)\big] < \infty,$$

where

$$\widehat{G}_n := \arg\min_{G\in\mathcal{N}_n} R_n(G)$$

with $\mathcal{N}_n = \mathcal{N}_{C_1'L_0(\tau_n^{-1}), C_2's_0(\tau_n^{-1}), C_3'c_0(\tau_n^{-1})}$ and L_0, s_0, c_0 from Lemma 3.3.5.

Corollary 3.3.6 together with Theorem 3.4.1 from the next section prove optimal convergence rates, which was the main goal of this work.

Following up, note that the rates we receive from Corollary 3.3.6 are affected by the curse of dimensionality. Observe that the rates obtained by Theorem 3.3.4 are influenced by condition (c) on the one hand and the ability of neural networks to approximate sets in $\mathcal{F}$ on the other. The dependence on the dimension d in Corollary 3.3.6 comes from the latter. Thus, a natural approach to circumvent the curse of dimensionality is to approximate a smaller set $\mathcal{F}$. Intuitively, it is clear that without strong restrictions on the distribution we can only overcome the curse if the complexity of the boarders of the sets we approximate is small enough so that they themselves can overcome the curse. In the literature, many different sets are considered which infer useful approximation capabilities of neural networks. Here, we use a class of sets introduced by [109] which is close to class $\mathcal{F}_{\beta,B,d}$.

Definition 3.3.7. *Let $r \in \mathbb{N}$, $t \in \mathbb{N}^r$, $d \in \mathbb{N}^{r+1}$, $\beta \in \mathbb{R}^r$ and $B > 0$ with $t_i \leq d_i$, $\beta_i > 0$ for $i = 1, \dots, r$, $d_{r+1} = 1$. Define*

$$\begin{aligned}\mathcal{G}_{r,t,\beta,B,d} := \big\{\gamma = \gamma_r \circ \cdots \circ \gamma_1 \;\big|\; &\gamma_i = (\gamma_{ij} \circ \iota_{ij})_{j=1}^{d_{i+1}},\ \gamma_{ij} \in \mathcal{F}_{\beta_i,B,t_i},\ \iota_{ij} \in \mathcal{ID}_i,\\ &\gamma_{ij} : [0,1]^{t_i} \to [0,1] \text{ for } i = 1, \dots, r-1,\\ &\gamma_{r1} : [0,1]^{t_r} \to \mathbb{R}\big\},\end{aligned}$$

where

$$\mathcal{ID}_i = \big\{\iota : [0,1]^{d_i} \to [0,1]^{t_i} \;\big|\; \iota(x) = (x_{i_1}, \dots, x_{i_{t_i}}),\ i_j \in \{1, \dots, d_i\}\big\}.$$

Instead of requiring that γ_{ij} is supported on $[0,1]^{t_i}$, we could have used the condition that γ_{ij} is supported on $\prod_{k=1}^{t_i}[a_k, b_k]$ for some values $a_k, b_k \in \mathbb{R}$. However, this does not enlarge the class considerably. It can easily be seen that we can instead increase the bound B to find an even larger class. The idea for using the set $\mathcal{G}_{r,t,\beta,B,d}$ is that its complexity does not depend on the input dimension d_1, but only on the most difficult component to approximate. The complexity of the components depend on their effective dimension t_i and their implied smoothness. As described by [109], the correct smoothness parameter to consider is

$$\beta_i^* := \beta_i \prod_{k=i+1}^{r} \min\{\beta_k, 1\}.$$

Examples for sets that can profit from Definition 3.3.7 are additive models ($r = 1$, $t_1 = 1$), interaction models of order k ($r = 1$, $t_1 = k$), or multiplicative models (they are a subset of $\mathcal{G}_{r,t,\beta,B,d}$ when $r = \log_2(d) + 1$, $t_i = 2$ for all i). Next, our goal is to establish a convergence result when the set of boundary functions is $\mathcal{G}_{r,t,\beta,B,d}$. Similarly to the approach above, we first provide a lemma which provides approximation results using neural networks.

Lemma 3.3.8. *Let $r \in \mathbb{N}$, $t \in \mathbb{N}^r$, $d \in \mathbb{N}^{r+1}$, $\beta \in \mathbb{R}^r$ and $B > 0$ with $t_i \leq d_i$, $\beta_i > 0$ for $i = 1, \dots, r$, $d_{r+1} = 1$. Let $\mathcal{G}_{r,t,\beta,B,d}$ be defined as in Definition 3.3.7 and define*

$$\rho := \max_{i=1,\dots,r} \left(\frac{t_i}{\beta_i^*} \right).$$

Then there exist $\epsilon_0, c_1, c_2 > 0, c_3, c_4 \in \mathbb{N}$ such that the following holds. For any function $\gamma \in \mathcal{G}_{r,t,\beta,B,d}$ and any $\epsilon \in (0, \epsilon_0)$, there exists a neural network Φ with $L \leq L_0(\epsilon) := c_1 \lceil \log(\epsilon^{-1}) \rceil$ layers, sparsity $s \leq s_0(\epsilon) := c_2 \epsilon^{-\rho} \log(\epsilon^{-1})$ and weights in $\mathcal{W}_c$ with $c = c_0(\epsilon) := c_3 + c_4 \lceil \log(\epsilon^{-1}) \rceil$ such that

$$\|R(\Phi)(x) - \gamma\|_\infty \leq \epsilon.$$

The following corollary establishes the corresponding convergence result. Theorem 3.4.2 provides the lower bound in the case where

$$t_i \leq \min\{d_1, \dots, d_i\}.$$

Corollary 3.3.9. *Let $r_2 \in \mathbb{N}$, $t \in \mathbb{N}^{r_2}$, $d \in \mathbb{N}^{r_2+1}$, $\beta_1 \geq 0$, $\beta_2 \in \mathbb{R}^{r_2}$, and $B_1, B_2 > 0$ with $\beta_{2,i} > 0$ for $i = 1, \dots, r_2$, $d_{r_2+1} = 1$. Additionally, $t_1 < d_1$ and $t_i \leq d_i$ for $i \neq 1$. Define $d' \in \mathbb{N}^{r_2+1}$ with $d'_1 = d_1 - 1$, $d'_i = d_i$ for $i \neq 1$ and*

$$\rho := \max_{i=1,\dots,r_2} \frac{t_i}{\beta_{2,i}^*}$$

Define $\kappa := 1 + \beta_1$ and let $\mathfrak{Q}$ be a class of potential joint distributions $\mathbb{Q}$ of (X, Y) such that the following conditions hold.

(a) There is a constant $M > 1$ such that for all $\mathbb{Q} \in \mathfrak{Q}$ the marginal distribution of $\mathbb{Q}_X$ has a Lebesgue density bounded by M.

(b) There are constants $r_1 \in \mathbb{N}$ and $\epsilon_1, \epsilon_2 > 0$ such that for all $\mathbb{Q} \in \mathfrak{Q}$ the bayes rule satisfies $G_\mathbb{Q}^* \in \mathcal{K}^{\mathcal{F}}_{\mathbb{Q},\beta_1,B_1,\epsilon_1,\epsilon_2,r_1,d_1}$ with

$$\mathcal{F} := \mathcal{G}_{r_2,t,\beta_2,B_2,d'}.$$

(c) We have

$$d_{f_\mathbb{Q}}(G, G_\mathbb{Q}^*) \geq c_1 d_\Delta^\kappa(G, G_\mathbb{Q}^*)$$

for some constant $c_1 > 0$, all $G \in \mathcal{N}$ and all $\mathbb{Q} \in \mathfrak{Q}$, where

$$\mathcal{N} := \bigcup_{n \in \mathbb{N}} \mathcal{N}_{n,n,n}$$

is the class of sets corresponding to any neural network.

Let

$$\tau_n := \frac{n^{\frac{1}{2\kappa+\rho-1}}}{\log^{\frac{2}{\rho}}(n)}.$$

Then there exist constants $C_1', C_2' > 0$ and $C_3' \in \mathbb{N}$ such that for all $p \geq 1$ we have

$$\lim_{n\to\infty} \sup \sup_{\mathbb{Q}\in\mathfrak{Q}} \tau_n^{p\kappa} \, \mathbb{E}\big[d^p_{f_{\mathbb{Q}}}(\widehat{G}_n, G^*_{\mathbb{Q}})\big] < \infty,$$
$$\lim_{n\to\infty} \sup \sup_{\mathbb{Q}\in\mathfrak{Q}} \tau_n^{p} \, \mathbb{E}\big[d^p_{\Delta}(\widehat{G}_n, G^*_{\mathbb{Q}})\big] < \infty,$$

where

$$\widehat{G}_n := \arg\min_{G\in\mathcal{N}_n} R_n(G).$$

with $\mathcal{N}_n = \mathcal{N}_{C_1' L_0(\tau_n^{-1}), C_2' s_0(\tau_n^{-1}), C_3' c_0(\tau_n^{-1})}$.

Note that Corollary 3.3.9 is a generalisation of Corollary 3.3.6. The rate now depends on ρ which in turn depends on $t_1, \dots, t_{r_2}$ instead of the input dimension d_1. Clearly, the effective dimensions t_i can be much smaller then the input dimension d_1, for example, when the boundary functions come from an additive function.

3.4 Lower Bound

We now establish lower bounds on the convergence rates from corollaries 3.3.6 and 3.3.9. Note that the lower bounds also prove that the rates obtained in Theorem 3.3.4 can not be improved up to a log-factor. Since Corollary 3.3.9 is a generalisation of 3.3.6, we only have to prove a lower bound for the setting given in the former. For clarity, we provide both statements. The proofs of this section can be found in Subsection 7.2.3.

Theorem 3.4.1. *Let $\beta_1 \geq 0$, $B_1, \beta_2, B_2, \rho > 0$ and $d \in \mathbb{N}$ with $d \geq 2$. Let $\mathfrak{Q}$ be the class of all potential joint distributions $\mathbb{Q}$ of (X, Y) such that (a),(b) from Theorem 3.3.4 hold with $\rho := \frac{d-1}{\beta_2}$, $\kappa := 1 + \beta_1$, $\beta := \beta_1$, $B := B_1$, $\mathcal{F} := \mathcal{F}_{\beta_2, B_2, d-1}$ and some $M > 1$, $r \in \mathbb{N}$, $\epsilon_1, \epsilon_2 > 0$. Let (c) hold with $c_1 > 0$ large enough and set*

$$\tau_n := n^{\frac{1}{2\kappa+\rho-1}}.$$

Then

$$\liminf_{n\to\infty} \inf_{G_n \in \mathfrak{G}} \sup_{\mathbb{Q}\in\mathfrak{Q}} \tau_n^p \mathbb{E}\left[d_\Delta^p(G_n, G_\mathbb{Q}^*)\right] > 0,$$
$$\liminf_{n\to\infty} \inf_{G_n \in \mathfrak{G}} \sup_{\mathbb{Q}\in\mathfrak{Q}} \tau_n^{p\kappa} \mathbb{E}\left[d_{f_\mathbb{Q}}^p(G_n, G_\mathbb{Q}^*)\right] > 0$$

for every $p \geq 0$, where $\mathfrak{G}$ contains all estimators depending on the data $(X_1, Y_1), \ldots, (X_n, Y_n)$.

Intuitively, B_1 bounds the factor of the x^{β_2}-term of $f_\mathbb{Q}$ close to the boundary from above. On the other hand, c_1 bounds this term from below. Thus, not every combination of $B_1, c_1 > 0$ is possible. We prove Theorem 3.4.1 for large $c_1 > 0$. We do not provide the exact ratio of B and c_1 required since it is not important for the statement. Lastly, the lower bound corresponding to Corollary 3.3.9 is given.

Theorem 3.4.2. *Let $r_2 \in \mathbb{N}$, $t \in \mathbb{N}^{r_2}$, $d \in \mathbb{N}^{r_2+1}$, $\beta_1 \geq 0$, $\beta_2 \in \mathbb{R}^{r_2}$, and $B_1, B_2 > 0$ with $\beta_{2,i} > 0$ for $i = 1, \ldots, r_2$, $d_{r_2+1} = 1$. Additionally, $t_1 < d_1$ and $t_i \leq \min\{d_1, \ldots, d_i\}$ for $i \neq 1$. Let*

$$\rho := \max_{i=1,\ldots,r_2} \frac{t_i}{\beta_{2,i}^*}$$

Define $\kappa := 1 + \beta_1$ and let $\mathfrak{Q}$ be the class of all potential joint distributions $\mathbb{Q}$ of (X, Y) such that (a),(b) from Corollary 3.3.9 hold for some $M > 1$, $r \in \mathbb{N}$, $\epsilon_1, \epsilon_2 > 0$. Let (c) hold with $c_1 > 0$ large enough and set

$$\tau_n := n^{\frac{1}{2\kappa+\rho-1}}.$$

Then

$$\liminf_{n\to\infty} \inf_{G_n \in \mathfrak{G}} \sup_{\mathbb{Q}\in\mathfrak{Q}} \tau_n^p \mathbb{E}\left[d_\Delta^p(G_n, G_\mathbb{Q}^*)\right] > 0,$$
$$\liminf_{n\to\infty} \inf_{G_n \in \mathfrak{G}} \sup_{\mathbb{Q}\in\mathfrak{Q}} \tau_n^{p\kappa} \mathbb{E}\left[d_{f_\mathbb{Q}}^p(G_n, G_\mathbb{Q}^*)\right] > 0$$

for every $p \geq 0$, where $\mathfrak{G}$ contains all estimators depending on the data $(X_1, Y_1), \ldots, (X_n, Y_n)$.

3.5 Concluding Remarks

We establish optimal convergence rates up to a log-factor in a classification setting under the (3.1.1) using neural networks. Theorem 3.3.4 can be applied for many different boundary functions. The complexity of the class of boundary functions $\mathcal{F}$ is one of the main driving factors of the convergence rate. In particular, many approaches which circumvent the curse of dimensionality in a regression setting can be used to circumvent the curse in this classification setting.
Note that the ideas provided here are of a theoretical nature. While sparsity constraints are considered thoroughly in the theoretical literature, they are not widely used in practice. Additionally, we did not discuss the minimization required for the calculation of $\widehat{G}_n$. This is a very interesting but complicated topic which is not in the scope of this article. Observe that the class

of neural networks used in Theorem 3.3.4 depends on κ as well as ρ. We believe that one can extend the results developed here by either having adaptive classes of neural networks or a class independent of κ and ρ in a similar manner to [126]. One obstacle to overcome is the fact that the conditions on the probability distribution $\mathfrak{Q}$ required are not strictly weaker for larger κ and ρ.

Lastly, while our goal is to prove results considering neural networks, it also contains new insights on the noise condition (3.1.1). In order to establish optimal convergence, an additional condition in order to show approximation results of neural networks with respect to the metric $d_{f_{\mathbb{Q}}}$ is necessary. Intuitively, the reverse inequality is required for certain sets. Note that requiring the reverse inequality is an overly restrictive assumption which holds for almost no classes of possible distributions $\mathfrak{Q}$ for $\kappa \neq 1$. This proved to be a major challenge and is solved by (3.) in Definition 3.3.3. While this condition is always also satisfied for larger but not lower β (and thus κ), the reverse is true in condition (3.1.1). Thus, together the requirement is that κ is the "correct rate". Note that condition (3.) still allows for highly non-continuous $f_{\mathbb{Q}}$ close to the boundary of $G^*_{\mathbb{Q}}$. This is essential, since considering only smooth $f_{\mathbb{Q}}$ close to the boundary leads to different convergence rates as shown in Theorem 2 of [71].

4 Random Planted Forest

This chapter follows the paper [54]. We included some slight modifications in order to embed it into this thesis. Additionally, it includes joint research with Lukas Burk, Munir Hiabu, Enno Mammen and Marvin Wright on a generalisation of random planted forest which will be published in the future.

We introduce a novel interpretable, tree based algorithm for prediction in a regression setting in which each tree in a classical random forest is replaced by a family of planted trees that grow simultaneously. The motivation for our algorithm is to estimate the unknown regression function from a functional decomposition perspective, where each tree corresponds to a function within that decomposition. The maximal order of approximation in the decomposition can be specified or left unlimited. If a first order approximation is chosen, the result is an additive model. In the other extreme case, if the order of approximation is not limited, the resulting model places no restrictions on the form of the regression function. In a simulation study we find encouraging prediction and visualisation properties of our random planted forest method. We also develop theory for an idealised version of random planted forests in cases where the maximal order of approximation is low. We show that if the order is smaller than three, the idealised version achieves asymptotically optimal convergence rates up to a logarithmic factor.

4.1 Introduction

In many ways, machine learning has been very disruptive in the last two decades. The class of neural networks has shown unprecedented and previously unthinkable predictive accuracy in fields such as image recognition [78, 106], speech recognition [57], and natural language processing [27]. Deep neural networks are strong in applications where huge amounts of data can be assembled and many variables interact with each other. A second disruptive class of machine learning algorithms are decision tree ensembles. The gradient boosting machine [37] in particular excels in many applications. It often is the best performing algorithm for tabular data [44] and as such it is praised by many practitioners. Among the 29 challenge-winning solutions posted on Kaggle during 2015, 17 used xgboost – a variant of a gradient boosting machine [20]. The second most popular method, deep neural networks, was used in 11 solutions.
These two classes of machine learning algorithms are in contrast to classical statistical models that assume an explicit structure such as a linear model [97] or an additive model [15, 38]. Traditionally, if the data does not fit into the required structure, the analyst tries to transform the variables at hand to achieve better fits – for example by using a log-transformation. Estimators

received from the classical statistical models are highly accurate if the model is correctly specified. However, they perform poorly if the data deviates strongly from the structure assumed by the model.[1]

Our aim is to combine the best of the two worlds. We propose a new tree based algorithm we call random planted forests (rpf). It starts with a simple structure and becomes increasingly flexible in a data-driven way as the algorithm unfolds.

We believe that one important factor of the success of gradient boosting is that it indirectly limits the order of interaction between predictors by limiting the depth of the trees. Thus, higher order interaction terms are ignored which usually require a large amount of data to be predicted accurately. This heuristic is also supported by [122] who find that random forests perform poorly in additive models and by their proposal to look for modifications of random forests that adapt to structures in the data. Motivated by this paper, in [123] a modification of random forest was proposed where in each iteration the tree is replaced by a finite fixed number of trees that grow in parallel. We consider a regression problem and assume that the regression function can be well approximated by lower order terms in a functional decomposition

$$m(x) = \sum_{t\subseteq\{1,\ldots,d\}} m_t(x_t) = m_\emptyset + \sum_{t\in\{1,\ldots,d\}} m_{\{t\}}(x_t) + \sum_{t_1<t_2} m_{\{t_1,t_2\}}(x_{t_1}, x_{t_2}) + \cdots . \tag{4.1.1}$$

The rpf algorithm approximates this decomposition by hierarchically estimating the summands from lower to higher order interaction terms. Figure 4.1 provides a short illustration. The interaction terms included can be selected in advance. For example, one can fix the maximum order of approximation in the decomposition. If all interaction terms of all orders are included one has the full model where no restrictions are applied to the regression function. Uniqueness of the components is ensured via identification constraints, see Subsection 6.1.2. In contrast to the problem considered by [2], the constraint itself is of secondary importance. The reason is that we do not aim to approximate a multivariate estimator by (4.1.1). The rpf estimator is already in the form of (4.1.1). Hence, the constraint does not have an effect on the quality of the estimator.

In the simulation study presented in Section 4.4, we find that even if the order of approximation in the rpf algorithm is not bounded the results are promising. However, bounding the order of interaction does increase the performance slightly if the true model satisfies the same constraint. An additional advantage of choosing a first or second order approximation is that the prediction can be visualised easily. Figure 4.2, Figure 4.3, and Figure 4.4 show plots that visualise additive and two dimensional components of a high dimensional model, respectively. The models we compare to where chosen since they are the strongest competitors which are easily visualisable regarding the simulation results in Section 4.4. Our rpf algorithm differs from classical random forests in several respects:

- Each tree in a random forest is replaced by a tree family. Each tree in a family corresponds

[1]There are also fully nonparametric estimators which do not assume a specific structure, e.g., local polynomial estimators. However these are exposed to the curse of dimensionality, i.e. exponentially deteriorating estimation performance with growing number of variables.

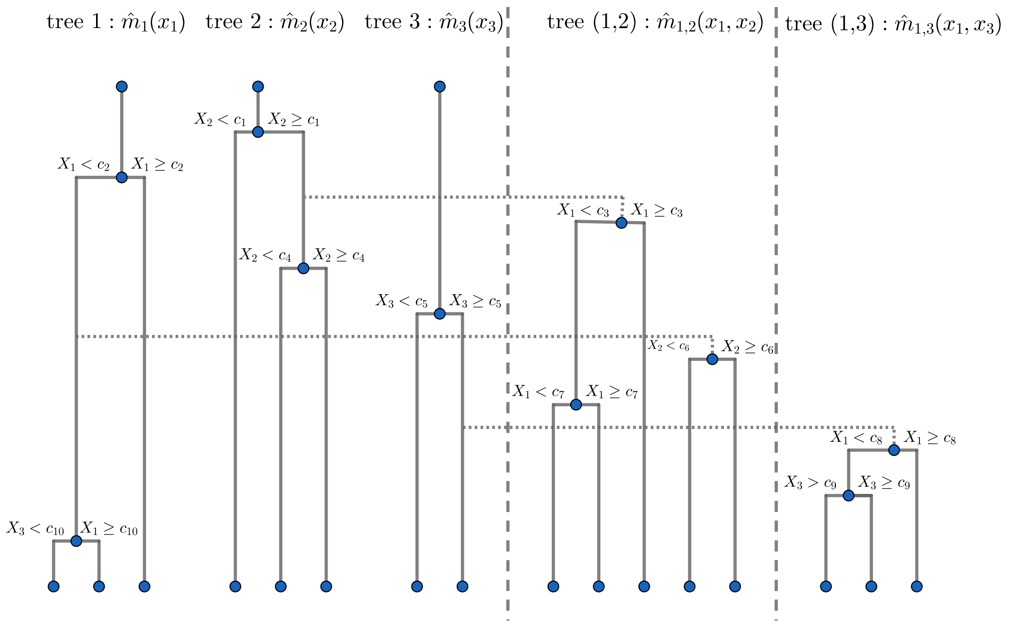


Figure 4.1: Illustration of a family of planted trees. Higher order trees are descendants of lower order trees. Trees grow simultaneously and the height of the edges indicate the order in which splits occurr.

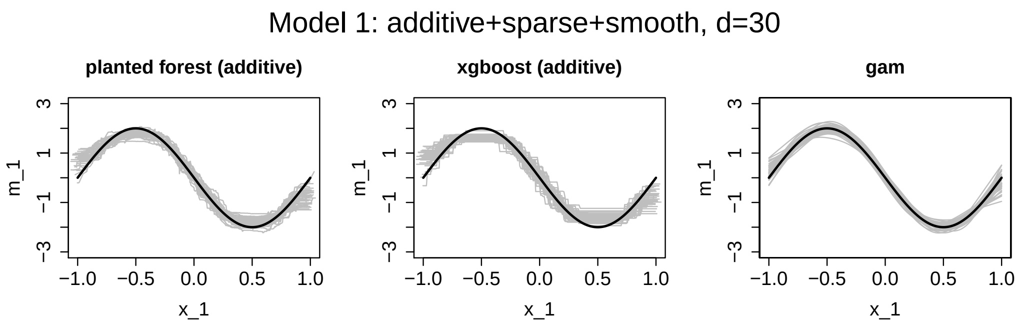


Figure 4.2: Estimates from 40 simulations for m_1 of Model 1: $m(x_1, \ldots, x_{30}) = m_1(x_1) + m_2(x_2)$ with $m_k(x_k) = 2(-1)^k \sin(\pi x_k)$. The true function is visualised as a black solid line. Sample size is $n = 500$. Predictors have an approximate pairwise correlation of 0.3 and the noise has variance 1. For rpf and xgboost, parameters are picked from a grid search. The gam curves have data-driven parameters.

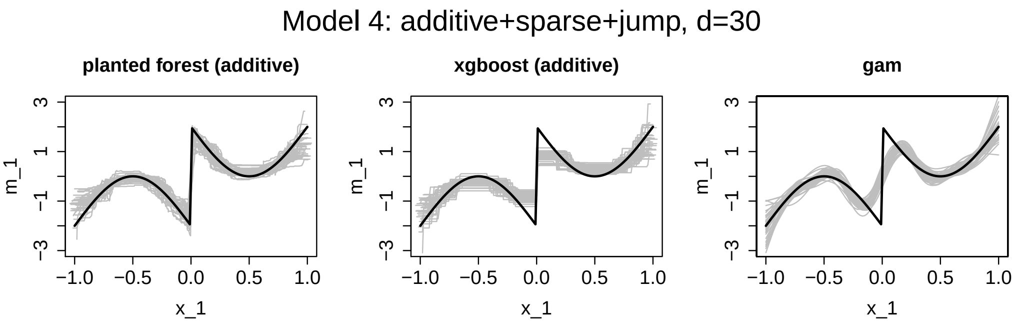


Figure 4.3: Estimates from 40 simulations for m_1 of Model 4: $m(x_1, \dots, x_{30}) = m_1(x_1) + m(x_2)$ with $m_k(x_k) = (2(-1)^k \sin(\pi x_k) - 2)\mathbb{1}(x \geq 0) + (-2\sin(\pi x_k) + 2)\mathbb{1}(x < 0)$. The true function is visualised as a black solid line. Sample size is $n = 500$. Predictors have an approximate pairwise correlation of 0.3 and the noise has variance 1. For rpf and xgboost parameters are picked from a grid search. The gam curves have data-driven parameters.

to one term in the decomposition. The trees grow simultaneously and the global fit is given by the sum of the tree estimates.

- The algorithm for a tree family starts by growing main effect trees. Interaction trees of order two are generated as descendants of main effect trees. More generally, interaction trees of higher order come from interaction trees of one degree less.
- As in classical random forests, splits are only allowed in a random subset of variables. This is done to reduce dependence between tree families with the aim of decreasing the variance of the forest estimator. Constructing random subsets is more involved in the rpf algorithm because different trees in one family have different dimensions. We call the parameter controlling the size of these random subsets `t_try`.
- Split points are selected from a finite set of random points. Not considering all possible points for splitting reduces computation time. Choosing random points further reduces the dependence between tree families. Additionally, bias is reduced in the case of smooth components. This can be deducted from our simulation study as well as theory developed in Section 4.5. We call the parameter controlling the number of possible split points `split_try`.

The idea of using splits from a finite set of random points is very similar to the method used in extremely random forests [41].
A tree family in the rpf algorithm can be thought of as one traditional decision tree in which a leaf is not removed from the algorithm when split in some cases. This happens when a leaf is split with respect to a component which was not used to construct the leaf so far. Additionally, each leaf may not be constructed by using more than r covariates. A simple heuristic why it may be beneficial not to remove a leaf from a tree when splitting is the following. Consider the regression problem $Y_i = m(X_i) + \epsilon_i$ with $i = 1, \dots, n$ and $m(x) = \sum_{j=1}^{d} \mathbb{1}(x_j \leq 0.5)$ for large

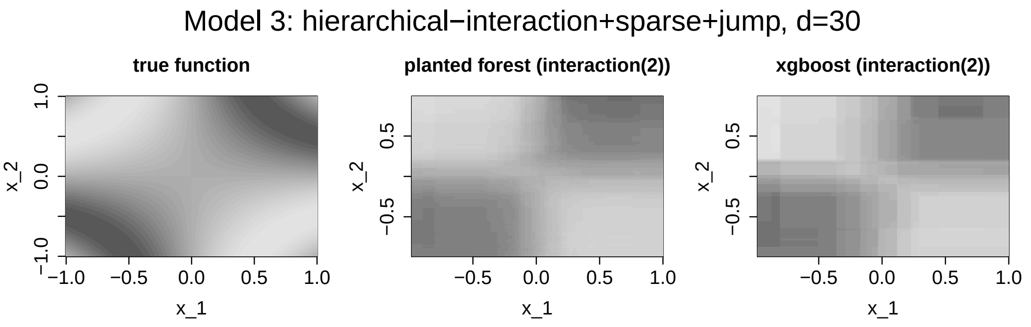


Figure 4.4: Heatmap from the median performing run for each method, measured via mean squared error, out of 40 simulations. Predictions are for $m_{1,2}$ of Model 3: $m(x_1,\dots,x_{30}) = \sum_{k=1}^{3} m_k(x) + \sum_{1\le k<j\le 3} m_{k,j}(x_k,x_j)$ with $m_{k,j}(x_k,x_j) = 2(-1)^k \sin(\pi x_k x_j)$, $m_k(x_k) = (-1)^k 2\sin(\pi x_1)$. Sample size is $n = 500$. The predictors have an approximate pairwise correlation of 0.3 and the noise has variance 1. Parameters are picked from a grid search.

d. If we never remove the original leaf, one can approximate m by splitting the original leaf once with respect to each covariate j. Thus, for each direction, n data points are considered for finding the optimal split value. Additionally we end up with on average $n/2$ data points in each leaf. In the original random forests algorithm, in order to find a similar function, one would have to grow a tree with depth d, where each leaf is constructed by splitting once with respect to each covariate. This implies that we end up with 2^d leaves, which on average contain $n/(2^d)$ data points. Thus, the estimation should be much worse both because of less precise split points and less accurate fitted values.

If we only allow for a maximum interaction of two or less, the rpf algorithm provides an estimator which is easily interpretable. In recent years, there have been other algorithms with similar benefits: The rpf method differs from the tree based explainable boosting machine [16, 80, 81, 82] and the neural network based neural additive model [1] recently introduced in several ways. Similar to rpf, these methods aim to approximate the regression function via the functional decomposition (4.1.1). However, they rather resemble classical statistical methods described earlier. After specifying a fixed structure, the explainable boosting machine assumes that every component is relevant and all components are fitted via a backfitting algorithm [14]. A similar principle using backpropagation instead of backfitting is applied in the neural additive model. In contrast, the rpf algorithm may ignore certain components completely. Additionally, interaction terms do not need to be specified beforehand. Hence explainable boosting machine and the neural additive model do not share two key strengths that rpf has: a) automatic interaction detection up to the selected order, b) strong performance in sparse settings.

Some model-based boosting algorithms [59] have similar properties to the ones given above. In particular, they also suffer from the fact that they do not have automatic interaction detection, especially if a large number of covariates are present. An algorithm not suffering from this problem is multivariate adaptive regression splines [36]. However, it has a continuous output as composition of hinge functions. Thus, it does not adapt well when single jumps are present.

Furthermore, as seen in our simulation study in Section 4.4, multivariate adaptive regression splines are less accurate when many covariates are active. A related algorithm which copes well with interaction terms is bayesian additive regression trees [25]. As seen in our simulation study in Section 4.4, the algorithm is competitive in all cases considered. However, bayesian additive random trees provides an estimator which does not follow a functional decomposition. Thus, it does not allow for interpretability as rpf does. Additionally, the MCMC method required is computationally expensive.

In Section 4.5, we develop theory for an idealised version of rpf. From a theoretical point of view, the most comparable algorithm to rpf is random forests. A theoretical study of Breiman's original version of random forests [13] is rather complex due to the double use of data, once in a CART splitting criterion and once in the final fits. [110] provide a consistency proof for Breiman's algorithm while assuming an underlying additive model. They also show that the forest adapts to sparsity patterns. In more recent papers, theory has been developed for subsampled random forests. By linking to theory of infinite order U-statistics, asymptotic unbiasedness and asymptotic normality has been established for these modifications of random forests, see [94, 95, 103, 127]. In our analysis of rpf, we assume that the splitting values do not depend on the response variable. In this respect, we follow a strand of literature on random forests, see e.g. [6, 7]. Thus, we circumvent theoretical problems caused by the double use of data as in Breiman's random forest. We leave it to future research to study the extent to which theory introduced here carries over to random forests with data-dependent splitting rules, in particular for the case of random forests based on subsampling or sample-splitting. For a review of theoretical aspects of random forests, we also refer to [8].

In Section 4.2, we introduce and explain the rpf algorithm in detail. In Section 4.3, the influence of various parameters is discussed. Additionally, we explain the intuition behind choices made in the construction of the algorithm. In Section 4.4, we perform a simulation study and compare our rpf algorithm to various other methods. Section 4.5 is a theoretical section. Among others, we show that an algorithm related to the rpf algorithm achieves asymptotically optimal convergence rates up to a logarithmic factor when the maximum order of approximation $r \leq 2$. Lastly, in Section 4.6, we introduce a generalisation of the algorithm, which is part of an ongoing research project. Among others, the generalisation can be used in classification settings.

4.2 The Algorithm

In the following, we describe the rpf algorithm. We are handed data $(Y_i, X_{i,1}, \dots, X_{i,d})_{i=1}^n$ consisting of i.i.d. observations with $Y_i, X_{i,k} \in \mathbb{R}$. We consider the regression problem

$$E[Y_i|X_i = x] = m(x),$$

with the goal of estimating m. We assume that m can be approximated well by a functional decomposition [18, 60, 117]

$$m(x) = \sum_{t \in T_r} m_t(x_t), \tag{4.2.1}$$

with a specified maximum order of approximation $r = 1, \ldots, d$, where

$$T_r := \{t \subset \{1, \ldots, d\} \mid |t| \leq r\}.$$

Note that $m_\emptyset$ is a constant. Rpf is based on tree-like structures, which we refer to as trees for simplicity. A tree consists of a set of dimensions $t \in T_r$, a finite class of subsets $\{\mathcal{I}_{t,1}, \ldots, \mathcal{I}_{t,p}\}$ and values $m_{t,1}, \ldots, m_{t,p} \in \mathbb{R}$. For $t \in T_r$ and $i = 1, \ldots, p$, the sets $\mathcal{I}_{t,i}$, which we refer to as leaves, are hyperrectangles

$$\mathcal{I}_{t,i} \in \left\{ \prod_{j=1}^{d} A_j \;\middle|\; A_j \in \mathcal{A} \text{ for } j \in t,\ A_j = \mathbb{R} \text{ for } j \notin t \right\} \tag{4.2.2}$$

with

$$\mathcal{A} = \{A \subseteq \mathbb{R} \mid A = (b, c] \text{ or } A = (b, \infty) \text{ for } b \in [-\infty, \infty), c \in \mathbb{R}\}.$$

The corresponding estimator for m_t is given by

$$\widehat{m}_t(x) = \sum_{j=1}^{p} m_{t,j} \mathbb{1}(x \in \mathcal{I}_{t,1}).$$

Note that for any $x \in \mathbb{R}^d$, due to (4.2.2), $\widehat{m}_t$ only depends on the value of x_t. The sets $\mathcal{I}_{t_j}$ are obtained by an iterative procedure, where in each step a leaf is selected and split into two new leaves. In the following, we usually call the components m_t trees. In the definition of trees in the literature, the leaves of a tree typically form a partition of the domain. However, in general, the leaves of trees in the rpf algorithm are neither disjoint, nor do they unify to the domain. The algorithm provides an estimator of the regression function m from an ensemble of tree families. Each tree family estimates the function m from a bootstrap sample. The final estimator is the average of the estimators derived from the different tree families. A family of planted trees consists of a tree for each set of coordinates $t \in T_r$, where a tree is grown using only information on its respective coordinates. Thus, the resulting estimator is in the form of (4.2.1).
An illustration of the algorithm unfolding for a family of planted trees is given in Figure 4.1. Nodes correspond to leaves. The children of a leaf are connected via edges and are located below the respective node. The vertical position of a node indicates when the split occurred. The final leaves are located at the bottom. If a node is connected to a node above via a dashed edge, this represents a split where the original leaf is not replaced. This happens when growing from one tree to another. The example given here includes three coordinates. Each data point is contained in between three and six leaves. Note that the illustration does not contain a single

node of the tree corresponding to $t = \emptyset$. Instead, the top node of tree 1, tree 2 and tree 3 correspond to the tree $t = \emptyset$. This choice was made to simplify the illustration.

We proceed by explaining the algorithm precisely, including most technical details. We start by introducing the calculation of a split in Subsection 4.2.1 followed by the iterations in Subsection 4.2.2. We conclude with the extension to a forest and a discussion on the driving parameters.

4.2.1 Calculating a Split

We are handed a coordinate $k \in \{1, \dots, d\}$, a tree $t \in T_r$, a set $\mathcal{I} \subseteq \mathbb{R}^d$, a current value $m_{\mathcal{I}}$, residuals R_i, observations X_i, and a value $c \in \mathbb{R}$. Define a partition $\mathcal{I} = \mathcal{I}^+ \cup \mathcal{I}^-$ by

$$\mathcal{I}^+ := \{x \in \mathcal{I} \mid x_k > c\}, \qquad \mathcal{I}^- := \{x \in \mathcal{I} \mid x_k \leq c\}.$$

Let $g_{\mathcal{I}^+} := \sum_{X_i \in \mathcal{I}^+} R_i / \sum_{X_i \in \mathcal{I}^+} 1$, $g_{\mathcal{I}^-} := \sum_{X_i \in \mathcal{I}^-} R_i / \sum_{X_i \in \mathcal{I}^-} 1$ and

$$g_{\mathcal{I},c} : \mathbb{R}^d \to \mathbb{R},\ g_{\mathcal{I},c}(x) = \mathbb{1}(x \in \mathcal{I}^+) g_{\mathcal{I}^+} + \mathbb{1}(x \in \mathcal{I}^-) g_{\mathcal{I}^-}.$$

Note that $g_{\mathcal{I},c}$ is constant on each of the sets $\mathcal{I}^+$, $\mathcal{I}^-$ and $\mathbb{R}^d \backslash \mathcal{I}$. Using the function $g_{\mathcal{I},c}$, we update the residuals $R_i^{\text{new}} := R_i - g_{\mathcal{I},c}(X_i)$ and values

$$m_{\mathcal{I}^+}^{\text{new}} := m_{\mathcal{I}} + g_{\mathcal{I}^+}, \qquad m_{\mathcal{I}^-}^{\text{new}} := m_{\mathcal{I}} + g_{\mathcal{I}^-}. \tag{4.2.3}$$

Pseudo-code of this procedure is given in Algorithm 5.

Algorithm 5 Calculating a Split

```
Input $k, t, \mathcal{I}, c, m_{\mathcal{I}}, R_1, \dots, R_n, X_1, \dots, X_n$
Calculate $\mathcal{I}^+, \mathcal{I}^-$
for $i = 1, \dots, n$ do
    $R_i := R_i - g_{\mathcal{I},c}(X_i)$;
Calculate $m_{\mathcal{I}^+} := m_{\mathcal{I}} + g_{\mathcal{I}^+}$;    $m_{\mathcal{I}^-} := m_{\mathcal{I}} + g_{\mathcal{I}^-}$
Output $m_{\mathcal{I}^+}$, $m_{\mathcal{I}^-}$, $\mathcal{I}^+$, $\mathcal{I}^-$, $R_1, \dots, R_n$
```

4.2.2 Planted Tree Family

We start off by setting the residuals $R_i^{(0)} := Y_i$ for $i = 1, \dots, n$. We set a single leaf $I_{\emptyset,1}^{(0)} := \mathbb{R}^d$ and thus the number of leaves for $t = \emptyset$ is $p_{\emptyset}^{(0)} = 1$. The corresponding value is $m_{\emptyset,1} = 0$. The trees $t \neq \emptyset$ have no starting leaves and thus, $p_t^{(0)} = 0$.
An iteration step is carried out as follows. In step s we are handed residuals $R_i^{(s-1)}$, leaves $\mathcal{I}_{t,1}^{(s-1)}, \dots, \mathcal{I}_{t,p_k^{(s-1)}}^{(s-1)}$ and values $m_{t,1}^{(s-1)}, \dots, m_{t,p_k^{(s-1)}}^{(s-1)}$. The leaves are of the form (4.2.2). In each step s, we select a tree t_s, a coordinate $k_s = 1, \dots, d$, an index $j_s = 1, \dots, p_{t_s}^{(s-1)}$, and a split point c_s in order to split $\mathcal{I}_{t_s,j_s}^{(s-1)}$ using Algorithm 5. The chosen combination must be viable in the sense that it must fulfill the following three conditions.

(C_1) $p_{t_s}^{(s-1)} \geq 1$.

(C_2) If $|t_s| = r$, then $k_s \in t_s$.

(C_3) We have

$$\{i \in \{1, \dots, n\} \mid X_{i,k_s} \in \mathcal{I},\ X_{i,k_s} > c_s\}, \{i \in \{1, \dots, n\} \mid X_{i,k_s} \in \mathcal{I},\ X_{i,k_s} \leq c_s\} \neq \emptyset.$$

Now, there are two cases with different updating procedures for the corresponding estimators. If $k_s \in t_s$, we use $m_{\mathcal{I}} := m_{t_s,j_s}^{(s-1)}$ as an input for Algorithm 5. The leaf $\mathcal{I}_{t_s,j_s}^{(s-1)}$ is replaced by $\mathcal{I}^+$ and $\mathcal{I}^-$ in tree t_s. If $k_s \notin t_s$, we use $m_{\mathcal{I}} := 0$ as an input for Algorithm 5. The sets $\mathcal{I}^+$,$\mathcal{I}^-$ are added to the tree $\{k_s\} \cup t_s$. The corresponding values and residuals are updated according to Algorithm 5. In order to select t_s, k_s, j_s, and c_s, we use the CART methodology. For suitable t, k, j, and c, let $R_i^{t,k,j,c}$ denote the residual one obtains by using Algorithm 5 with inputs k, t $\mathcal{I} := \mathcal{I}^{(s-1)}(t, j)$, c, and $m_{\mathcal{I}} := m_{t,j}^{(s-1)}$, if $k \in t$, $m_{\mathcal{I}} := 0$ otherwise. We now select t_s, k_s, j_s, and c_s as

$$(t_s, k_s, j_s, c_s) := \arg\min_{t,k,j,c} \sum_{i=1}^{n} (R_i^{t,k,j,c})^2. \tag{4.2.4}$$

Note that this procedure guarantees that the leaves $\mathcal{I}_{t,j}^{(s)}$ of every tree t_s are of the form (4.2.2) for all s. Therefore, they are easy to keep track of. The algorithm stops after `nsplits` iterations. Pseudo-code of this procedure is given in Algorithm 6.

Algorithm 6 Random Planted Trees

```
Input $(Y_1, X_1), \dots, (Y_n, X_n)$
$R_i \leftarrow Y_i$;   $\mathcal{I}_{\emptyset,1} \leftarrow \mathbb{R}^d$;   $m_{\emptyset,1} \leftarrow 0$;   $p_t \leftarrow 0$;   $p_\emptyset \leftarrow 1$
for $s = 1, \dots,$ nsplits do
    Calculate $t_s, k_s, j_s, c_s$ using Equation (4.2.4)
    $k \leftarrow k_s$;   $t \leftarrow t_s$;   $\mathcal{I} \leftarrow \mathcal{I}_{t_s,j_s}$;
    if $k_s \in t_s$ then
        $m_{\mathcal{I}} \leftarrow m_{t_s,j_s}$
        Calculate $m_{\mathcal{I}^+}, m_{\mathcal{I}^-}$, $\mathcal{I}^+$,$\mathcal{I}^-$, $R_1, \dots, R_n$ (Algorithm 5)
        $\mathcal{I}_{t_s,j_s} \leftarrow \mathcal{I}^+$;   $\mathcal{I}_{t_s,p_{t_s}+1} \leftarrow \mathcal{I}^-$
        $m_{t_s,j_s} \leftarrow m_{\mathcal{I}^+}$;   $m_{t_s,p_{t_s}+1} \leftarrow m_{\mathcal{I}^-}$;   $p_{t_s} \leftarrow p_{t_s} + 1$
    else
        $m_{\mathcal{I}} \leftarrow 0$
        Calculate $m_{\mathcal{I}^+}, m_{\mathcal{I}^-}$, $\mathcal{I}^+$,$\mathcal{I}^-$, $R_1, \dots, R_n$ (Algorithm 5)
        $t_s \leftarrow t_s \cup k_s$
        $\mathcal{I}_{t_s,p_{t_s}+1} \leftarrow \mathcal{I}^+$;   $\mathcal{I}_{t_s,p_{t_s}+2} \leftarrow \mathcal{I}^-$
        $m_{t_s,p_{t_s}+1} \leftarrow m_{\mathcal{I}^+}$;   $m_{t_s,p_{t_s}+2} \leftarrow m_{\mathcal{I}^-}$;   $p_{t_s} \leftarrow p_{t_s} + 2$
```

4.2.3 From a Tree Family to a Forest

In order to reduce variance, an extension of Algorithm 6 similar to the random forest procedure is considered. We draw `ntrees` independent bootstrap samples $(Y_1^b, X_1^b), \dots, (Y_n^b, X_n^b)$ from our original data. On each of these bootstrap samples, we apply Algorithm 6 with two minor adjustments. Instead of minimizing over all $c \in \mathbb{R}$ satisfying the constraint (C_3), in each iteration we uniformly at random select `split_try` values for each coordinate in each leaf $\mathcal{I}$ in each tree t. For each coordinate k, the values are chosen (with replacement) from

$$\big\{X_{i,k} \;\big|\; X_{i,k} \in \mathcal{I},\ X_{i,k} \neq \max\{X_{i,j} \mid X_{i,j} \in \mathcal{I}\}\big\}.$$

Secondly, define a set representing the viable combinations

$$V_r := \big\{(k,t) \in \{1,\dots,d\} \times T_r \;\big|\; k \in t,\ \max\{p_t, p_{t\setminus k}\big\} \geq 1\}$$

In each iteration step we select a subset $M \subseteq V_r$ uniformly at random, where $|M| = \lceil |V_r| \cdot \texttt{t\_try} \rceil$ for some given value $\texttt{t\_try} \in (0,1]$. In step 4, when calculating (t_s, k_s, j_s, c_s) in Equation (7.2.2), we minimize under the additional conditions that c is one of the `split_try` values and $(k_s, t_s) \in M$ or $(k_s, t_s \backslash k_s) \in M$. The resulting estimators are $\widehat{m}_t^b(x) := \sum_{j=1}^{p_t^b} \mathbb{1}(x \in \mathcal{I}_{t,j}^b) m_{t,j}^b$ for $b = 1, \dots, \texttt{ntrees}$. The overall estimators are given by

$$\widehat{m}_t(x) := \frac{1}{\texttt{ntrees}} \sum_{b=1}^{\texttt{ntrees}} \widehat{m}_k^b(x).$$

4.3 Discussion of Parameters

In this section, we shortly discuss the parameters introduced and explain why choices were made when designing the algorithm. Further remarks on the algorithm can be found in Section 6.1.

4.3.1 t_try

In the rpf algorithm, the number of combinations of trees and coordinates considered in each iteration step is defined as a fixed proportion `t_try` of the number of viable combinations $|V|$. Thus, the number of combinations considered increases as the algorithm unfolds. We assessed many different similar mechanisms in order to restrict the number of combinations used in an iteration step. First of all, we found that using a random subset of the viable combinations instead of simply restricting the coordinates for splitting is far superior. Roughly speaking, the reason is that the algorithm may lock itself in a tree if all trees are splitable in each step. The question remains how to quantify the amount of combinations used. Selecting a constant number of viable combinations has the disadvantage of either allowing all combinations in the beginning or having essentially random splits towards the end of the algorithm. Thus, the number of combinations considered must depend on the number of viable combinations. While other functional connections between the number of viable and the number of considered combinations are possible, choosing a proportional connection seems natural.

4.3.2 split_try

In contrast to `t_try`, the parameter `split_try` implies an almost constant number of considered split points in each iteration. The `split_try` split options are selected uniformly at random with replacement. In Section 4.4, we find that the optimal value for `split_try` is usually small compared to the data size. In our experience, setting a constant proportion either leads to totally random splits for small leaves or essentially allows all splits for large leaves. The reason is that for large leaves, if a small number of data points are added or removed, the estimation is basically the same. For the same reason, having a low amount of potential split points in a large leaf is not a problem. The version we use here yielded the best results. The choice `t_try` < 1 as well as only using `split_try` split points is done in order to reduce the correlation between tree families and additionally reducing bias in latter case. It also reduces computational cost. While it is not obvious which of the two parameters, `split_try` or `t_try`, should be lowered in order to reduce variance, our results significantly improved as soon as we introduced the mechanisms regarding the parameters.

4.3.3 Driving Parameters

Although the presence of the mechanisms involving `split_try` and `t_try` improve the results of rpf, the exact value they take on is not as relevant. Rather, we consider them to be fine-tuning parameters. Similarly, the number of tree families `ntrees` in a forest does not have much impact as long as `ntrees` is large enough; in our simulation study, `ntrees` $= 50$ seemed satisfactory. The main driving parameter for the estimation quality of rpf is the number of iteration steps `nsplits`. While the estimation is quite stable under small changes of `nsplits`, strongly lowering `nsplits` results in a bias, while vastly increasing the parameter leads overfitting.

4.4 Simulations

In this section, we conduct an extensive simulation study in order to arrive at an understanding of how the rpf algorithm copes with different settings and how it compares to other methods. For each of 100 Monte-Carlo simulations $s = 1, \dots, 100$, We consider the regression setup

$$Y_i^s = m(X_i^s) + \varepsilon_i^s, , \quad i = 1, \dots, 500,$$

where $\varepsilon_i^s \sim N(0,1)$ i.i.d. with twelve different models, i.e. different functional shapes of m. The models are outlined in Table 4.1. In sparse models, we consider cases with $d = 4, 10, 30$ predictors. For dense models we use $d = 4, 10$. Following [99], the predictors $(X_{i,1}^s, \dots, X_{i,d}^s)$ are distributed as follows. We first generate $(\widetilde{X}_{i,1}^s, \dots, \widetilde{X}_{i,d}^s)$ from a d-dimensional standard multi-normal distribution with mean equal to 0 and $\mathrm{Cov}(\widetilde{X}_{i,j}^s, \widetilde{X}_{i,l}^s) = \mathrm{Corr}(\widetilde{X}_{i,j}^s, \widetilde{X}_{i,l}^s) = 0.3$ for $j \neq l$. Then, we set

$$X_{i,k}^s = 2.5\pi^{-1}\mathrm{arctan}(\widetilde{X}_{i,k}^s).$$

This procedure is repeated independently 500 times. The methods we compare are given in Table

Table 4.1: Dictionary for model descriptions. In total we consider 12 models. Each model has a model structure (first six rows) as well as a function shape (last two rows). The constant d denotes the total number of predictors.

Description	Meaning
additive+sparse	$m(x) = m_1(x_1) + m_2(x_2)$
hierarchical-interaction +sparse	$m(x) = m_1(x_1) + m_2(x_2) + m_3(x_3) + m_{1,2}(x_1, x_2) + m_{2,3}(x_2, x_3)$
pure-interaction+sparse	$m(x) = m_{1,2}(x_1, x_2) + m_{2,3}(x_2, x_3)$
additive+dense	$m(x) = m_1(x_1) + \cdots + m_d(x_d)$
hierarchical-interaction+dense	$m(x) = \sum_{k=1}^{d} m_k(x_k) + \sum_{k=1}^{d-1} m_{k,k+1}(x_k, x_{k+1})$
pure-interaction+dense	$m(x) = \sum_{k=1}^{d-1} m_{k,k+1}(x_k, x_{k+1})$
smooth	$m_k(x_k) = (-1)^k 2\sin(\pi x_k)$ $m_{k,k+1}(x_k, x_{k+1}) = m_k(x_k x_{k+1})$
jump	$m_k(x_k) = \begin{cases} (-1)^k 2\sin(\pi x_k) - 2 & \text{for } x \geq 0, \\ (-1)^k 2\sin(\pi x_k) + 2 & \text{for } x < 0 \end{cases}$ $m_{k,k+1}(x_k, x_{k+1}) = m_k(x_k x_{k+1})$

4.2[2]. Performance is evaluated by the average mean squared error (MSE) from 100 simulations.

Table 4.2: Dictionary for algorithms. In total we consider 7 different ones. Additionally, 2 toy algorithms are considered for benchmark values.

Description	short	Code-Reference
A gradient boosting variant	xgboost	[21]
rpf	rpf	Github
Random forest	rf	[130]
Smooth backfitting[3]	sbf	Github
Generalized additive models[4]	gam	[129]
Bayesian additive regression trees	BART	[115]
multivariate additive regression splines	MARS	[50]
1 nearest neighbours	1-NN	
Sample average	mean	

The MSE is evaluated on test points $X^s_{501}, \ldots, X^s_{1000}$ which are generated independently of the data:

$$\frac{1}{100} \sum_{s=1}^{100} \frac{1}{500} \sum_{i=501}^{1000} \{m(X_i^s) - \widehat{m}^s(X_i^s)\}^2,$$

where s runs over the different simulations and $\widehat{m}^s$ represents the estimator depending on the parameters in question as well as the data $(X_i^s, Y_i^s)_{i=1}^{500}$.

For xgboost and rpf, we record results for the case when the estimators are additive. Additionally, we show results for estimators which can approximate higher order interaction terms. For the rpf algorithm, we further distinguish between an estimator able to approximate interaction terms of order up to two as well as an unbounded configuration. Details are presented in Table

[2]Some codes are available on GitHub: `https://github.com/PlantedML/randomPlantedForest`.

8.1 in Section 8.1. Beforehand, we ran 30 simulations to find the optimal parameter combinations for each method from the parameter options outlined in Table 8.1. These simulations where conducted independently from the final simulations using training data $(\bar{X}_1^s, \bar{Y}_1^s), \dots, (\bar{X}_{500}^s, \bar{Y}_{500}^s)$ for $s = 1, \dots, 30$. A parameter combination was considered optimal if it minimized the mean squared error averaged over all 30 simulations. For xgboost and rpf, we also considered data-driven parameter choices (indicated by CV). We ran a 10-fold cross validation considering all parameters outlined in Table 8.1 including all sub-methods and options. The initial parameter options are hand-picked from preliminary simulation runs. Note that we did not optimize parameters for the method gam. Instead, we used a fully data-driven version. We now present the results from 100 Monte Carlo simulations. We only show selected parts of the overall study here. Additional tables can be found in Subsection 8.1.
Table 4.3 contains the additive sparse smooth setting. We observe that algorithms relying on continuous estimators (sbf, gam, MARS) outperform the others with gam being the clear winner, irrespective of the number of predictors d. From the other algorithms, bayesian additive random trees and additive (`max_interaction=1`) rpf do best with similar performance between the two algorithms. Note that smooth backfitting (sbf) falls off considerably for $d = 10, 30$, hinting that it cannot take advantage of the sparsity condition. This is not surprising given that no penalty terms are used for the L_2 losses.

Table 4.3: Model 1: Additive Sparse Smooth Model. We report the average MSE from 100 simulations. The standard deviations are provided in brackets.

Method	dim=4	dim=10	dim=30
xgboost (`depth=1`)	0.119 (0.021)	0.142 (0.021)	0.176 (0.027)
xgboost	0.141 (0.024)	0.166 (0.028)	0.193 (0.033)
xgboost-CV	0.139 (0.028)	0.152 (0.029)	0.194 (0.035)
rpf (`max_interaction=1`)	0.087 (0.018)	0.086 (0.017)	0.097 (0.019)
rpf (`max_interaction=2`)	0.107 (0.015)	0.121 (0.025)	0.142 (0.026)
rpf	0.112 (0.017)	0.134 (0.026)	0.162 (0.028)
rpf-CV	0.103 (0.02)	0.102 (0.035)	0.105 (0.022)
rf	0.209 (0.021)	0.252 (0.027)	0.3 (0.029)
sbf	0.071 (0.026)	0.134 (0.013)	0.388 (0.073)
gam	0.033 (0.012)	0.035 (0.013)	0.058 (0.021)
BART	0.085 (0.019)	0.076 (0.017)	0.091 (0.023)
BART-CV	0.09 (0.019)	0.081 (0.014)	0.09 (0.02)
MARS	0.054 (0.014)	0.061 (0.025)	0.076 (0.031)
1-NN	1.509 (0.1)	3.228 (0.182)	5.534 (0.313)
average	3.811 (0.217)	3.689 (0.183)	3.748 (0.202)

Interestingly, random rpf (`max_interaction=2`) and rpf perform very well and even outperform the additive (`depth=1`) xgboost. Random forest (rf) shows the worst performance in this setting. However, it seems to catch up to smooth backfitting with increasing dimension, indicating that it makes better use of the sparsity condition.
In the hierarchical-interaction sparse smooth setting, which is visualized in Table 4.4, we only show results from methods which can deal with interactions. Other cases are deferred to Sec-

tion 8.1. Here, we find that BART as well as multivariate additive regression splines (MARS) outperform our rpf algorithm. However, the latter slightly outperforms xgboost. In the Table 8.6 in Section 8.1, a similar picture can be observed in the pure-interaction case. In particular, BART proves to be much stronger than all competing algorithms. This indicates that BART does not rely on a hierarchical interaction structure as much as the other methods. However, it comes at the cost of increased computational time. Recall that in contrast to rpf, MARS provides a continuous estimator and the estimator provided by BART is not interpretable. Next, we re-visit the additive case. However, this time, the regression function is not continuous. The results are tabulated in Table 8.2 in Section 8.1. In this setting, our additive rpf algorithm and the BART algorithm perform best. A visual picture of the excellent fit of rpf was provided in Figure 4.3. Random forest and xgboost share the third position. The best interpretable competitor is xgboost (additive). Unsurprisingly, models based on continuous estimators can not deal with the jumps in the regression functions. Cases including interaction terms and jumps in the regression function are deferred to Section 8.1.

Table 4.4: Model 2&3: Hierarchical and Pure Interaction Sparse Smooth Model. We report the average MSE from 100 simulations. The standard deviations are provided in brackets.

	Method	dim=4	dim=10	dim=30
Hierarchical-interaction	xgboost	0.374 (0.035)	0.481 (0.064)	0.557 (0.089)
	xgboost-CV	0.393 (0.051)	0.499 (0.058)	0.563 (0.089)
	rpf (`max_interaction=2`)	0.248 (0.038)	0.327 (0.045)	0.408 (0.07)
	rpf	0.263 (0.034)	0.357 (0.044)	0.452 (0.076)
	rpf-CV	0.277 (0.039)	0.366 (0.051)	0.463 (0.083)
	rf	0.432 (0.039)	0.575 (0.061)	0.671 (0.08)
	BART	0.214 (0.03)	0.223 (0.04)	0.252 (0.037)
	BART-CV	0.242 (0.043)	0.276 (0.053)	0.315 (0.047)
	MARS	0.355 (0.089)	0.282 (0.038)	0.414 (0.126)
	1-NN	2.068 (0.156)	5.988 (0.624)	11.059 (0.676)
	average	8.366 (0.43)	8.086 (0.246)	8.207 (0.496)

We now consider dense models. In the additive dense smooth model, see Table 8.3 in Section 8.1, the clear winners are sbf and gam, with gam having the best performance. This is not surprising, since these methods have more restrictive model assumptions than the others. Therefore, if the assumptions are met, one would expect the methods to do well. Additionally, gam typically has an advantage over sbf in our setting. This is because splines usually fit trigonometric curves well, while kernel smoothers would be better off with a variable bandwidth in this case, which we did not implement. Observe that MARS does not deal well with this situation, in particular in the case d=10. Our rpf method is in third place, tied with BART and slightly outperforming xgboost. This suggests that the rpf algorithm is especially strong in sparse settings. In dense settings, the advantage towards xgboost and BART shrinks. This observation is underpinned by the next scenario.

As a final setting, we consider the hierarchical-interaction dense smooth model. The results are tabulated in Table 8.4 in Section 8.1. The results further strengthen our intuition that the current version of the rpf algorithm performs worse in dense settings. While it is by far the best performing algorithm with only four predictors, the performance deteriorates faster than that of other methods when adding predictors. In dimension 10, i.e, with ten predictors, xboost outperforms rpf whereas the reverse can be observed for d=4.

Concluding remarks on simulations results From the conducted simulation study, we find that BART showed the strongest performance while the rpf algorithm is the runner-up. Compared to BART, rpf has the advantage that results can be easily plotted and interpreted (if only lower-interactions are fitted). Additionally, the BART algorithm is computationally quite intensive and we indeed struggled to get BART running in the higher dimensional cases without error already in our study with a sample size of 500 (probably because of memory limitations). Xgboost showed very strong results in terms of accuracy while being very fast and resources effective. However, accuracy turns out to be slightly worse than rpf in low-interaction settings and it does not provide the possibility to visualize the fit.

4.5 Theoretical Properties

In this section we derive asymptotic properties for a slightly modified rpf algorithm. The main difference is that in the tree family construction of the modified forest estimator, splitting does not depend directly on the responses $(Y_i : i = 1, ..., n)$. With this modification we follow other studies of forest based algorithms to circumvent mathematical difficulties which arise if settings are analyzed where the same data is used to choose the split points as well as to calculate the fits in the leaves. Clearly, one can apply the results in this section to a similar modification of rpf making use of data splitting to separate splitting and fitting. Our results imply two findings. First, for $r \leq 2$ the estimator can achieve optimal rates up to logarithmic terms in the nonparametric model where the interaction terms m_t (for $|t| \leq 2$) allow for continuous second order derivatives. Secondly, for all choices of r one achieves faster rates of convergence for the forest estimator than for tree family estimators that are based on calculating only one single tree family. We comment below why the situation changes for $r \geq 3$ compared to $r \leq 2$. A major challenge in studying rpf lies in the fact that the estimator is only defined as the result of an iterative algorithm and not as the solution of an equation or of a minimizing problem. In particular, our setting differs from other studies of random forests where tree estimates are given by leaf averages of terminal nodes. In such settings the tree estimator only depends on the leaves of terminal nodes, but not on other structural elements of the tree, and in particular not on the way the tree was grown. Secondly, the definition of the estimator as a leaf average allows for simplifications in the mathematical analysis. We cannot make use of either of these advantages. The main point of our mathematical approach is to show that approximately, the tree family estimators and the forest estimators are given by a least squares problem defined by the leaves of at the end of the algorithm. We assume that the regression function fulfills

$m(x) := m^0(x) = \sum_{t \in T_r} m_t^0(x_t)$ for some $r \in \mathbb{N}$. The data is generated as

$$Y_i = m^0(X_i) + \varepsilon_i = \sum_{t \in T_r} m_t^0(X_{i,t}) + \varepsilon_i,$$

where the additive components $m_t^0 : t \in T_r$ are smooth functions. In the model equation, the ε_i's are mean zero error variables, and, for simplicity, the covariables $X_{i,k}$ are assumed to lie in $[0,1]$; $i = 1, \dots, n; k = 1, \dots, d$.

We now describe the iteration steps of the tree family algorithm discussed in this section. As initialisation we set $\mathcal{I}_{t,1}^0 = [0,1]^d$, $L_{t,0} = 1$ and $\widehat{m}_t^0 \equiv 0$ for $t \in T_r$. In iteration steps $s = 1, ..., S$ partitions $\mathcal{I}_{t,l}^s = \prod_{k \in t}(a_{t,k,l}^s, b_{t,k,l}^s] \times \prod_{k \notin t}(0,1]$ of $[0,1]^d$ with $l = 1, ..., L_{t,s}$ are updated by splitting one of the rectangles $\mathcal{I}_{t,l}^s$ for one $t \in T_r$, one $l \in \{1, ..., L_{t,s}\}$ along one coordinate k. Here, in abuse of notation, we write $\prod_{k \in t} \cdot \times \prod_{k \notin t} \cdot$ for the set of tuples with coordinates ordered according to the value of k and not according to the appearance in the product sign subindices. We now describe step s where the rectangles $\mathcal{I}_{t,l}^{s-1}$ are updated. At step s one chooses a $t_s \in T_r$, an $l_s \in \{1, ..., L_{t_s,s-1}\}$, a $k_s \in t_s$ and a splitting value $b_{t_s,k_s,l_s}^s \in (a_{t_s,k_s,l_s}^{s-1}, b_{t_s,k_s,l_s}^{s-1})$. The values are chosen by some random procedure that satisfies the assumptions $(A_1), \dots, (A_8)$ below. Using these values one splits $(a_{t_s,k_s,l_s}^{s-1}, b_{t_s,k_s,l_s}^{s-1}]$ into $(a_{t_s,k_s,l_s}^s, b_{t_s,k_s,l_s}^s]$ and $(a_{t_s,k_s,L_{t_s,s}}^s, b_{t_s,k_s,L_{t_s,s}}^s]$, where $a_{t_s,k_s,l_s}^s = a_{t_s,k_s,l_s}^{s-1}$, $a_{t_s,k_s,L_{t_s,s}}^s = b_{t_s,k_s,l_s}^s$, $b_{t_s,k_s,L_{t_s,s}}^s = b_{t_s,k_s,l_s}^{s-1}$ and $L_{t_s,s} = L_{t_s,s-1} + 1$. For $(k,l) \neq (k_s, l_s)$ we set $(a_{t_s,k,l}^s, b_{t_s,k,l}^s] = (a_{t_s,k,l}^{s-1}, b_{t_s,k,l}^{s-1}]$. Then we update the rectangle $\mathcal{I}_{t_s,l}^s$ for $l = l_s$ and $l = L_{t_s,s}$ by defining $\mathcal{I}_{t_s,l}^s = \mathcal{I}_{t_s,l}^{\text{marg},s} \times \prod_{k \notin t_s}(0,1]$ with $\mathcal{I}_{t_s,l}^{\text{marg},s} = \prod_{k \in t_s}(a_{t_s,k,l}^s, b_{t_s,k,l}^s]$. All other rectangles are taken over identically from the last step. Finally, for the chosen tree t_s, the values $\widehat{m}_{t_s}^s$ are updated by averaging residuals over the intervals $\mathcal{I}_{t_s,l}^s$ for $l = 1, ..., L_{t_s,s}$. For a subset $\mathcal{I}$ of $[0,1]^d$ we write $|\mathcal{I}|$ for the Lebesgue measure of the set. Note that for finite sets Q, we also use $|Q|$ for the number of Elements in Q. From context, it is always clear which version is used. Additionally, we write $|\mathcal{I}|_n = |\{i : X_i \in \mathcal{I}\}|/n$ for the empirical measure of $\mathcal{I}$. Then, for $x_{t_s} \in \mathcal{I}_{t_s,l}^{\text{marg},s}$ with $l \in \{1, ..., L_{t_s,s}\}$ the estimator can be written as

$$\widehat{m}_{t_s}^s(x_{t_s}) \quad = \quad \frac{1}{n|\mathcal{I}_{t_s,l}^s|_n} \sum_{i: X_i \in \mathcal{I}_{t_s,l}^s} \left(Y_i - \sum_{t \in T_r, t \neq t_s} \widehat{m}_t^{s-1}(X_{i,t}) \right).$$

Because $\widehat{m}_t^{s-1}$ is constant on the set $\mathcal{I}_{t,k}^s$ we can rewrite the formula with $\widehat{m}_t^{s-1}(\mathcal{I}_{t,k}^s)$ equal to $\widehat{m}_t^{s-1}(u)$ for $u \in \mathcal{I}_{t,k}^s$ as follows

$$\widehat{m}_{t_s}^s(x_{t_s}) \quad = \quad \widehat{\widehat{m}}_{t_s}^s(x_{t_s}) - \frac{1}{|\mathcal{I}_{t_s,l}^s|_n} \sum_{t \in T_r, t \neq t_s} \sum_{k=1}^{L_{t,s}} |\mathcal{I}_{t_s,l}^s \cap \mathcal{I}_{t,k}^s|_n \ \widehat{m}_t^{s-1}(\mathcal{I}_{t,k}^s).$$

Here for $x_{t_s} \in \mathcal{I}_{t_s,l}^{\text{marg},s}$ with $l \in \{1, ..., L_{t_s,s}\}$ the function $\widehat{\widehat{m}}_{t_s}^s$ is a marginal estimator defined by $\widehat{\widehat{m}}_t^s(x_t) = \frac{1}{n|\mathcal{I}_{t,l}^s|_n} \sum_{X_i \in \mathcal{I}_{t,l}^s} Y_i$. Furthermore we define the density estimator $\widehat{p}_t^s(x_t) = |\mathcal{I}_{t,l}^s|_n / |\mathcal{I}_{t,l}^s|$ for $x \in \mathcal{I}_{t,l}^s$ and the $|t \cup t'|$-dimensional estimator $\widehat{p}_{t \cup t'}^s(x_{t \cup t'}) = |\mathcal{I}_{t,l}^s \cap \mathcal{I}_{t',l'}^s|_n / |\mathcal{I}_{t,l}^s \cap \mathcal{I}_{t',l'}^s|$ for

$x \in \mathcal{I}^s_{t,l} \cap \mathcal{I}^s_{t',l'}$. With this notation we get

$$\widehat{m}^s_{t_s}(x_{t_s}) = \widehat{\widehat{m}}^s_{t_s}(x_{t_s}) - \sum_{t \in T_r, t \neq t_s} \int_{(0,1]^{|t \setminus t_s|}} \frac{\widehat{p}^s_{t_s,t}(x_{t_s}, u_{t \setminus t_s})}{\widehat{p}^s_{t_s}(x_{t_s})} \widehat{m}^{s-1}_t(x_{t \cap t_s}, u_{t \setminus t_s}) \mathrm{d}u_{t \setminus t_s}. \quad (4.5.1)$$

Here and in the following, we sometimes write $f(x)$ instead of $f(x_t)$ for functions f that depend only on x via x_t, e.g. $\widehat{m}^{s-1}_t(x) = \widehat{m}^{s-1}_t(x_t)$. For $t \neq t_s$ we set $\widehat{m}^s_t = \widehat{m}^{s-1}_t$. After S (S corresponds to `nsplits`) steps we get the estimators $\widehat{m}^S_t$ for $t \in T_r$. These estimators are not calibrated in the sense that we do not have $\sum_{i=1}^n \widehat{m}^S_t(X^i_{t \setminus t'}, x_{t'}) = 0$ for $t' \subsetneq t$ and $x_{t'} \in [0,1]^{|t'|}$. Calibration can be achieved by straight forward modifications of $\widehat{m}^S_t$ which may require further splits of rectangles $\mathcal{I}^S_{t',l}$ for $t' \subsetneq t$. This is not discussed here.

Now, the tree family estimator of the function m^0 is given by $\widehat{m}^S(x) = \sum_{t \in T_r} \widehat{m}^S_t(x_t)$. For the forest estimator, tree family estimators are repeatedly constructed. They are denoted by $\widehat{m}^{S,v}_t$ for $t \in T_r$ and $\widehat{m}^{S,v}$ for $v = 1, ..., V$ (V corresponds to `ntrees`). We define the forest estimator as $\widehat{m}_t = V^{-1} \sum_{v=1}^V \widehat{m}^{S,v}_t$ for $t \in T_r$ and $\widehat{m} = V^{-1} \sum_{v=1}^V \widehat{m}^{S,v}$. If necessary, we also write $\mathcal{I}^{s,v}_{t,l}$, $a^{s,v}_{t,k,l}$, t^v_s ,k^v_s, ... instead of $\mathcal{I}^s_{t,l}$, $a^s_{t,k,l}$, t_s ,k_s, ... to indicate that we discuss the tree family estimator with index v.

Considering the above, there are three further modifications in the theoretical version of the algorithm studied in this section compared to the algorithm of the preceding sections. First, we construct trees based on the full sample and not on bootstrap samples. Below, we observe that bootstrap is necessary neither for rate optimality for $r \leq 2$ nor rate improvement by averaging tree families. Our results can be generalized to versions that make use of bootstrap. Second, each tree in a family is grown from an own root. Higher order trees do not come from lower order ones. This assumption is made to simplify mathematical theory. Third, in the iteration steps we make an update of the estimator $\widehat{m}^s_{t_s}(x)$ for all rectangles $\mathcal{I}^s_{t_s,l}$ ($l = 1, ..., L_{t_s,s}$) of the chosen tree t_s. In our implementation of rpf in the preceding sections we only did this for the splitted rectangle, i.e. for $l = l_s$ and $l = L_{t_s,s}$. From simulations, we concluded that the third change is not severe. For our result, we make use of the following assumptions.

Assumption (A1) The tuples (X_i, ε_i) are i.i.d. The functions $m^0_t : [0,1]^{|t|} \to \mathbb{R}$ are twice continuously differentiable and $E[m^0_t(X_{i,t})] = 0$ for all $t \in T_r$. The covariable X_i has a density p that is bounded from above and below (away from 0). For $t \in T_{2r}$, the joint density p_t of the tuple $X_{i,t}$ allows continuous derivatives of order 2. Conditionally on X_i and the iterative constructions of the leaves in the trees, the error variables ε_i have mean zero and variance bounded by a constant and the products $\varepsilon_i \varepsilon_j$ are mean zero for $i \neq j$. Conditionally on X_i, the iterative constructions of the leaves in the tree families are i.i.d. for $v = 1, ..., V$.

Assumption (A2) The number S of iterations in the tree family construction and the number V of contructed trees is allowed to depend on n.

Note that S is of the same order as the number of rectangles in the final partition. S is assumed to converge to infinity, see (A3). When considering the forest estimator one usually requires V to converge to infinity in order to obtain useful convergence rates, see also (A7),(A8).

Assumption (A3) For $C_{A3} > 0$ large enough we assume that there exists a constant $C'_{A3} > 0$ such that, with probability tending to one, for $\{s : S - J' \leq s \leq S\}$ and $1 \leq v \leq V$ there exist a partition $S - J' = s_0^v < s_1^v < ... < s_J^v = S$ such that for $1 \leq j \leq J$ the set $\{t_s^v : s_{j-1}^v < s \leq s_j^v\}$ contains all elements of T_r. Here J and J' are the smallest integers larger or equal to $C_{A3} \log n$ or $C'_{A3}(\log n)^2$, respectively.

Assumption (A4) For $x \in [0,1]$, $t \in T_r$, $1 \leq v \leq V$ we define $l_t(x) = l_t^v(x)$ such that $x \in \mathcal{I}^{S,v}_{t,l_t^v(x)}$. We assume that with probability tending to one uniformly over $t \in T_r$, $1 \leq v \leq V$, $k \in t$,

$$\frac{1}{n}\sum_{i=1}^{n}\left(b^{S,v}_{t,k,l_t^v(X_{i,t})} - a^{S,v}_{t,k,l_t^v(X_{i,t})}\right)^2 \leq \delta_{1,n}^2$$

for a sequence $\delta_{1,n}$ with $(\log n)^2\delta_{1,n} \to 0$ and $n^R\delta_{1,n} \to \infty$ for $R > 0$ large enough.

Assumption (A5) It holds for $t, t' \in T_r$, $1 \leq v \leq V$ and $S - J' \leq s \leq S$ that

$$\begin{aligned}\sup_{t,t' \in T_r, t \neq t'} \int_{(0,1]^{|t \cup t'|}} \left(\frac{\widehat{p}^{S,v}_{t\cup t'}(u)}{\widehat{p}^{S,v}_t(u_t)} - \frac{\widehat{p}^{s,v}_{t\cup t'}(u)}{\widehat{p}^{s,v}_t(u_t)}\right)^2 \mathrm{d}u_{t\cup t'} &\leq \delta_{1,n}^2(\log n)^{-4},\\ \sup_{t \in T_r}\int_{(0,1]^{|t|}} \left(\widehat{m}^{S,v}_t(u_t) - \widehat{m}^{s,v}_t(u_t)\right)^2 \mathrm{d}u_t &\leq \delta_{1,n}^2(\log n)^{-4}\end{aligned}$$

with probability tending to one.

Assumption (A6) It holds uniformly for $t, t' \in T_r$, $1 \leq v \leq V$ that

$$\begin{aligned}\int_{(0,1]^{|t \cup t'|}} \left(\frac{\widehat{p}^{S,v}_{t\cup t'}(u)}{\widehat{p}^{S,v}_t(u_t)} - \frac{p_{t\cup t'}(u)}{p_t(u_t)}\right)^2 \mathrm{d}u_{t\cup t'} &\leq \delta_{2,n}^2,\\ \sup_{u \in (0,1]^{|t|}} |\widehat{p}^{s,v}_t(u) - p_t(u)| &\leq \eta_{1,n} \text{ for } s = S, S - J' - 1,\end{aligned}$$

with probability tending to one for sequences $\delta_{2,n}$, $\eta_{1,n}$ with $(\log n)^2\delta_{2,n} \to 0$ and $\eta_{1,n} \to 0$.

Theorem 4.5.1. *Under (A1)–(A6), for $1 \leq v \leq V$ the tree family estimators satisfy*

$$\left\|\sum_{t \in T_r}(\widehat{m}^{S,v}_t - m^0_t)\right\| = O_P(\delta_{1,n} + S^{1/2}n^{-1/2})$$

with probability tending to one, where $\|\cdot\|$ denotes the $L_2(P)$ norm.

We shortly discuss this result. Suppose that the rectangles $\mathcal{I}^s_{t,l}$ have side lengths of order h for some sequence $h \to 0$. Then $\delta_{1,n}$ is of order h, S is of order h^{-r} and up to logarithmic terms we obtain a rate of order $h + (nh^r)^{-1/2}$ for the estimation error of $\widehat{m}^{S,v} = \sum_{t\in T_r} \widehat{m}_t^{S,v}$. For discussing the complete forest estimator, we need the following assumptions.

Assumption (A7) It holds uniformly for $t, t' \in T_r$, $1 \le v \le V$ that

$$\begin{aligned} \sup_{u\in(0,1]^{|t\cup t'|}} \left|\widehat{p}_{t\cup t'}^{S,v}(u) - p_{t\cup t'}(u)\right| &\le \eta_{2,n}, \\ \|\widehat{p}_t^{S,v} - p_t(u)\|, \left\|\int_{(0,1]^{|t'\setminus t|}} \frac{\widehat{p}_{t\cup t'}^{S,v}(u) - p_{t\cup t'}(u)}{p_t(u_t)} m_{t'}^0(u_{t'})\mathrm{d}u_{t'\setminus t}\right\| &\le \delta_{3,n} \end{aligned}$$

with probability tending to one for some sequences $\eta_{2,n}$, $\delta_{3,n}$ with $\eta_{2,n}, \delta_{3,n} \to 0$.

Assumption (A8) For $t, t' \in T_r$ we write $\widehat{p}_{t\cup t'}^{S,+} = V^{-1}\sum_{v=1}^V \widehat{p}_{t\cup t'}^{S,v}$ and $\widehat{p}_t^{S,+} = V^{-1}\sum_{v=1}^V \widehat{p}_t^{S,v}$. It holds uniformly for $t, t' \in T_r$ that

$$\|\widehat{p}_t^{S,+} - p_t(u)\|_1, \left\|\int_{(0,1]^{|t'\setminus t|}} \frac{\widehat{p}_{t\cup t'}^{S,+}(u) - p_{t\cup t'}(u)}{p_t(u_t)} m_{t'}^0(u_{t'})\mathrm{d}u_{t'\setminus t}\right\|_1 \le \delta_{4,n}$$

with probability tending to one for a sequence $\delta_{4,n} \to 0$. Here, $\|\cdot\|_1$ denotes the $L_1(P)$ norm.

Below we argue that under certain assumptions the averaged estimators in (A8) can achieve rates of convergence $\delta_{4,n} = O(\delta_{1,n}^2)$, compared to their summands which have a bias of order $\delta_{1,n}$. If the density p_t is twice continuously differentiable, qualitatively, $\widehat{p}_t^{S,+}$ behaves like a kernel density estimator with bandwidth h of order $\delta_{1,n}$. We switch from the $L_2(P)$ norm to the $L_1(P)$ norm in Assumption (A8) and in the next theorem, because at the boundary of size Ch with C large enough we have a bias of order h and in the interior the bias is of order h^2. Thus, measured by the $L_1(P)$ norm we get a bias of order h^2 whereas the $L_2(P)-$norm has a slower rate caused by the boundary effects.

Theorem 4.5.2. *Under (A1)–(A8), for the forest estimator we have*

$$\left\|\sum_{t\in T_r}(\widehat{m}_t - m_t^0)\right\|_1 \le C(\delta_{1,n}^2 + \delta_{1,n}\delta_{2,n} + \delta_{3,n}^2 + \delta_{4,n} + S^{1/2}n^{-1/2})$$

with probability tending to one.

Again, we shortly discuss this result. As above, suppose that the rectangles $\mathcal{I}_{t,l}^s$ have side lengths of order h for some sequence $h \to 0$. Then $\delta_{1,n}$ is of order h and S is of order h^{-r}. The sequence $\delta_{2,n}$ is the rate of a histogram estimator of dimension $\le 2r$ which up to logarithmic terms is of order $h + (nh^{2r})^{-1/2}$ for $2r$ dimensional estimators. Assume for $x \in [0,1]^{|t|}$ sufficiently far away from the boundary of $[0,1]^{|t|}$ the random variables $x_k - a_k^S$ and $b_k^S - x_k$ approximately follow the same distribution, where a_k^S, b_k^S are the lower and upper bounds of the leaf which contains x with respect to dimension k. Then the bias terms of order h cancel and we get that the bias terms measured by $\delta_{4,n}$ are of order h^2. Thus up to logarithmic terms, we get a bound on the accuracy of the forest estimator of order $h^2 + h(nh^{2r})^{-1/2} + (nh^r)^{-1/2}$. This rate is faster

than the tree family rate if $nh^{2r} \to \infty$, i.e. we need consistency of $2r$ dimensional histogram estimators. These histogram estimators show up as kernels in integral equations which define the tree family and forest estimator. This means that, approximately, the tree family estimators $\widehat{m}^v$ are given as solutions of integral equations of the form $\widehat{m}^v = \bar{m}^v + A^v \widehat{m}^v$ with random integral operators A^v, where the operators A^v are defined by up to $2r$ dimensional density estimators. If these density estimators are consistent the operators A^v are approximately equal to an operator A not depending of v. Then, $\widehat{m}^v$ approximately solves $\widehat{m}^v = \bar{m}^v + A\widehat{m}^v$ and the forest estimator $\widehat{m} = V^{-1}\sum_{v=1}^V \widehat{m}^v$ solves approximately $\widehat{m} = V^{-1}\sum_{v=1}^V \bar{m}^v + A\widehat{m}$. For getting faster convergence rates for the forest estimator one shows that the average $V^{-1}\sum_{v=1}^V \bar{m}^v$ has faster rates as the summands $\bar{m}^v$. For the whole argument it is crucial that up to $2r$ dimensional density estimators are consistent. Also in case these estimators are inconsistent we expect a better performance of forest estimators compared to tree family estimators. However, additional terms show up in the expansion of the rpf estimator that are not of order h^2. Let us discuss this for a bandwidth h that is rate optimal for dimension r when one estimates r dimensional twice differentiable functions. Then h is of order $n^{-1/(r+4)}$ and the $2r$-dimensional estimators are consistent only for $r \leq 3$. Thus our result shows that for optimal tuning parameters forest estimators perform better as tree family estimators for $r \leq 3$. For $r \leq 2$ we obtain optimal rates by the forest estimator, i.e. $n^{-2/5}$ for $r = 1$ and $n^{-1/3}$ for $r = 2$.
Note that our results also hold for other iterative partition schemes as long as Assumptions (A1)–(A8) are fulfilled. In particular, in the proofs we did not use that partitions are based on iterative splitting as described in the first part of this section.

4.6 Generalized Random Planted Forest

This section follows an ongoing research project with Lukas Burk, Munir Hiabu, Enno Mammen and Marvin Wright. We include the general ideas of the algorithm as well as some simulation results.

We are given data $(Y_i, X_{i,1}, \dots, X_{i,d})_{i=1}^n$ consisting of i.i.d. observations with $Y_i \in \mathbb{R}^{d_1}, X_{i,k} \in \mathbb{R}$ and wish to estimate some function $\mu : \mathbb{R}^d \to \mathbb{R}^{d_2}$. General planted forests are constructed with an iterative algorithm. The algorithm depends on a link function $\sigma : \mathbb{R}^{d_1} \to \mathbb{R}^{d_2}$, an updating function g, and a loss function L. The resulting estimator is of the form

$$\widehat{\mu}(x) = \sigma\left(\sum_{t \in \mathcal{T}_r} \widehat{m}_t(x)\right),$$

for some $r \in \{1, \dots, d\}$. For every $t \in \mathcal{T}$, the function $\widehat{m}_t$ is of the form

$$\widehat{m}_t(x) = \sum_{i=1}^{p_t} a_i \mathbb{1}(x \in \mathcal{I}_{t,i}),$$

where $a_i \in \mathbb{R}^{d_1}$ and $\mathcal{I}_{t,i}$ is a hyperrectangle satisfying (4.2.2). In order to obtain the functions $\widehat{m}_t$

a forest-type procedure similar to the one described in 4.2 is used. Alterations are the following.

- In each iteration, a single set $\mathcal{I}_{t,i}$ is split into two subsets according to Subsection 4.2.1 for some $t \in \mathcal{T}$. The estimator $\widehat{m}_t$ is updated on each of these subsets. However, instead of using equation (4.2.3) for updating, we update by setting

$$\widehat{m}^{\text{new}}_{t\cup\{k\}}(x) := \widehat{m}_{t\cup\{k\}}(x) + \mathbb{1}(x \in \mathcal{I}^+)g^+ + \mathbb{1}(x \in \mathcal{I}^-)g^-,$$

  where

$$\begin{aligned} g^+ &:= g\big(\mathcal{I}^+, X_1, \dots, X_d, Y_1, \dots, Y_d, (\widehat{m}_t)_{t\in\mathcal{T}}\big),\\ g^- &:= g\big(\mathcal{I}^-, X_1, \dots, X_d, Y_1, \dots, Y_d, (\widehat{m}_t)_{t\in\mathcal{T}}\big), \end{aligned}$$

  and g is the updating function. Pseudo code for the algorithm is given in Algorithm 7.

Algorithm 7 Calculating a Split

```
**Input** $t, k, \mathcal{I}, c, (X_1, Y_1), \dots, (X_n, Y_n), (\widehat{m}_t)_{t\in\mathcal{T}}$
**Calculate** $\mathcal{I}^+, \mathcal{I}^-$
**Calculate** $\widehat{m}_{t\cup\{k\}}(x) := \widehat{m}_{t\cup\{k\}}(x) + \mathbb{1}(x \in \mathcal{I}^+)g^+ + \mathbb{1}(x \in \mathcal{I}^-)g^-$
**Output** $\widehat{m}_{t\cup\{k\}}$, $\mathcal{I}^+$, $\mathcal{I}^-$
```

- Again, we begin with a single leaf $I^{(0)}_{\emptyset,1} := \mathbb{R}^d$ which implies $p^{(0)}_{\emptyset} = 1$. The corresponding value is $m_{\emptyset,1} = 0$. Additionally, we have $p^{(0)}_t = 0$ for $t \neq \emptyset$. The iterative algorithm is carried out as in Section 4.2.2 with one difference. Instead of using Equation (4.2.4), the values t_s, k_s, j_s, and c_s are chosen by

$$(t_s, k_s, j_s, c_s) := \arg\min_{t,k,j,c} L(t, k, j, c). \tag{4.6.1}$$

  Note that L also may depend on the data $(X_1, Y_1), \dots, (X_n, Y_n)$, the current estimators $\widehat{m}^{(s-1)}_t$ as well as the current leaves $\mathcal{I}^{(s-1)}_{t,k}$, $k = 1, \dots, p_t$, $t \in \mathcal{T}_r$. The dependencies where omitted in (4.6.1) for simpler notation. Typically, L will strongly relate to the updating function g as well as the link function σ. The algorithm stops after `nsplits` iterations. Pseudo code for this procedure is given in Algorithm 8.

- Extending the planted tree family type algorithm to a forest is analogous to Section 4.2.3.

The rest of this section is structured as follows. In Subsection 4.6.1 we include some remarks on the algorithm. In Subsections 4.6.2 – 4.6.5 we introduce some examples for combinations of link functions σ, updating functions g and loss functions L which can be used in classification setups. Lastly, Subsection 4.6.6 our current Simulation results.

4.6.1 Remarks

The algorithm described above can be used for a wide variety of estimation problems by selecting specific link, updating and loss functions. While different choices lead to different estimators,

Algorithm 8 Random Family of Planted Trees

```
Input $(Y_1, X_1), \dots, (Y_n, X_n)$
for $|t|_0 = 1$ do $\widehat{m}_t \leftarrow 0$;   $\mathcal{I}_{t,1} \leftarrow \mathbb{R}^d$;   $p_t \leftarrow 1$
for $|t|_0 > 1$ do $\widehat{m}_t \leftarrow 0$;   $p_t \leftarrow 0$
for $s = 1, \dots,$ nsplits do
    Calculate $t_s, k_s, j_s, c_s$ using Equation (4.6.1)
    $t \leftarrow t_s$;   $k \leftarrow k_s$;   $c \leftarrow t_s$;   $\mathcal{I} \leftarrow \mathcal{I}_{t_s,j_s}$
    Calculate $\widehat{m}_{t_s \cup \{k_s\}}$, $\mathcal{I}^+$,$\mathcal{I}^-$ using Algorithm 7
    if $k_s \in t_s$ then
        $\mathcal{I}_{t_s,j_s} \leftarrow \mathcal{I}^+$;   $\mathcal{I}_{t_s,p_{t_s}+1} \leftarrow \mathcal{I}^-$;   $p_{t_s} \leftarrow p_{t_s} + 1$
    else
        $t \leftarrow t_s \cup \{k_s\}$
        $\mathcal{I}_{t,p_t+1} \leftarrow \mathcal{I}^+$;   $\mathcal{I}_{t,p_t+2} \leftarrow \mathcal{I}^-$;   $p_t \leftarrow p_t + 2$
Output $(\widehat{m}_t)_{t \in \mathcal{T}}$
```

functions returned by a general planted forest algorithm have certain structural aspects. First of all, observe that the resulting estimators $\widehat{m}_t$ are of the form

$$\widehat{m}_t(x) = \sum_{i=1}^{p_t} a_i \mathbb{1}(x \in \mathcal{I}_{t,i})$$

where $\mathcal{I}_{t,i}$ are of the form (4.2.2). Thus, the functions m_t only depend on the components in t so that they can be represented by $|t|$-dimensional functions. The d-dimensional input space is used for simpler notation.

While r in the algorithm described in Section 4.2 bounds the maximal order of interaction in an additive sense, the implications of the bound r in a general random planted forest algorithm depends on the link function σ. This allows for many different structures. For example, if we set $r = d_1 = d_2 = 1$ and $\sigma(x) = \exp(x)$ the resulting estimator is multiplicative in the sense that it is an element of

$$\left\{ f : \mathbb{R}^d \to \mathbb{R} \;\middle|\; f(x_1, \dots, x_d) = \prod_{k=1}^{d} f(x_k) \right\}.$$

Note that r and σ are specified beforehand. We may however use them as tuning parameters for our algorithm and select them via some selection procedure such as cross validation. This method was also used in Subsection 4.6.6.

Note that $\widehat{\mu}$ takes on values in $\mathbb{R}^{d_2}$, where we allow for $d_2 > 1$. This is useful for classification problems which include more then two classes. Then, Y_i is of the form $Y_i = e_j \in \mathbb{R}^{d_2}$, where $e_{jk} = 1$ for $j = k$ and $e_{jk} = 0$ otherwise. Each Component of Y_i corresponds to a class, where we define $Y_{ik} = 1$ if Y_i is in the k-th class. With this definition, we grow one general random planted forests for the classification problem. Alternatively, we could also grow a general random planted forest for each class and then use some normalizing constraint in the end. While we consider multiclass classification problems in our project, in the examples below we only consider binary classification for simplicity.

4.6.2 Least Squares Regression

Assume we are handed data $(X_i, Y_i)_{i=1}^n$ with $X_i \in \mathbb{R}^d$ and $Y_i \in \{0,1\}$. We wish to obtain an estimator $\widehat{m}$ which takes on values in $[0,1]$.

As a first variant, we consider the original random planted forest algorithm introduced in Section 4.2 with a slight modification. While the original algorithm is a natural specification of general random planted forests, we do not expect the least squares approach to be optimal for classification. Nonetheless, it is interesting to see how the original algorithm performs in comparison to other variants. Additionally, it serves as an easy example to showcase the intuition of general random planted forests.

In order to define the updating and loss functions, we first define the residuals

$$R_i := Y_i - \sum_{t \in \mathcal{T}} \widehat{m}_t(X_i)$$

for $i = 1, \dots, n$. Then, the updating function is given by

$$g_{\mathrm{L}_2}\big(\mathcal{I}, X_1, \dots, X_d, Y_1, \dots, Y_d, (\widehat{m}_t)_{t \in \mathcal{T}}\big) = \frac{\sum_{X_i \in \mathcal{I}} R_i}{\sum_{X_i \in \mathcal{I}} 1}.$$

Next, given $t \in \mathcal{T}$, $k \in \{1, \dots, d\}$, $j \in \{1, \dots, p_t\}$ and $c \in \mathbb{R}$, the updated residuals are given by

$$\begin{aligned} R_i^{t,k,j,c} := R_i &- \mathbb{1}(X_i \in \mathcal{I}^+_{t,k,j,c}) g_{\mathrm{L}_2}\big(\mathcal{I}^+_{t,k,j,c}, X_1, \dots, X_d, Y_1, \dots, Y_d, (\widehat{m}_t)_{t \in \mathcal{T}}\big) \\ &- \mathbb{1}(X_i \in \mathcal{I}^-_{t,k,j,c}) g_{\mathrm{L}_2}\big(\mathcal{I}^-_{t,k,j,c}, X_1, \dots, X_d, Y_1, \dots, Y_d, (\widehat{m}_t)_{t \in \mathcal{T}}\big), \end{aligned}$$

where $\mathcal{I}^+_{t,k,j,c}, \mathcal{I}^-_{t,k,j,c}$ are the splits of $\mathcal{I}_{t,j}$ at c with respect to the coordinate k. The loss function is then

$$L_{\mathrm{L}_2}(t,k,j,c) := \sum_{i=1}^n \big(R_i^{t,k,j,c}\big)^2.$$

We would obtain the algorithm introduced in Section 4.2 by selecting $\sigma = \mathrm{id}_{\mathbb{R}}$. However, in order to obtain a function that returns values in $[0,1]$, we simply truncate the result at 0 and 1. Thus, the link function is

$$\sigma_{\mathrm{L}_2}(x) := \begin{cases} 0, & \text{for } x \le 0, \\ x, & \text{for } x \in (0,1), \\ 1, & \text{otherwise.} \end{cases}$$

4.6.3 L1 Loss

The setup is very similar to the previous case. Considering the L_1 loss relates to the Gini impurity loss. When considering only one split, both are equivalent.

The updating and link functions are again given by

$$g_{\mathrm{L}_1}\big(\mathcal{I}, X_1, \dots, X_d, Y_1, \dots, Y_d, (\widehat{m}_t)_{t\in\mathcal{T}}\big) = \frac{\sum_{X_i\in\mathcal{I}} R_i}{\sum_{X_i\in\mathcal{I}} 1},$$
$$\sigma_{\mathrm{L}_1}(x) := \begin{cases} 0, & \text{for } x \le 0, \\ x, & \text{for } x \in (0,1), \\ 1, & \text{otherwise.} \end{cases}$$

However, the loss function is now given by

$$L_{\mathrm{L}_1}(t,k,j,c) := \sum_{i=1}^{n} |R_i^{t,k,j,c}|.$$

Thus, we choose the variables t, k, j, s such that the updated estimate minimizes the resulting residuals with respect to the L_1-norm.

4.6.4 Logit Link Function with Maximum Likelihood

We use the logit link function

$$\sigma_{\mathrm{Logit}}(x) = \frac{1}{1+\exp(-x)}.$$

In order to define the updating function, for some fixed $\epsilon \in [0, 0.5)$ we introduce the functions

$$p^{\mathcal{I},\epsilon} = p^{\epsilon}(\mathcal{I}, X_1, \dots, X_d, Y_1, \dots, Y_d) = \min\left\{1-\epsilon, \max\left\{\epsilon, \frac{\sum_{X_i\in\mathcal{I}} Y_i}{\sum_{X_i\in\mathcal{I}} 1}\right\}\right\},$$
$$u^{\mathcal{I}} = u\big(\mathcal{I}, X_1, \dots, X_d, (\widehat{m}_t)_{t\in\mathcal{T}}\big) = \frac{\sum_{X_i\in\mathcal{I}} \widehat{m}(X_i)}{\sum_{X_i\in\mathcal{I}} 1},$$

where $\widehat{m}(X_i) = \sum_{t\in\mathcal{T}} \widehat{m}_t(X_i)$. Thus, the function $p^{\mathcal{I},\epsilon}$ returns the average response in leaf $\mathcal{I}$ which corresponds to an estimator for the probability $\mathbb{P}(Y = 1|X \in \mathcal{I})$, where the function is truncated by ϵ from below and $1-\epsilon$ from above. The updating function, which depends on ϵ, is then given by

$$g^{\epsilon}_{\mathrm{Logit}}\big(\mathcal{I}, X_1, \dots, X_d, Y_1, \dots, Y_d, (\widehat{m}_t)_{t\in\mathcal{T}}\big) = \log\left(\frac{p^{\mathcal{I},\epsilon}}{1-p^{\mathcal{I},\epsilon}}\right) - u^{\mathcal{I}}.$$

If we set $\epsilon = 0$ and $\{Y_i \mid X_i \in \mathcal{I}\} = \{1\}$ holds, the updating function $g^{\epsilon}_{\mathrm{Logit}}$ returns ∞. Thus, if the estimator $\widehat{m}_t(x)$ is finite, it is set to ∞ in $\mathcal{I}$. However, $g^{\epsilon}_{\mathrm{Logit}} = -\infty$ if $\{Y_i \mid X_i \in \mathcal{I}\} = \{0\}$. This may result in adding ∞ and $-\infty$. In our simulations, we use $\epsilon > 0$ so that the calculations are well defined. This also prohibits numerical obstacles which may occur when using high numbers.

For the definition of the loss function, we introduce the updated estimators

$$\begin{aligned} m^{t,k,j,c}(x) = \widehat{m}(x) &+ \mathbb{1}\big(x \in \mathcal{I}^{+}_{t,k,j,c}\big) g^{0}_{\text{Logit}}\big(\mathcal{I}^{+}_{t,k,j,c}, X_1, \dots, X_d, Y_1, \dots, Y_d, (\widehat{m}_t)_{t \in \mathcal{T}}\big) \\ &+ \mathbb{1}\big(x \in \mathcal{I}^{-}_{t,k,j,c}\big) g^{0}_{\text{Logit}}\big(\mathcal{I}^{-}_{t,k,j,c}, X_1, \dots, X_d, Y_1, \dots, Y_d, (\widehat{m}_t)_{t \in \mathcal{T}}\big) \end{aligned}$$

The loss function is given by the maximum likelihood criteria

$$L_{\text{Logit}}(t,k,j,c) := \sum_{i=1}^{n} Y_i \log\big(\sigma(m^{t,k,j,c}(X_i))\big) + (1 - Y_i)\log\big(1 - \sigma(m^{t,k,j,c}(X_i))\big). \tag{4.6.2}$$

Observe that we set $\epsilon = 0$ in the loss function. Thus, we do not truncate the estimators of the conditional probabilities when searching for the next split.

4.6.5 Logit Link Function with Exponential Loss

Next, we consider minimizing the exponential loss function

$$\mathbb{E}[\exp(-\text{sign}(Y_i - 0.5)m(X))].$$

The exponential loss is an upper bound on the misclassification error and has been considered by many others in classification settings [34, 35]. As discussed in [35], it is minimized by

$$m(x) = \frac{1}{2}\log\left(\frac{\mathbb{P}(Y=1|x)}{\mathbb{P}(Y=0|x)}\right).$$

In Result 2, the authors motivate an iterative procedure, where consecutive updates of weights and the estimator are given by

$$\begin{aligned} \widehat{m}^{\text{new}}(x) &= \widehat{m}^{\text{old}}(x) + g(x) = \widehat{m}^{\text{old}}(x) + \frac{1}{2}\log\left(\frac{\widehat{\mathbb{P}}_{w^{\text{old}}}(Y=1|x)}{\widehat{\mathbb{P}}_{w^{\text{old}}}(Y=0|x)}\right), \\ w^{\text{new}}(x,y) &= w^{\text{old}}(x,y)\exp(-\text{sign}(y - 0.5)g(x)), \end{aligned}$$

where $\widehat{\mathrm{P}}_w$ is an estimator of the weighted probability

$$\mathbb{P}_w(Y=1|x) = \mathbb{E}\big[\mathbb{1}_{\{Y=1\}} w(Y,x)\big|x\big].$$

We translate this to a general planted forest setting by defining an updating function g which includes the sample version of $\widehat{P}_w$ in a respective leaf. Additionally, we select the split which minimizes the empirical version of the exponential loss. Note that in contrast to the intuition of [35], the function $\widehat{m}$ is only updated for a subset of the domain in each iteration step.

We now describe the precise choices required for a general random planted forest. Again, we use the logit link function

$$\sigma_{\text{Exp}}(x) = \frac{1}{1+\exp(-x)}.$$

In order to define the updating function, we first introduce the loss at (X_i, Y_i) by

$$R_i := \exp(-0.5\text{sign}(Y_i - 0.5)\widehat{m}(X_i)).$$

For some fixed $\epsilon \in [0, 0.5)$, we define the weighted truncated estimators for the conditional probabilities $\mathbb{P}(Y_i = 1 \mid X_i \in \mathcal{I})$

$$p^{\mathcal{I},\epsilon} = p^{\epsilon}(\mathcal{I}, X_1, \dots, X_d, Y_1, \dots, Y_d) = \min\left\{1-\epsilon, \max\left\{\epsilon, \frac{\sum_{X_i \in \mathcal{I}} Y_i R_i}{\sum_{X_i \in \mathcal{I}} R_i}\right\}\right\}.$$

Observe that the responses Y_i are weighted with their corresponding losses. Thus, if the current estimator correctly predicts Y_i, the response has a relatively low impact on the current split. The updating function is given by

$$g^{\epsilon}_{\text{Exp}}\big(\mathcal{I}, X_1, \dots, X_d, Y_1, \dots, Y_d, (\widehat{m}_t)_{t\in\mathcal{T}}\big) = \log\left(\frac{p^{\mathcal{I},\epsilon}}{1-p^{\mathcal{I},\epsilon}}\right)$$

With a similar argumentation as in Subsection 4.6.4, we select $\epsilon > 0$ in our simulations. For the definition of the loss function, define the updated estimators

$$\begin{aligned} m^{t,k,j,c}(x) = \widehat{m}(x) &+ \mathbb{1}\big(x \in \mathcal{I}^{+}_{t,k,j,c}\big) g^{0}_{\text{Exp}}\big(\mathcal{I}^{+}_{t,k,j,c}, X_1, \dots, X_d, Y_1, \dots, Y_d, (\widehat{m}_t)_{t\in\mathcal{T}}\big) \\ &+ \mathbb{1}\big(x \in \mathcal{I}^{-}_{t,k,j,c}\big) g^{0}_{\text{Exp}}\big(\mathcal{I}^{-}_{t,k,j,c}, X_1, \dots, X_d, Y_1, \dots, Y_d, (\widehat{m}_t)_{t\in\mathcal{T}}\big) \end{aligned}$$

and corresponding residuals

$$R_i^{t,k,j,c} := \exp(-0.5\text{sign}(Y_i - 0.5)m^{t,k,j,c}(X_i)).$$

The loss function is then given by

$$L_{\text{Exp}}(t, k, j, c) = \sum_{i=1}^{n} R_i^{t,k,j,c}. \tag{4.6.3}$$

Note that the loss function is the sum of the single losses when updated with $\epsilon = 0$. In contrast, the weight imposed on any data point X_i is precisely the current value of the loss function when using the actual updating function ($\epsilon > 0$). It seams natural to use the loss of the actual estimate instead of the modified one. However, discarding numerical obstacles, intuitively one would rather use $\epsilon = 0$ for updating. Since numerical obstacles exist, one must deviate from the idealized version. The current version showed the most promising results which are given in Section 4.6.6. The updating procedure of the weights corresponds to the one introduced by

[35], since

$$\begin{aligned} R_i^{\text{new}} &= \exp(-0.5\text{sign}(Y_i - 0.5)\widehat{m}^{\text{new}}(X_i)) \\ &= \exp(-0.5\text{sign}(Y_i - 0.5)m^{\text{old}}(X_i) - \text{sign}(Y_i - 0.5)g(X_i)) \\ &= R_i^{\text{old}} \exp(-\text{sign}(Y_i - 0.5)g(X_i)), \end{aligned}$$

where

$$\begin{aligned} g(x) =& \frac{1}{2}\mathbb{1}\big(x \in \mathcal{I}^+_{t,k,j,c}\big) \log\left(\frac{p^{\mathcal{I}^+_{t,k,j,c},\epsilon}}{1 - p^{\mathcal{I}^+_{t,k,j,c},\epsilon}}\right) \\ &+ \frac{1}{2}\mathbb{1}\big(x \in \mathcal{I}^-_{t,k,j,c}\big) \log\left(\frac{p^{\mathcal{I}^+_{t,k,j,c},\epsilon}}{1 - p^{\mathcal{I}^+_{t,k,j,c},\epsilon}}\right). \end{aligned}$$

4.6.6 General RPF Simulations

In this section, we present first results from a simulation study using real world date. We wish to compare general random planted forests algorithms with state of the art methods. While the simulation study is not in its final state, it still provides some interesting insights.
The study was conducted as follows. We use all data sets from [10] with $n \cdot d \leq 100000$ and no missing values. Here, we present the results from data sets where the response Y_i has a binary outcome. A summary of the data sets is given in Table 8.14 in Subsection 8.1.2. Simulations concerning multiclass classification are provided in figures 8.2, 8.3, and 8.4 in Subsection 8.1.2. For each data set, we consider three algorithms which are given in Table 4.5[5]. Note that all

Table 4.5: Dictionary for the algorithms considered.

Description	short	Code-Reference
A gradient boosting variant	xgboost	[21]
genaral rpf	rpf	Github
Random forest	rf	[130]

algorithms given in Table 4.5 have tuning parameters. These are set via a 5-fold crossvalidation with respect to the AUC, where for each algorithm we minimize over 200 uniformly at random selected parameter combinations from a grid. The gids of parameters are collectively given in Table 8.15 in Subsection 4.6.6. We call the procedure described above the inner crossvalidation. Note that for rpf, the specific algorithm selected is also a tuning parameter. In the results presented here, the possible algorithms include `Loss`=L1 corresponding to the algorithm given in Subsection 4.6.3 as well as `Loss`=exponential corresponding to the algorithm given in Subsection 4.6.5. Additionally, for rpf and xgboost, we also consider a variant in which we bound the maximum order of interaction of $\widehat{m}$ by two. Thus, the resulting estimators are easily visualisable. To obtain figures 4.6 and 4.7 we obtain 10 estimates for each algorithm by applying the respective estimation procedure (including the inner crossvalidation) to one of 10 crossvlaidation sets of

[5]Some codes are available on GitHub: `https://github.com/PlantedML/randomPlantedForest`.

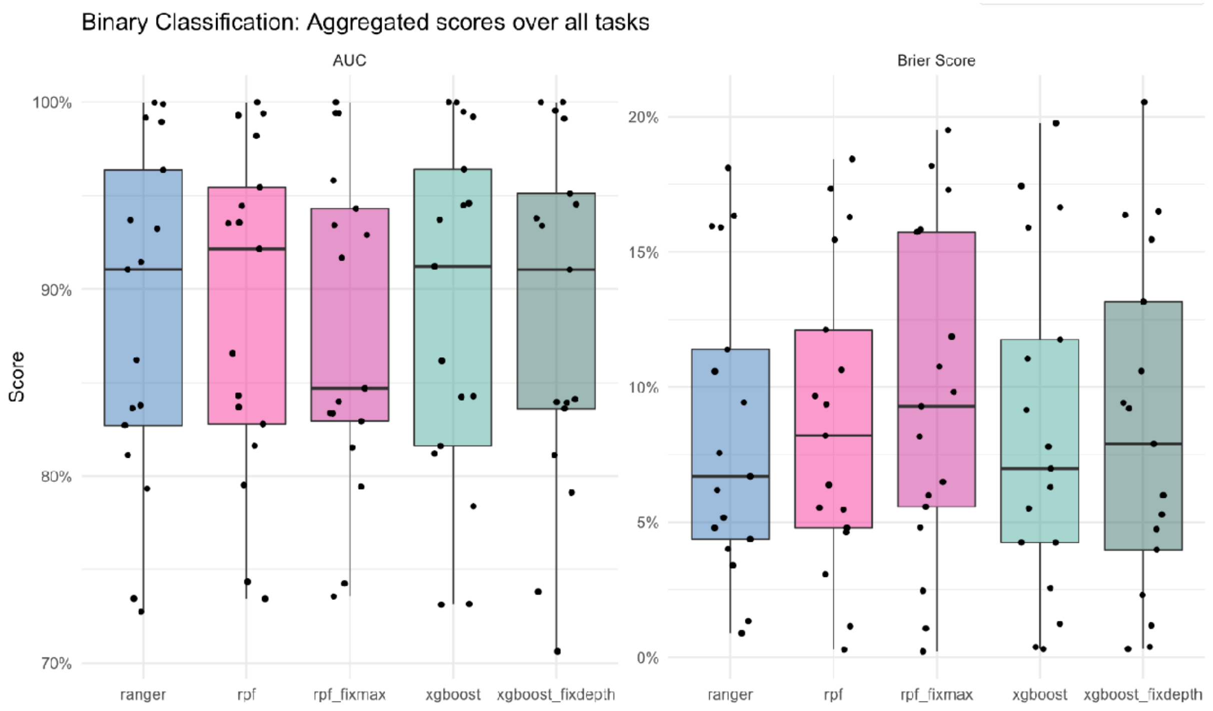


Figure 4.5: Summary of binary classification results over all 17 data sets. Each node corresponds to the AUC (left) or Brier Score (right) of one distinct data set. Lower and upper bounds of boxplots correspond to 25% and 75% quantiles respectively. Black lines correspond to the mean.

the original data. Each box plot is given by a lower bound corresponding to the 25% empirical quantile, an upper bound corresponding to the 75% quantile and a central bar corresponding to the median. We summerize the results given in figures 4.6 and 4.7 in Figure 4.5. Here, each node corresponds to the median of the respective estimator obtained for a specific data set. Since there are 17 data sets, Figure 4.5 has 17 nodes for each algorithm.

From the summary in Figure 4.5 we can see that the rpf algorithm on average can keep up with state of the art methods. In figures 4.6 and 4.7 we observe that in some cases, rpf outperforms xgboost as well as ranger. However, it sometimes is outperformed. When bounding the maximum order of interaction, the algorithm falls off in some cases. However, one can observe that for most data sets, the data can be well approximated by low dimensional structures. Additionally, the results for AUC are slightly better than the results for the Brier score.
While the results presented here are interesting, there are a few improvements which we are currently working on. First of all, regarding Figure 8.1.2, we can see that in two cases, namely tic-tac-toe and phoneme, `nsplits` $\sim$ 50 for most folds. This suggests that the optimal value for `nsplits` exceeds the value 50. This may be the reason why rpf is clearly worse than the other algorithms in these cases. We are currently testing which upper bound to set for `nsplits`. Furthermore, the parameters are tuned with respect to the AUC, not the Brier score. This may explain why the algorithm performs worse when considering this Scoring rule. This also

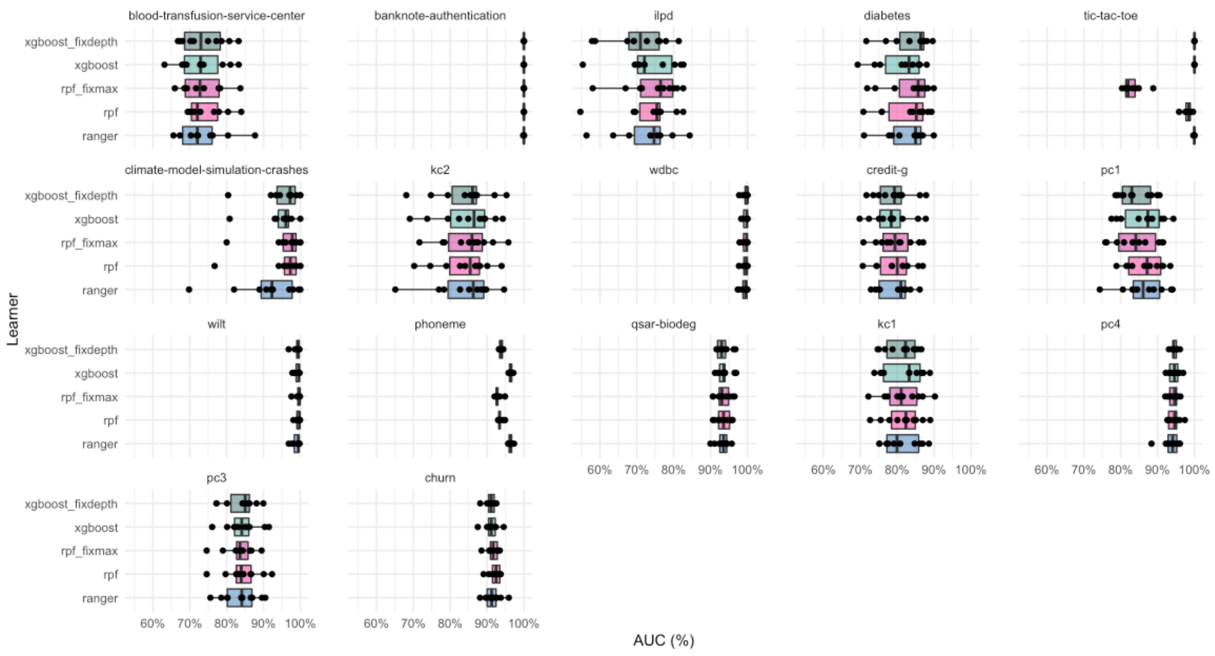


Figure 4.6: AUC results of each of the 17 data sets used for binary classification. Each plot contains 10 estimates obtained by splitting the data into test data and training data with a 10 fold crossvalidation. Each node corresponds to one of the 10 outer crossvalidation estimates. Lower and upper bounds of boxplots correspond to 25% and 75% quantiles respectively. Black lines correspond to the median.

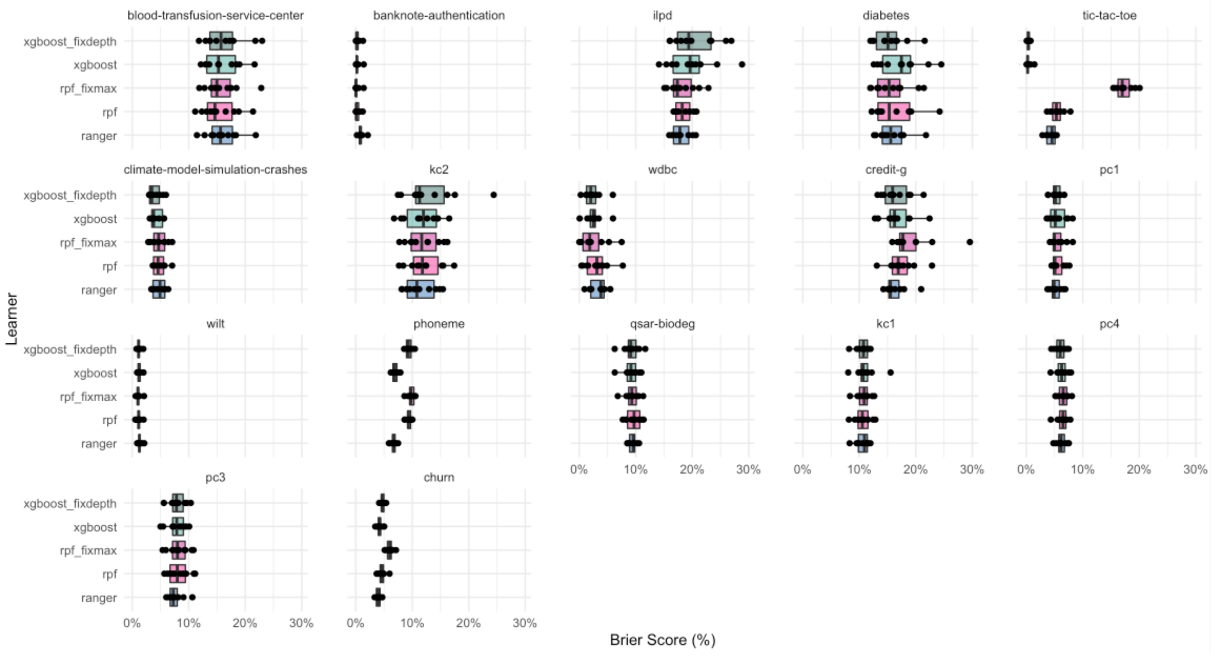


Figure 4.7: Brier scores of each of the 17 data sets used for binary classification. Each plot contains 10 estimates obtained by splitting the data into test data and training data with a 10 fold crossvalidation. Each node corresponds to one of the 10 outer crossvalidation estimates. Lower and upper bounds of boxplots correspond to 25% and 75% quantiles respectively. Black lines correspond to the median.

suggests that when training rpf with respect to the AUC, the resulting estimator does not well approximate probabilities, but only tendencies of the classification problem. Lastly, we use `Loss` which alters the specific algorithm used as a tuning parameter. Here, we only considered the algorithms explained in subsections 4.6.3 and 4.6.5. However, the algorithms given in subsections 4.6.2 and 4.6.4 may also be interesting for consideration.
Summarising, in this first simulation study we observe that the rpf algorithm can keep up with state of the art methods in binary classification problems. In Subsection 4.6.6, we see that the same holds for multiclass classification. However, this study can still be improved.

4.7 Conclusion

We introduce a new tree based prediction method coined random planted forest. While fully flexible, it follows a structured path by growing trees in a family simultaneously which approximate terms of a functional decomposition. A first simulation study provided in this thesis shows promising results. Rpf seems to be able to detect both jumps in the regression function as well as interactions between predictors. Especially in sparse settings, rpf demonstrated an unmatched combination of accuracy and flexibility under easily interpretable models. For classification problems, we found that rpf keeps up with state of the art methods. Rpf predictions are easily visualisable if the maximal order of interaction of the estimator is bounded by either one or two.

5 Unifying Local and Global Model Explanations by Functional Decomposition of Low Dimensional Structures

This chapter follows the paper [56]. We included some slight modifications in order to embed it into this thesis.

We consider a global representation of a regression or classification function by decomposing it into the sum of main and interaction components of arbitrary order. We introduce a new marginal identification constraint which defines the decomposition and show that q-interaction SHAP with an interventional value function is the unique solution to that constraint. Here, q denotes the highest order of interaction present in the decomposition. Additionally, under the same marginal identification, partial dependence plots correspond to main effect terms plus intercept. Our result provides a new perspective on SHAP values with various practical and theoretical implications: The decomposition of SHAP values into main and all interaction effects provides a truly global explanation in the sense that each component only depends on the values of its respective features. Furthermore, we show how the identification can have causal applications. In principle, the decomposition can be applied to any machine learning model. However, since the number of possible interactions grows exponentially with the number of features, exact calculation is only feasible for methods that fit low dimensional structures or ensembles of those. We provide an algorithm and efficient implementation that calculates this decomposition for gradient boosted trees (xgboost) and random planted forests. Conducted experiments suggest that our method provides meaningful explanations and reveals interactions of higher orders. Lastly, we investigate further potential of our new insights: By utilizing the truly global explanation, we motivate a new measure of feature importance and provide a method for reducing direct as well as indirect bias by post-hoc component removal.

5.1 Introduction

In the early years of machine learning interpretability research, the focus was mostly on single-value global feature importance methods that assign a single importance value to each feature. More recently, the attention has shifted towards local interpretability methods, which provide explanations for individual observations or predictions. Popular examples of the latter are LIME [107] and SHAP [84, 111]. The major reason for this shift is that local methods provide a more comprehensive picture of model explanations than single-value global methods, most importantly in presence of nonlinear effects and interactions. This, however, neglects the fact that global methods can be more than single-value methods: Ideally, a global method provides useful information about the entire regression or classification function by providing an explanation for each feature and each interaction effect of arbitrary order, relative to the values they take on. As with local methods, this gives us an explanation for each observation. The crucial difference is that two observations which have a set of feature values in common, receive the same explanation for main and interaction effects involving exclusively those features. We call a representation of a function with this property *truly global.* The components of a truly global explanation are not specific to all feature values of an observation but only to the corresponding feature values. This does not only give a more comprehensive picture than local methods but the complete picture. In summary, we distinguish between three properties of an explanation of a function.

- single-value global: Each feature $j \in \{1, \dots, d\}$ receives a single descriptive value $v_j \in \mathbb{R}$ which does not depend on $x \in \mathbb{R}^d$.
- truly global: Each subset of features $S \subseteq \{1, \dots, d\}$ receives a descriptive function $m_S : \mathbb{R}^S \to \mathbb{R}$ which only depends on values $x_S = \{x_k : k \in S\}$ and not on other values $x_{-S} = \{x_j : j \notin S\}$.
- local: Each subset of features $S \subseteq \{1, \dots, d\}$ receives a descriptive function $\phi_S : \mathbb{R}^d \to \mathbb{R}$ which may depend on all values of $x \in \mathbb{R}^d$.

We introduce a truly global explanation procedure by identifying components in a functional decomposition. We show that the proposed explanation is identical to q-interaction SHAP [125], where q corresponds to the maximal order of interaction present in the model to be analyzed. Hence, we provide a new interpretation of SHAP values which is not game-theoretically motivated. [125] argue that it is practically not feasible to calculate q-interaction SHAP exactly because of computational complexity. However, the authors implicitly assume $q = d$, i.e., the highest order of interaction present in the initial estimator is equal to the number of features. We argue that this is not the case for many state-of-the-art machine learning algorithms that only fit low dimensional structures or ensembles of those. We exploit this fact and discuss an implementation that exactly calculates q-interaction-SHAP for tree-based machine learning models. In principle, our results can be applied to any model and our algorithm can be applied to any tree-based model. However, since the number of components grows exponentially with increasing q, exact calculation is only feasible if q is sufficiently small. We provide a fast implementation for *xgboost* [20] and *random planted forest* from Chapter 4.
As a result, one dimensional contributions m_k and two-dimensional contributions m_{jk} are one and two-dimensional real-valued functions that can be plotted. Furthermore, together with higher order contributions they can be used to decompose simple SHAP values into main effects and all involved interaction effects. Additionally, main and interaction components can be summarised into feature importance values. Beyond explaining feature effects, our proposed

decomposition can be used to detect bias in models where LIME and SHAP fail [113] and reduce such bias by removing individual components from the decomposition.

5.1.1 Motivating Example

We give a toy example of how the interplay of correlations and interactions can give rise to misleading SHAP values. Consider the function $m(x_1, x_2) = x_1 + x_2 + 2x_1x_2$. The interventional SHAP value for the first feature is $\phi_1(x_1, x_2) = x_1 - E[X_1] + x_1x_2 - E[X_1X_2] + x_1E[X_2] - x_2E[X_1]$. If the features are standardized, i.e., X_1 and X_2 have mean zero and variance one, the expression reduces to

$$\phi_1(x_1, x_2) = x_1 + x_1x_2 - \text{corr}(X_1, X_2).$$

Hence, e.g., if $\text{corr}(X_1, X_2) = 0.3$, an individual with $x_1 = 1$ and $x_2 = -0.7$ would see a SHAP value of 0 for the first feature:

$$\phi_1(1, -0.7) = 0.$$

This is quite misleading, since clearly x_1 has an effect on the response m that is irrespective of the particular value of x_1. The underlying problem is that locally at $(x_1, x_2) = (1, -0.7)$, the main effect contribution and interaction contribution cancel each other out. Indeed, we see that the SHAP value ϕ_1 can be decomposed into a main effect contribution of x_1, which is $m_1(x_1) = x_1 - 2\text{corr}(X_1, X_2) = 0.4$ and an interaction contribution of $\{x_1, x_2\}$, which is $m_{12}(x_1, x_2) = x_1x_2 + \text{corr}(X_1, X_2) = -0.4$. Figure 5.1 shows SHAP values and the functional decomposition of an *xgboost* model of the function $m(x_1, x_2)$. The SHAP values ϕ_1 and ϕ_2 contain main effect contributions m_1 and m_2 as well as the interaction contribution m_{12}. The functional decomposition separates the contributions m_1, m_2 and m_{12}.
Those familiar with SHAP values may argue that one can detect the non-zero impact of x_1 by plotting ϕ_1 over all instances (see Figure 5.1). This argument has two problems. Firstly, this does not change the misleading local value. Secondly, SHAP values can be quite arbitrary: If two estimators m and $\tilde{m}$ are equal on the support $\text{supp}(X_1, X_2)$, the corresponding SHAP values at $x \in \text{supp}(X_1, X_2)$ are generally not equal. This is because SHAP values are constructed by extrapolating outside the support of the data. [113] has empirically demonstrated how this phenomenon can be exploited to hide the importance of protected features. One could ask for local explanations that do not extrapolate, hoping that this solves the problem. Unfortunately, this is not possible: If explanations are deduced only from the region with data support, those explanations are based on the correlation structure of the features [66]. In particular a feature that has zero effect on the model output can still be assigned a value stemming from a correlated feature [66, 120]. We conclude:

Local explanations that do not explicitly specify all interactions cannot lead to meaningful interpretations in the presence of correlated features.

This is important to remember, noting that interpretation tools are usually used for black-box algorithms with the main purpose being to explain the model well in cases where interactions are present. Intuitively speaking:

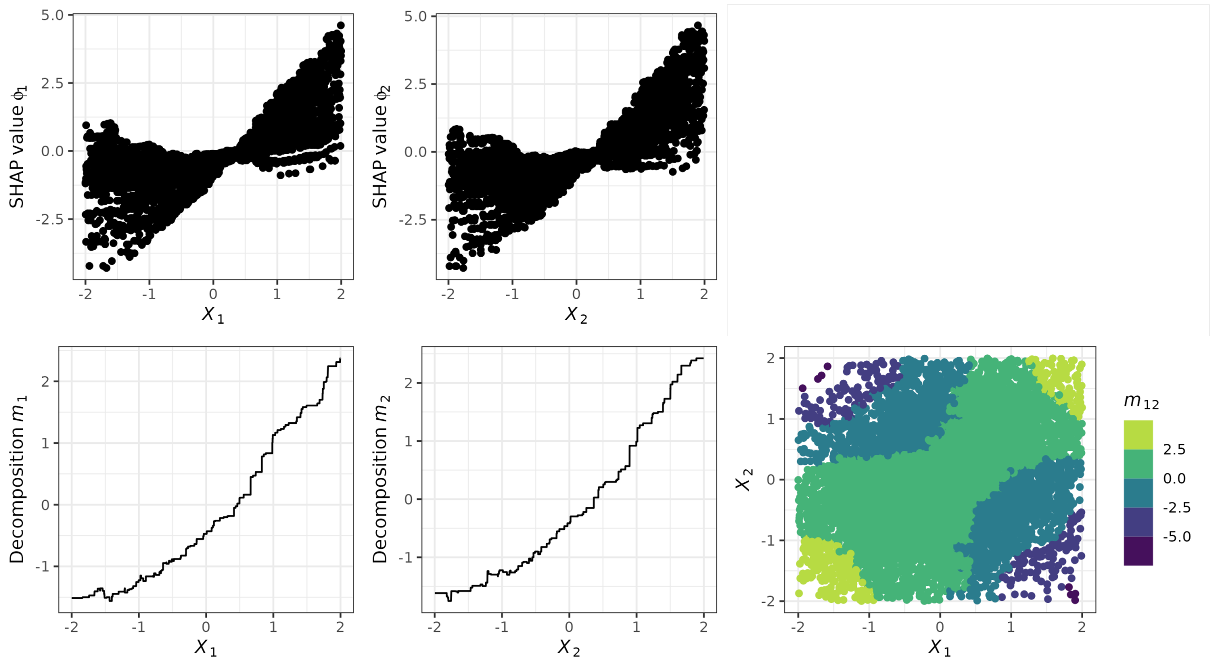


Figure 5.1: Simple example. Given an *xgboost* estimator $\widehat{m}$ estimating the true function $m(x_1, x_2) = x_1 + x_2 + 2x_1x_2$, we calculate SHAP values (top row) and functional decomposition (bottom row).

A local interpretation that explicitly considers all interactions is a truly global interpretation.

Hence the goal is to unify local and global explanations. We emphasize again that in contrast to simple SHAP or l-interaction SHAP ($l < q$), q-interaction SHAP provides a truly global explanation of a trained model.

5.1.2 Related Work

A functional decomposition for truly global interpretation of regression functions was introduced in the statistical literature in [117], and has been further discussed in [18, 60, 80]. These authors considered a different constraint called (generalized) functional ANOVA decomposition. In contrast, the constraint we introduce in this chapter is linked to Shapley values. There is considerable literature on interactions and Shapley values. In cooperative game theory, pairwise player-player interactions were first considered by [101] and later generalized to higher-order interactions by [42]. In the machine learning context, arbitrariness of Shapley values due to interactions and correlations has been discussed in [76, 113, 120], and possible solutions have been proposed in [49, 65, 77, 119, 135]. Recently, [125] introduced interaction SHAP for any given order and proposed an approximation to calculate them. In this chapter, we introduce an identification constraint for a functional decomposition which connects to partial dependent plots [37] and Shapley values with a value function that has recently been coined interventional SHAP [19]. Alternative value functions have been discussed in [39, 133]. There are a variety of methods to obtain single-value global feature importance measures implied by SHAP. These include

[17, 39, 128], among others. Similar to our method suggested in Section 5.4, these measures are weighted averages of local importance values. However, in contrast to our suggestion, most are motivated by additive importance measures [28].
Lastly, recently, two related and relevant works where published. [11] show that for every SHAP-value function there exists a one-to-one correspondence between SHAP values and an identification in a functional decomposition. But they do not provide explicit solutions. [52] describe an identification constraint that connects to observational SHAP. We, however, find an identification constraint that gives a one-to-one correspondence to interventional SHAP and provides a fast implementation for tree-based methods.

5.2 Main Result

Let $(Y_i, X_{i,1}, \dots, X_{i,d})$ be a data set of n i.i.d. observations with $X_{i,k} \in \mathbb{R}$, $i = 1, \dots, n$; $k = 1, \dots, d$. We consider the supervised learning setting

$$E[Y_i|X_i = x] = m(x),$$

where the function m is of interest and Y is a real valued random variable.[1] We assume that a reasonable estimator $\widehat{m}$ of m has been provided.

5.3 Truly Global Interpretation

With increasing dimension it can quickly get very hard, if not impossible, to visualize and thereby comprehend a multivariate function. Hence, a global interpretation of $\widehat{m}$ is arguably only feasible if it is a composition of low-dimensional structures. Let us consider a specific decomposition of a multivariate function into a sum of main effects, bivariate interactions, etc., up to a d-variate interaction term.

$$\begin{aligned}\widehat{m}(x) = \widehat{m}_0 + \sum_{k=1}^{d} \widehat{m}_k(x_k) \\ + \sum_{k<l} \widehat{m}_{kl}(x_k, x_l) + \cdots + \widehat{m}_{1,\dots,d}(x) \\ = \sum_{S \subseteq \{1,\dots,d\}} \widehat{m}_S(x_S).\end{aligned}$$

The heuristic of the decomposition is that if the underlying function $m(x)$ only lives on low-dimensional structures, then m_S should be zero for most feature subsets S and the order of maximal interaction $q = \max\{|S| : m_S \neq 0\}$ should be much smaller than the number of features: $q << d$. This discussion, however, is not very meaningful before one has agreed on an identification; without suitable identification constraints, it is possible to change components on

[1]We use $Y_i \in \mathbb{R}$ for notational convenience. It is straight-forward to extend to binary classification, whereas multiclass classification would require a slightly different procedure.

the right without altering the left hand side. We propose the following identification which we see as reasonable in its own right, but also connects interventional SHAP values [19, 66], partial dependence plots [37] and a causal application, as is explained in the next three subsections.

Marginal identification: For every $S \subseteq \{1, \dots, d\}$,

$$\sum_{T \cap S \neq \emptyset} \int \widehat{m}_T(x_T)\widehat{p}_S(x_S)\mathrm{d}x_S = 0, \tag{5.3.1}$$

where $\widehat{p}_S$ is some estimator of the density p_S of X_S.
The next theorem states existence and uniqueness of a decomposition that satisfies the identification constraint (5.3.1) and describes the solution explicitly.

Theorem 5.3.1. *Given any initial estimator $\widehat{m}^{(0)} = \{\widehat{m}_S^{(0)} | S \subseteq \{1, \dots, d\}\}$, there exists exactly one set of functions $\widehat{m}^* = \{\widehat{m}_S^* | S \subseteq \{1, \dots, d\}\}$ satisfying constraint* (5.3.1) *with $\sum_S \widehat{m}_S^* = \sum_S \widehat{m}_S^{(0)}$. The functions are given by*

$$\widehat{m}_S^*(x_S) = \sum_{T \supseteq S} \sum_{T \setminus S \subseteq U \subseteq T} (-1)^{|S| - |T \setminus U|} \times \int \widehat{m}_T^{(0)}(x_T)\widehat{p}_U(x_U)\mathrm{d}x_U. \tag{5.3.2}$$

In particular $\widehat{m}^*$ does not depend on the particular identification of $\widehat{m}^{(0)}$.

Remark 5.3.2. *In principle, in Theorem 5.3.1, we could have only considered $\widehat{m}_{\{1,\dots,d\}}^{(0)} = \sum_S \widehat{m}_S^{(0)}$ with $\widehat{m}_S^{(0)} = 0$ for $S \neq \{1, \dots, d\}$ as an initial estimator. This would have simplified the notation. However, a main point we aim to make here is that while it may be computational very expensive, if not infeasible, to calculate the marginal identification* (5.3.2) *in general, this is not the case if the initial estimator $\widehat{m}^{(0)}$ is the composition of low dimensional structures. In Subsection 6.2.4, we discuss that, if $\widehat{m}^{(0)}$ can be represented by functions $\widehat{m}_S^{(0)}$, where $\widehat{m}_S^{(0)} = 0$ for $|S| \geq q << d$, then $\widehat{m}^*$ can be calculated reasonably fast. We provide an implementation for xgboost [20] and random planted forest from Chapter 4 that calculates* (5.3.2) *exact and reasonably fast.*

Example 5.3.3. *Consider the setting of our simple example (Section 5.1.1), $m(x_1, x_2) = x_1 + x_2 + 2x_1x_2$, with X_1 and X_2 having mean zero and variance one. If $m(x) = m^*(x)$ and $m^*(x)$ satisfies the marginal identification* (5.3.1)*, then*

$$m^*(x_1, x_2) = m_0^* + m_1^*(x_1) + m_2^*(x_2) + m_{12}^*(x_1, x_2),$$

with

$$\begin{aligned} m_0^* &= 2corr(X_1, X_2) \\ m_1^*(x_1) &= x_1 - 2corr(X_1, X_2) \\ m_2^*(x_2) &= x_2 - 2corr(X_1, X_2) \\ m_{12}^*(x_1, x_2) &= 2x_1x_2 + 2corr(X_1, X_2). \end{aligned}$$

5.3.1 Describing SHAP Values in Terms of our Truly Global Explanation

We now show that there is a direct connection between the truly global explanation described in the previous section and SHAP values. In particular, this connection describes SHAP values uniquely without the use of game theoretically motivated Shapley axioms or a formula running through permutations, where the number of summands grows exponentially with d, see formula (6.2.1) in Subsection 6.2.2. Fix a value $x_0 \in \mathbb{R}^d$. A local approximation at x_0 of the function $\widehat{m}$ is given by

$$\widehat{m}\,(x_0) = \phi_0 + \sum_{k=1}^{d} \phi_k(x_0), \tag{5.3.3}$$

for constants $\phi_0, \phi_1(x_0), \ldots, \phi_d(x_0)$. Similar to the case of truly global explanations, the right hand-side is not identified. Local explanations add constraints to equation (5.3.3) such that $\phi_k(x_0)$ is uniquely identified and best reflects the local contribution of feature k to $\widehat{m}\,(x_0)$. Note that the explanation is local because the explanation for feature k depends on the value of all features $x_0 = x_{0,1}, \ldots, x_{0,d}$ and not on $x_{0,k}$ only.

In what follows, we consider the identification leading to intervention SHAP values [19, 66], i.e., Shapley values with value function

$$v_{x_0}(S) = \int \widehat{m}(x)\widehat{p}_{-S}(x_{-S})dx_{-S}|_{x=x_0}. \tag{5.3.4}$$

See Subsection 6.2.1 for a definition of Shapley values.

Theorem 5.3.4. *If $\widehat{m}$ is decomposed such that* (5.3.1) *is fulfilled, then the interventional SHAP values are weighted averages of the corresponding components, where an interaction component is equally split to all involved features:*

$$\phi_k(x) = \widehat{m}_k^*(x_k) + \frac{1}{2}\sum_j \widehat{m}_{kj}^*(x_{kj}) + \cdots + \frac{1}{d}\widehat{m}_{1,\ldots,d}^*(x_{1,\ldots,d}).$$

Remark 5.3.5. *A crucial point of Theorem 5.3.4 is that the local SHAP values can be described by the components of a truly global explanation. The result is also intriguing since usually the contribution or importance of a single feature in a general truly global representation as in* (4.2.1) *is a complicated interplay between various interactions, see Subsection 6.2.3.*

$$\begin{tikzcd}
\boxed{X_U} \arrow[d] \arrow[dr] & \\
\boxed{X_V} \arrow[r] & \boxed{\text{m(U,V)}}
\end{tikzcd}
\qquad\qquad
\begin{tikzcd}
\boxed{X_U} \arrow[dr] & \\
\boxed{do(X_V = x_V)} \arrow[r] & \boxed{\text{m(U,V)}}
\end{tikzcd}$$

Figure 5.2: Left: Initial causal structure. Right: Causal Structure after removing effect of X_U on X_V.

5.3.2 Describing Partial Dependence Plots in Terms of our Truly Global Explanation

Given an estimator $\widehat{m}$ and a target subset $S \subset \{1, \ldots, d\}$, the partial dependence plot [37], ξ_S, is defined as

$$\xi_S(x_S) = \int \widehat{m}(x) p_{-S}(x_{-S}) \mathrm{d}x_{-S}.$$

It is straight forward to verify that partial dependence plots are linked to a functional decomposition $\{\widehat{m}^*_S\}$ identified via (5.3.1) through

$$\xi_S = \sum_{U \subseteq S} \widehat{m}^*_U.$$

In particular if S is only one feature, i.e., $S = \{k\}$, we have

$$\xi_k(x_k) = m^*_0 + m^*_k(x_k).$$

5.3.3 A Causal Application Stemming from our Truly Global Explanation

Assume U is a set of features that should not have an effect on $\widehat{m}$. For example, one could set $U = \{\text{gender, ethnicity}\}$ in the case of non-discriminatory regulation requirements. Assume $\{1, \ldots, d\}$ is the disjoint union of U and V with a directed acyclic graph structure $X_U \to X_V \to m$, $X_U \to m$; as illustrated in Figure 5.2. Eliminating the causal relationship between X_V and X_U can be achieved via the do-operator, $do(X_V = x_V)$, that removes all edges going into X_V, see Figure 5.2. The function $E[m(X)|\, do(X_V = x_V))$ does not use information contained in X_U; neither directly nor indirectly. Under the assumed causal structure, standard calculations [102] lead to

$$E[m(X)|\, do(X_V = x_V)] = \int m(x) p_U(x_U) dx_U.$$

If $\widehat{m}$ is identified via (5.3.1), then $\tilde{m}(x_{-U}) := \int \widehat{m}(x) \widehat{p}_U(x_U) dx_U$ can be extracted from $\widehat{m}$ by dropping all components that include features in U:

$$\tilde{m}(x_{-U}) = \int \widehat{m}(x) \widehat{p}_U(x_U) dx_U = \sum_{S \subseteq V} \widehat{m}_S(x_S).$$

5.4 Feature Importance

The truly global interpretation also provides a new perspective on feature importance. SHAP value feature importance for feature k is usually given by an empirical version of $E[|\phi_k(x)|]$. By Corollary 5.3.4,

$$E[|\phi_k(x)|] = E\left[\left|\sum_{S:k\in S}\frac{1}{|S|}\widehat{m}^*_S(x_S)\right|\right].$$

In this definition, contributions from various interactions and main effects can cancel each other out, which may not be desirable. An alternative is to consider

$$E\left[\sum_{S:k\in S}\frac{1}{|S|}\left|\widehat{m}^*_S(x_S)\right|,\right]$$

or to extend the definition of feature importance to interactions by defining feature importance as $E\left[|m^*_S(x_S)|\right]$, for a set $S\subseteq\{1,\dots,d\}$.

Example 5.4.1. *Going back to our simple example (Section 5.1.1), where $m(x_1,x_2)=x_1+x_2+2x_1x_2$, SHAP feature importance for feature x_1 is an empirical version of*

$$\begin{aligned}&E\left[\left|X_1-2corr(X_1,X_2)-\frac{1}{2}\{2X_1X_2+2corr(X_1,X_2)\}\right|\right]\\&=E\left[|X_1-X_1X_2-corr(X_1,X_2)\}|\right],\end{aligned}$$

which merges main effect and interaction effect. Alternatively, one may consider

$$E\left[|X_1-2corr(X_1,X_2)|+|X_1X_2+corr(X_1,X_2)\}|\right].$$

5.5 Experiments

We apply our method to several real and simulated datasets to show that the functional decomposition provides additional insights compared to SHAP values and SHAP interaction values. First, we show on real data that a truly global explanation can provide a more comprehensive picture than a local explanation method. Second, we show on real and simulated data that the same holds for the feature importance measure proposed in Section 5.4. Finally, we show that the functional decomposition allows post-hoc removal of features from a model, which can be used to reduce bias of prediction models. We performed all experiments with *xgboost* and *random planted forests*. The results with *xgboost* are presented in Sections 5.5.1-5.5.3, whereas the results with *random planted forests* are in Subsections 8.2.2-8.2.4.

5.5.1 Truly Global Explanations

As an example of a real data application, we apply our method to the *bike sharing* data [33], predicting the number of rented bicycles per day, given seasonal and weather information. Figure 5.3 shows SHAP values, main effects, 2-way interactions and 3-way interactions of the features *hour of the day* (hr, 0-24 full hours), *Temperature* (temp, normalized to 0-1) and *working day* (workingday, 0=no, 1=yes).

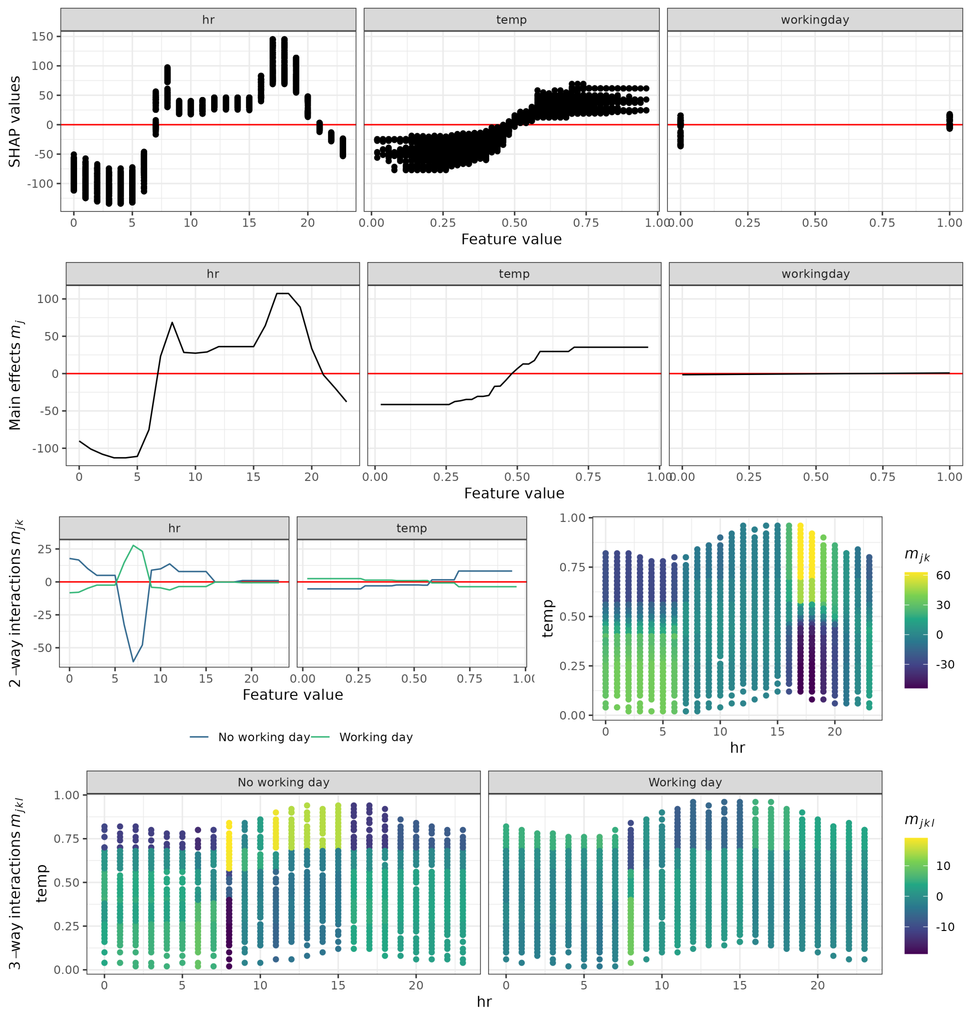


Figure 5.3: Bike sharing example (*xgboost*). SHAP values (top row), main effects (second row), 2-way interactions (third row) and 3-way interactions (bottom row) of the features *hour of the day* (hr, 0-24 full hours), *Temperature* (temp, normalized to 0-1) and *working day* (workingday, 0=no, 1=yes) of the bike sharing data.

In the top row, we see that different SHAP values are observed for the same values of the features and conclude that SHAP values are not sufficient to describe the features' effects on the outcome, due to interactions. In the second row, the main effects from the decomposition show a strong effect of the hour of the day: Many bikes are rented in the typical commute times in the morning and afternoon. We also see a positive effect of the temperature and no main effect of whether or not it is a working day. The 2-way interactions in the third row reveal strong interactions between the hour of the day and working day: On working days, more bikes are rented in the morning and less during the night and around noon. We also see that the temperature has a slightly higher effect on non working days and in the afternoon. In the

bottom row, the 3-way interactions show that interactions between the hour of the day and the temperature are stronger on non working days than on working days.
We conclude that the full functional decomposition provides a more comprehensive picture of the features' effects, compared to usual SHAP value interpretations and 2-way interaction SHAP, as e.g. proposed by [83]. Note that, as described above, our methods do indeed provide the full picture, including all higher-order interactions, whereas Figure 5.3 only shows a subset of these interactions.

5.5.2 Feature Importance

As described in Section 5.4, the functional decomposition can also be used to calculate feature importance. Figure 5.4 shows the feature importance for the function $m(x) = x_1 + x_3 + x_2x_3 - 2x_2x_3x_4$ and the bike sharing data from Section 5.5.1 based on SHAP values and our functional decomposition. For the simple function, the SHAP feature importance identifies x_1 and x_3 as equally important and x_2 and x_4 as less important but it gives no information about interactions. On the other hand, the feature importance based on the functional decomposition shows that x_1 has a strong main effect but no interactions, whereas x_2 and x_4 have only interaction effects but no main effects and x_3 both kinds of effect. Similarly, on the bike sharing data, the hour of the day (feature *hr*) and the temperature (*temp*) have both main and interaction effects, whereas the feature *working day* has 2-way interaction effects but no main effects (compare Figure 5.3). Note that both definitions of feature importance are based on absolute values of SHAP values or components m_S and thus are non-negative, in contrast to other methods of feature importance [17, 98].

5.5.3 Post-Hoc Feature Removal

We show that our method can be used to remove features and all their effects, including interactions, from a model *post-hoc*, i.e. after model fitting. We trained models on simulated data and the *adult* dataset [31]. Both models contained a feature *sex* or *gender*, which is a protected attribute and should not have an effect in fair prediction models [4]. In the simulation, we considered the simplified scenario where we predict a person's salary, based on their sex and weekly working hours. We set the weekly working hours to an average of 40 for men and to 30 for women. Salary was simulated as 1 unit (e.g. thousand Euro per year) per weekly working hour and an additional 20 for males (see Figure 5.2). Thus, men earn more for working longer hours (on average) and for being male per se. The first effect should be kept by a fair machine learning model, whereas the second effect is discriminating women. In the *adult* data, we have the same features *sex* and *hours* but we do not know the causal structure.
Figure 5.5 shows the prediction for females and males of the full model, a refitted model without the protected feature *sex* and a decomposed model where the feature *sex* was removed post-hoc. In the simulated data, we see that refitting the model does not change the predictions at all: Because of the high correlation between *sex* and *hours*, the effects of *sex* cannot be removed by not considering the feature in the model. Our decomposition on the other hand allows us to

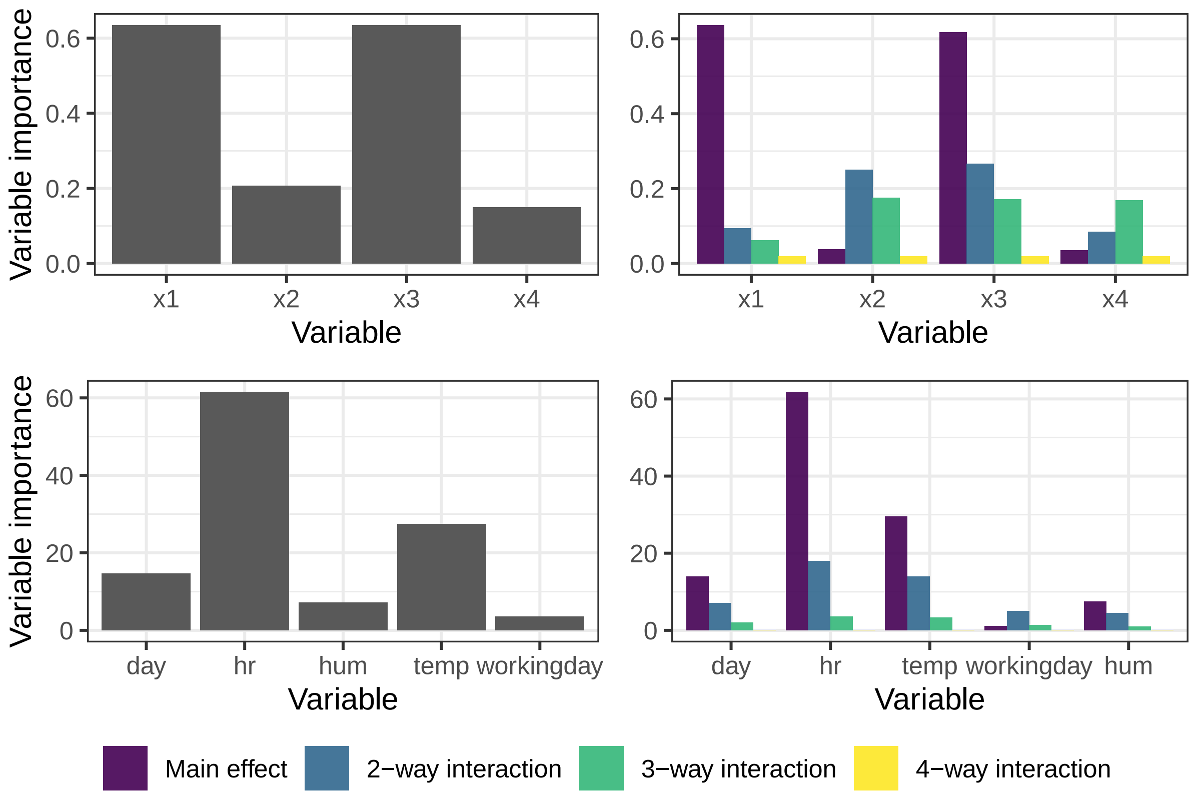
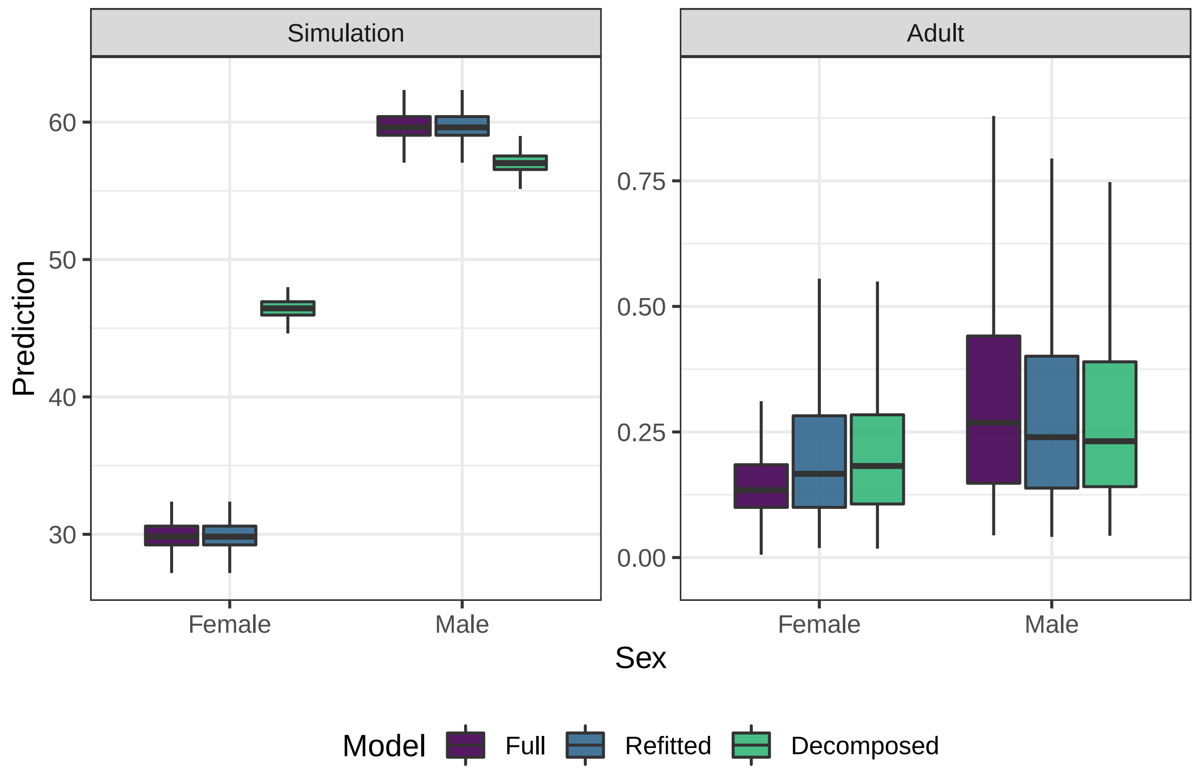


Figure 5.4: Feature importance (*xgboost*) for the function $m(x) = x_1 + x_3 + x_2x_3 - 2x_2x_3x_4$ (top row) and the bike sharing data from Section 5.5.1 (bottom row) based on SHAP values (left column) and our functional decomposition separately for main effects and interactions of different orders (right column).

remove the (unwanted) direct effect of *sex* while keeping the (wanted) indirect effect through *hours*. On the *adult* data, we see a similar difference, but less pronounced.

5.6 Concluding Remarks and Limitations

We have introduced a way to turn local Shapley values into a truly global explanation using a functional decomposition. The explanation has a causal application under the DAG structure given in Figure 5.2. This causal structure might be quite realistic in many fairness considerations, but the true causal structure is generally unknown. In this respect, it would be interesting to look into other causal structures that could motivate different identification constraints, which may connect to other local explanations than interventional SHAP. Indeed, [11] show that every SHAP-value functions corresponds to a specific identification. Also, while our suggestions for feature importance measures paint a more precise picture in many cases, it is not directly motivated by a theoretical constraint, such as usual additive importance measures. It will require more research to back these ideas by theory. Another point not considered in here is the difference between the estimate $\widehat{m}$ and a potential true function m. In particular, it is not clear if a method that estimates m well is also a good estimator for a selection of components m_S. This



Setting	Median difference		
	Full	Refitted	Decomposed
Simulation	29.79	29.79	10.57
Adult	0.13	0.07	0.05

Figure 5.5: Post-hoc feature removal (*xgboost*). Predictions in a simulation (left) and the *adult* dataset for males and females of the full model, a refitted model without the protected feature *sex* and a decomposed model where the feature *sex* was removed post-hoc. The table below shows the median differences between females and males for the three models.

discussion is related to work done in double/debiased machine learning [23]. Moving forward, it could be interesting to modify out of the box machine learning algorithms to specifically learn the low dimensional structures well.

Ethical implications Generally, explaining prediction models can help to reduce bias or discrimination. Specifically, our methods can be used to reveal higher-order interactions with protected attributes and by that detect bias and reduce such bias by post-hoc feature removal (see Section 5.5.3). However, there is more to *fair machine learning* than removing effects of protected attributes [see e.g. 4] and, as shown by [113], machine learning explanation methods are not immune to (adversarial) attacks. Thus, results should be interpreted with care.

6 Additional Remarks

6.1 Random Planted Forest: A Directly Interpretable Tree Ensemble

In this section, we add some technical details which improve the understanding of the random planted forest algorithm. We first give a short overview over the algorithm in the additive case, i.e. if $r = 1$. The simplification is easier to understand and is of particular interest in our simulation studies in Section 4.4. Next, we include the discussion of an identification constraint for the functional decomposition, which is important for plotting the components. Lastly, we add some additional remarks.

6.1.1 Additive Random Planted Forests

In this section, we explain the rpf algorithm in the additive case. Thus, we assume that m has an additive structure

$$m(x) = m_\emptyset + m_1(x_1) + \cdots + m_d(x_d).$$

The algorithm then simplifies in the following way. The algorithm now grows a tree for each coordinate $k \in \{1, \ldots, d\}$. As discussed above, the tree $t = \emptyset$ never grows during the algorithm. Thus, in every step, a coordinate k is chosen for which the tree k is grown. Furthermore, the set V_r introduced in Subsection 4.2.3 reduces to

$$V = \big\{(k, \{k\}) \;\big|\; k \in \{1, \ldots, d\}\big\}.$$

In particular, it does not depend on the current state of the family of planted trees. Noting that $|V| = d$, the value $\texttt{m\_try} := \lceil \texttt{t\_try} \cdot d \rceil$ is constant throughout the algorithm. The parameter `t_try` or equivalently `m_try` act exactly the same as the parameter `m_try` in Breimans implementation of random forests. In this case, the algorithms mainly differ due to the fact that instead of growing a single tree including all coordinates, the rpf algorithm grows a tree for each coordinate separately.

An illustration of the construction of a family of additive planted trees is given in Figure 6.1. Observe that in this case the leaves of a tree form a partition of $\mathbb{R}$. Each data point is contained in exactly one leaf of each tree. Note that the latter two statements do not hold in general in the additive case.

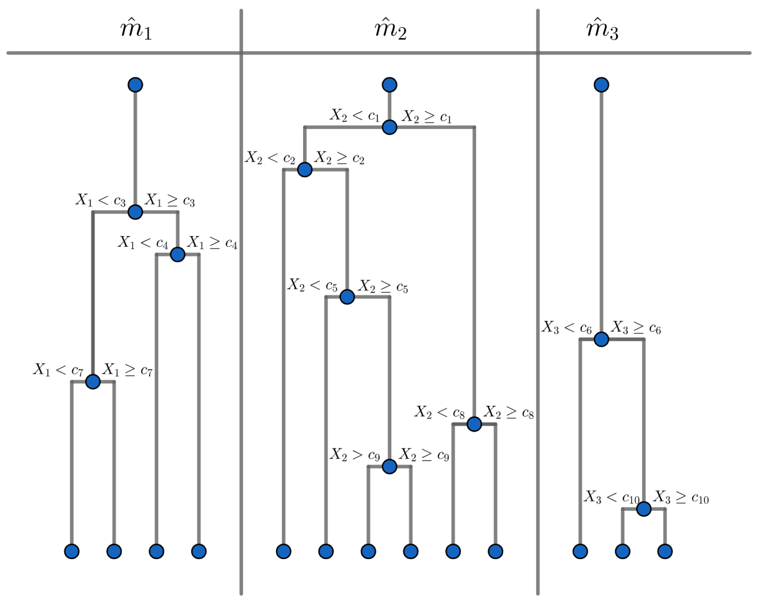


Figure 6.1: Illustration of an additive family of planted trees. Trees grow simultaneously and the height of an edge indicates when the split occurred.

6.1.2 Identification Constraint

If $r \in \{1, 2\}$, the components can be visualised easily. In order to reasonably compare plots, we need a condition that ensures uniqueness of the functional decomposition. We assume that for every $u \subseteq \{1, \ldots, d\}$ and $k \in u$,

$$\int m_u\left(x_u\right) \int w(x) \mathrm{d}x_{-u} \, \mathrm{d}x_k = 0, \tag{6.1.1}$$

for some weight function w. With this constrained the decomposition is known as generalized functional ANOVA. Observe that this is not an additional assumption on the overall function m. The constraint ensures that the functional decomposition is unique for non-degenerated cases. One possibility is $w(x) = \prod_k \widehat{p}_k(x_k)$, where $\widehat{p}_k$ is an estimate of the marginal density of the design density of X. The resulting plots are known as partial dependence plots [37]. With the burden of extra computational cost, one can choose w as an estimate of the full design density, see also [80]. Compared to the previous example, a computationally more efficient constraint has recently been proposed in [2]. Note that we could also use the constraint proposed in Chapter 5. Depending on the viewpoint, different choices for w may be advantageous. Since the precise constraint is not the focus of Chapter 4, we settled for the simple constraint $w \equiv 1$ which suffices in the sense that it allows us to compare plots of different estimators.

Note that while we obtain a tree for every component in the ANOVA decomposition (except the constant), in general, these estimators do not satisfy the constraint (6.1.1). Thus, in order to obtain suitable estimators for the components, we must normalize the estimated components without changing the overall estimator. This can be achieved by using an algorithm similar to the purification algorithm given in [80].

6.1.3 Additional Remarks

This subsection contains various remarks to further enhance the understanding of the algorithm.

As for the backfitting algorithm [14], updating the estimator in each iteration step is important, especially when the predictors are correlated. We also considered alternative updating procedures such as updating all leaves in the tree where the split occurs. While this may be worth some consideration, we observed the best results with our current version. Secondly, we impose 3 conditions that must be satisfied for a split to viable. The first condition (C_1) is obviously required, since we need a leaf to split. The second condition is necessary in order to assure that the resulting trees are elements of T_r - we do not want leaves of trees to depend on more than r coordinates. We impose (C_3) so that each leaf contains at least 1 observation. Also, note that the tree $t = \emptyset$ never grows during the algorithm. Rather, it functions as the root leaf of the trees t with $|t| = 1$. This implies $\widehat{m}_\emptyset = 0$ throughout the algorithm.
Besides interpretability, advantages of restricting the order of approximation in the ANOVA expansion (4.2.1) are potentially faster convergence rates as well as the possibility of using more constrained estimators. In the simulation study in Section 4.4, we find that using an unconstrained rpf works surprisingly well, even if the data generating regression function is additive, i.e. a first order approximation is exact. A constrained version slightly improves the results. The flexibility, however, comes with the cost of reduced interpretability.

6.2 Unifying Local and Global Model Explanations by Functional Decomposition of Low Dimensional Structures

6.2.1 SHAP Values

Consider a value function v_{x_0} that assigns a real value $v_{x_0}(S)$ to each subset $S \subseteq \{1, \dots d\}$. Shapley axioms provide a unique solution under the four axioms efficiency, symmetry, dummy and additivity [111], see Subsection 6.2.2. Defining $\Delta_v(k, S) = v(S \cup k) - v(S)$, the Shapley values are

$$\begin{aligned}\phi_k &= \frac{1}{d!} \sum_{\pi \in \Pi_d} \Delta_v \left(k, \{\pi(1), \dots, \pi(k-1)\}\right) &\quad (6.2.1)\\ &= \frac{1}{d!} \sum_{S \subseteq \{1,\dots,d\} \setminus \{k\}} |S|!(d - |S| - 1)! \Delta_v(k, S), &\quad (6.2.2)\end{aligned}$$

where Π_d is the set of permutations of $\{1, \dots, d\}$. We follow [66] and define SHAP values as Shapley values with the value function

$$v_{x_0}(S) = \int \widehat{m}(x) \widehat{p}_{-S}(x_{-S}) dx_{-S}|_{x=x_0}, \quad (6.2.3)$$

which is also the version implemented in TreeSHAP [83].

6.2.2 Shapley Axioms

Given a function m, a point x_0, and a value function v, the Shapley axioms [111] are

- **Efficiency**: $m(x_0) = \phi_0 + \sum_{k=1}^{d} \phi_k(x_0)$.
- **Symmetry**: Fix any $k, l \in \{1, \ldots, d\}, k \neq l$. If $v_{x_0}(S \cup k) = v_{x_0}(S \cup l)$, for all $S \subseteq \{1, \ldots d\} \setminus \{k, l\}$, then $\phi_k(x_0) = \phi_l(x_0)$
- **Dummy** If $v_{x_0}(S \cup k) = v_{x_0}(S)$, for all $S \subseteq \{1, \ldots d\} \setminus \{k\}$, then $\phi_k = 0$
- **Linearity** If $m(x_0) = m^1(x_0) + m^2(x_0)$, then $\phi_k(x_0) = \phi_k^1(x_0) + \phi_k^2(x_0)$, where ϕ^l is the explanation corresponding to the function m^l.

6.2.3 Connecting a General Truly Global Expansion to SHAP Values

If a regression or classification function m is not identified via (5.3.1), then calculating SHAP values from such a decomposition leads to lengthy and non-trivial expressions. Here, we show how the terms up to dimension three in a general non-identified decomposition enter into a SHAP value. The following formula follows from straight forward calculations using (5.3.3).

For $v_x(S) = \int m(x) p_{-S}(x_{-S}) dx_{-S}$, we get

$$\begin{aligned}
\phi_1(x_0) = {} & m_1(x_1) - E[m_1(X_1)] \\
& + \frac{1}{2} \Bigg\{ \sum_{j \neq 1} m_{1j}(x_1, x_j) - E[m_{1j}(X_1, X_j)] \\
& \qquad + \sum_{j \neq 1} E[m_{1j}(x_1, X_j)] - E[m_{1j}(X_1, x_j)] \Bigg\} \\
& + \frac{1}{3} \Bigg\{ \sum_{j,k \neq 1, j<k} m_{1jk}(x_1, x_j, x_k) - E[m_{1jk}(X_1, X_j, X_k)] \\
& \qquad + \sum_{j,k \neq 1, j<k} E[m_{1jk}(x_1, X_j, X_k)] - E[m_{1jk}(X_1, x_j, x_k)] \\
& \qquad + \frac{1}{2} \sum_{j,k \neq 1, j<k} E[m_{1jk}(x_1, X_j, x_k)] - E[m_{1jk}(X_1, x_j, X_k)] \\
& \qquad + \frac{1}{2} \sum_{j,k \neq 1, j<k} E[m_{1jk}(x_1, x_j, X_k)] - E[m_{1jk}(X_1, X_j, x_k)] \Bigg\} \\
& \qquad \qquad \cdots
\end{aligned}$$

6.2.4 Functional Decomposition of SHAP Values from Low-Dimensional Tree Structures

Our proposed decomposition can be calculated from tree-based models by directly applying Theorem 5.3.1. Inspired by [83], we first describe naïve algorithms for *xgboost* and *random planted forest* models and then describe an improved algorithm for *xgboost* that only needs a single recursion through each tree.

Naïve xgboost Algorithm

For all subsets of features $S \subseteq \{1, \dots, d\}$, we calculate the decomposition $\widehat{m}_S(x_i)$ for all observations of interest $x_i \in \mathbf{X}$ recursively for each tree with features T by considering all subsets U, T with $T \setminus S \subseteq U \subseteq T$. In each node of a tree, if the node is a leaf node we return it's prediction (e.g. the mean in CART-like trees). For internal (non-leaf) nodes, the procedure depends on whether the feature used for splitting in the node is in the subset U or not. If the feature is in the subset U, we continue in both the left and right children nodes, each weighted by the coverage, i.e. the proportion of training observations going left and right, respectively. If the feature is not in the subset U, we apply the splitting criterion of the node and continue with the respective node selected by the splitting procedure for observation x_i. See Algorithm 9 for the full algorithm in pseudo code.

Random Planted Forest Algorithm

For the random planted forest algorithm (rpf), we use a different approach. By slightly altering the representation of an rpf in [54], the result of an rpf is given by a set

$$\widehat{m}^{(0)} = \{\widehat{m}^{(0)}_{S,b} | S \subseteq \{1, \dots, d\},\ b \in \{1, \dots, B\}\},$$

where each estimator $\widehat{m}^{(0)}_{S,b}$ can be represented by a finite partition defined by an $|S|$-dimensional grid (leaves) and corresponding values. Thus, we start with

- a grid $G_k = \{x_{k,1}, \dots, x_{k,t_k}\}$ for each coordinate $k \in \{1, \dots, d\}$,
- for each $S \subseteq \{1, \dots, d\}, b \in \{1, \dots, B\}$ an array representing the value of $m^{(0)}_{S,b}(x)$ for each coordinate $x \in \times_{k \in S} G_k$. Here x is considered to be the bottom left corner of a hyperrectangle.

Note that every tree-based algorithm can be described in such a manner. Given an estimator $\widehat{p}_S$ and using this representation, directly calculating (5.3.2) is simple, where for each combination of sets $U, T \subseteq \{1, \dots, d\}$ with $U \subseteq T$, we only need to calculate the term $\int \widehat{m}^{(0)}_{T,b}(x_T)\widehat{p}_U(x_U)\mathrm{d}x_U$ once and then add/subtract it to the correct estimators $\widehat{m}^*_S$. See Algorithm 10 for the full algorithm in pseudo code. For the calculation of an estimator $\widehat{p}_S$ in our simulations we used the following. For each $S \subseteq \{1, \dots, d\}$, let $a_S(x)$ be the number of data points residing in the hyperrectangle with bottom left corner x for each coordinate $x \in \times_{k \in S} G_k$. For $|S|$-dimensional

Algorithm 9 NAÏVE XGBOOST ALGORITHM

```
Procedure: DECOMPOSE($\mathbf{X}$, $\widehat{m}(x)$)
Input: Data $\mathbf{X} \in \mathbb{R}^{n\times d}$, tree-based model $\widehat{m}(x)$ with $B$ trees
Output: Components $\widehat{m}_S(x_i)$ for all $S \subseteq \{1,\dots,d\}$ and $x_i \in \mathbf{X}$
for $i \in 1,\dots,n$ do
    for $S \subseteq \{1,\dots,d\}$ do
        $\widehat{m}_S(x_i) \leftarrow 0$
        for tree $\in \{1,\dots,B\}$ do
            if $T \supseteq S$ then
                for $U : T \setminus S \subseteq U \subseteq T$ do
                    $\widehat{m}_S(x_i) \leftarrow \widehat{m}_S(x_i) + (-1)^{|S|-|T\setminus U|}\text{REC}(\text{tree}, U, x_i, 0)$
Return: $\widehat{m}_S$

Procedure: REC$(\text{tree}, U, x_i, \text{node})$
Input: Tree ID (tree), subset $U$, data point $x_i$, node ID (node)
Output: Coverage-weighted prediction
if ISLEAF(node) then
    Return: PREDICTION(node)
else
    $j \leftarrow$ SPLIT-FEATURE(node)
    if $j \in U$ then
        $C_{\text{left}} \leftarrow$ COVERAGE(left-node)
        $C_{\text{right}} \leftarrow$ COVERAGE(right-node)
        Return: $C_{\text{l}}\text{REC}(\text{tree}, U, x_i, \text{l-node}) + C_{\text{r}}\text{REC}(\text{tree}, U, x_i, \text{r-node})$
    else
        if $x_i^j \leq$ SPLIT-VALUE(node) then
            Return: REC$(\text{tree}, U, x_i, \text{left-node})$
        else
            Return: REC$(\text{tree}, U, x_i, \text{right-node})$
```

Algorithm 10 NAÏVE RPF ALGORITHM

```
Procedure: DECOMPOSE($\mathbf{X}$, $\widehat{m}(x)$)
Input: Data $\mathbf{X} \in \mathbb{R}^{n\times d}$, tree-based model with initial decomposition $\widehat{m}^{(0)}_{S,b}(x)$ for $S \subseteq \{1,\dots,d\}$ and trees $b = \{1,\dots,B\}$, estimator $\widehat{p}_S$
Output: Components $\widehat{m}_S(x_i)$ for all $S \subseteq \{1,\dots,d\}$ and $x_i \in \mathbf{X}$
for $i \in 1,\dots,n$ do
    for $S \subseteq \{1,\dots,d\}$ do
        $\widehat{m}_S(x_i) \leftarrow 0$
    for $b \in \{1,\dots,B\}$ do
        for $T \subseteq \{1,\dots,d\}$ do
            for $U \subseteq T$ do
                $\text{update}_{T,U} \leftarrow \int \widehat{m}^{(0)}_{T,b}(x_{i,T\setminus U}, x_U)\widehat{p}_U(x_U)\mathrm{d}x_U$
                for $S : T \setminus S \subseteq U,\ S \subseteq T$ do
                    $\widehat{m}_S(x_i) \leftarrow \widehat{m}_S(x_i) + (-1)^{|S|-|T\setminus U|}\text{update}_{T,U}$
    $\widehat{m}_S(x_i) \leftarrow \widehat{m}_S(x_i)/B$
Return: $\widehat{m}_S$
```

y we then set

$$\widehat{p}_S(y) = \frac{a_S(x_y)}{\sum_{x \in \times_{k \in S} G_k} a_S(x)} \frac{1}{\mathrm{vol}(x)},$$

where x_y is the coordinate of the bottom left corner of the hyperrectangle which includes y and $\mathrm{vol}(x)$ is the volume of the hyperrectangle corresponding to x. Using this estimator, the updating function in the algorithm simplifies to

$$\mathrm{update}_{T,U} = \sum_{x_U \in \times_{k \in U} G_k} \widehat{m}^{(0)}_{T,b}(x_{i,T \setminus U}, x_U) \widehat{p}_U(x_U).$$

Improved xgboost Algorithm

To improve the algorithm described in Subsection 6.2.4 and Algorithm 9, we pre-calculate the contribution of each tree for all n observations and tree-subsets T in a single recursive procedure by filling an $n \times 2^D$ matrix, where D is the tree depth. In a second step, we just have to sum these contributions with the corresponding sign (see Theorem 5.3.1).

7 Proofs

7.1 Local Linear Smoothing in Additive Models as Data Projection

7.1.1 Projection Operators

In this section we state expressions for the projection operators $\mathcal{P}_0$, $\mathcal{P}_k$, P_k and $\mathcal{P}_{k'}$ $(1 \le k \le d)$ mapping elements of $\mathcal{H}$ to $\mathcal{H}_0$, $\mathcal{H}_k$, $\mathcal{H}_k + \mathcal{H}_0$ and $\mathcal{H}_{k'}$, respectively, see Section 2.2. For an element $f = (f^{i,j})_{i=1,\dots,n;\ j=0,\dots,d}$ the operators $\mathcal{P}_0$, $\mathcal{P}_k$, and P_k $(1 \le k \le d)$ set all components to zero but the components with indices $(i,0), i = 1,\dots,n$. Furthermore, in the case $d < k \le 2d$ only the components with index $(i, k-d), i = 1,\dots,n$ are non-zero. Thus, for the definition of the operators it remains to set

$$(\mathcal{P}_0(f))^{i,0}(x) \quad = \quad \frac{1}{n}\sum_{i=1}^{n}\int_{\mathcal{X}}\{f^{i,0}(u) + \sum_{j=1}^{d} f^{i,j}(x)(X_{ij}-u_j)\}K_h^{X_i}(X_i-u)\mathrm{d}u.$$

For $1 \le k \le d$ it suffices to define $(\mathcal{P}_k(f))^{i,0}(x) = (P_k(f))^{i,0}(x) - (\mathcal{P}_0(f))^{i,0}$ and

$$\begin{aligned}&(P_k(f))^{i,0}(x)\\ &= \frac{1}{\widehat{p}_k(x_k)}\left[\frac{1}{n}\sum_{i=1}^{n}\int_{u\in\mathcal{X}_{-k}(x_k)}\left\{f^{i,0}(u) + \sum_{j=1}^{d} f^{i,j}(u)(X_{ij}-u_j)\right\}\right.\\ &\qquad \left.\times K_h^{X_i}(X_i-u)\mathrm{d}u_{-k}\right],\end{aligned}$$

$$\begin{aligned}&(P_{k'}(f))^{i,0}(x)\\ &= \frac{1}{\widehat{p}_k^{**}(x_k)}\left[\frac{1}{n}\sum_{i=1}^{n}\int_{u\in\mathcal{X}_{-k}(x_k)}\left\{f^{i,0}(u) + \sum_{j=1}^{d} f^{i,j}(u)(X_{ij}-u_j)\right\}\right.\\ &\qquad \left.\times (X_{ik}-x_k)K_h^{X_i}(X_i-u)\mathrm{d}u_{-k}\right].\end{aligned}$$

For the orthogonal projections of functions $m \in \mathcal{H}_{add}$ one can use simplified formulas. In particular, these formulas can be used in our algorithm for updating functions $m \in \mathcal{H}_{add}$. If $m \in \mathcal{H}_{add}$ has components $m_0, \dots, m_d, m_1^{(1)}, \dots, m_d^{(1)}$ the operators P_k and $P_{k'}$ are defined as follows

$$(\mathcal{P}_0(m)(x)))^{i,0} = m_0 + \sum_{j=1}^{d} \int_{\mathcal{X}_j} m_j^{(1)}(u_j)\widehat{p}_j^*(u_j)\mathrm{d}u_j,$$

$$(P_k(m)(x))^{i,0} = m_0 + m_k(x_k) + m_k^{(1)}(x_k)\frac{\widehat{p}_k^*(x_k)}{\widehat{p}_k(x_k)}$$

$$+ \sum_{1\le j\le d, j\neq k} \int_{\mathcal{X}_{-k,j}(x_k)} \left[m_j(u_j)\frac{\widehat{p}_{jk}(u_j,x_k)}{\widehat{p}_k(x_k)} + m_j^{(1)}(u_j)\frac{\widehat{p}_{jk}^*(u_j,x_k)}{\widehat{p}_k(x_k)} \right] \mathrm{d}u_j,$$

as well as

$$(\mathcal{P}_k(m)(x))^{i,0} = m_k(x_k) + m_k^{(1)}(x_k)\frac{\widehat{p}_k^*(x_k)}{\widehat{p}_k(x_k)} - \sum_{1\le j\le d} \int_{\mathcal{X}_j} m_j^{(1)}(u_j)\widehat{p}_j^*(u_j)\mathrm{d}u_j$$

$$+ \sum_{1\le j\le d, j\neq k} \int_{\mathcal{X}_{-k,j}(x_k)} \left[m_j(u_j)\frac{\widehat{p}_{jk}(u_j,x_k)}{\widehat{p}_k(x_k)} + m_j^{(1)}(u_j)\frac{\widehat{p}_{jk}^*(u_j,x_k)}{\widehat{p}_k(x_k)} \right] \mathrm{d}u_j,$$

$$(P_{k'}(m)(x))^{i,0} = m_k^{(1)}(x_k) + (m_0 + m_k(x_k))\frac{\widehat{p}_k^*(x_k)}{\widehat{p}_k^{**}(x_k)}$$

$$+ \sum_{1\le j\le d, j\neq k} \int_{\mathcal{X}_{-k,j}(x_k)} \left[m_j(u_j)\frac{\widehat{p}_{kj}^*(x_k,u_j)}{\widehat{p}_k^{**}(x_k)} + m_j^{(1)}(u_j)\frac{\widehat{p}_{jk}^{**}(u_j,x_k)}{\widehat{p}_k^{**}(x_k)} \right] \mathrm{d}u_j,$$

where for $1 \le j,k \le d$ with $k \neq j$

$$\begin{aligned}
\widehat{p}_{jk}(x_j,x_k) &= \frac{1}{n}\sum_{i=1}^{n} \int_{\mathcal{X}_{-(jk)}(x_j,x_k)} K_h^{X_i}(X_i - x)\mathrm{d}x_{-(jk)}, \\
\widehat{p}_{jk}^*(x_j,x_k) &= \frac{1}{n}\sum_{i=1}^{n} \int_{\mathcal{X}_{-(jk)}(x_j,x_k)} (X_{ij}-u_j)K_h^{X_i}(X_i - x)\mathrm{d}x_{-(jk)}, \\
\widehat{p}_{jk}^{**}(x_j,x_k) &= \frac{1}{n}\sum_{i=1}^{n} \int_{\mathcal{X}_{-(jk)}(x_j,x_k)} (X_{ij}-u_j)(X_{ik}-x_k)K_h^{X_i}(X_i - x)\mathrm{d}x_{-(jk)}
\end{aligned}$$

with $\mathcal{X}_{-(jk)}(x_j,x_k) = \{u \in \mathcal{X} : u_k = x_k, u_j = x_j\}$, $\mathcal{X}_{-k,j}(x_k) = \{u \in \mathcal{X}_j : \text{there exists } v \in \mathcal{X}$ with $v_k = x_k$ and $v_j = u\}$ and $u_{-(jk)}$ denoting the vector $(u_l : l \in \{1,\dots,d\}\backslash\{j,k\})$.

7.1.2 Proofs of Propositions 2.3.4 and 2.3.6

In this section we give proofs for Propositions 2.3.4 and 2.3.6. They are used in Section 2.3 for the discussion of the existence of the smooth backfitting estimator as well as the convergence of an algorithm for its calculation.

Proof of Proposition 2.3.4. (**ii**) $\Rightarrow$ (**i**). Let $g^{(n)} \in L$ be a Cauchy sequence. We must show $\lim_{n\to\infty} g^{(n)} \in L$. By definition of L there exist sequences $g_1^{(n)} \in L_1$ and $g_2^{(n)} \in L_2$ such that $g^{(n)} = g_1^{(n)} + g_2^{(n)}$. With (2.3.3), for $i = 1, 2$ we obtain

$$\|g_i^{(n)} - g_i^{(m)}\| \leq \frac{1}{c}\|g^{(n)} - g^{(m)}\| \to 0.$$

Hence, $g_1^{(n)}$ and $g_2^{(n)}$ are Cauchy sequences. Since L_1 and L_2 are closed their limits are elements of $L_1 \subseteq L$ and $L_2 \subseteq L$, respectively. Thus,

$$\lim_{n\to\infty} g^{(n)} = \lim_{n\to\infty} g_1^{(n)} + g_2^{(n)} \in L.$$

(i) $\Rightarrow$ **(iii)**. We write $\Pi_1(L_2) = \Pi_1$. Since L is closed, it is a Banach space. Using the closed graph theorem, it suffices to show the following: If $g^{(n)} \in L$ and $\Pi_1 g^{(n)} \in L_1$ are converging sequences with limits g, g_1, then $\Pi_1 g = g_1$.

Let $g^{(n)} \in L$ and $\Pi_1 g^{(n)} \in L_1$ be sequences with limits g and g_1, respectively. Write $g^{(n)} = g_1^{(n)} + g_2^{(n)}$. Since

$$\|g_2^{(n)} - g_2^{(m)}\| \leq \|g_1^{(n)} - g_1^{(m)}\| + \|g^{(n)} - g^{(m)}\|$$

$g_2^{(n)}$ is a Cauchy sequence converging to a limit $g_2 \in L_2$. We conclude $g = g_1 + g_2$, meaning $\Pi_1 g = g_1$.

(iii) $\Rightarrow$ **(ii)**. If Π_1 is a bounded operator, then so is Π_2, since $\|g_2\| \leq \|g\| + \|g_1\|$. Denote the corresponding operator norms by C_1 and C_2, respectively. Then

$$\max\{\|g_1\|, \|g_2\|\} \leq \max\{C_1, C_2\}\|g\|$$

which concludes the proof by choosing $c = \frac{1}{\max\{C_1, C_2\}}$.

(iii) $\Leftrightarrow$ **(iv)**. This follows from

$$\|\Pi_1\| = \sup_{g\in L} \frac{\|g_1}{\|}\|g\| = \sup_{g_1\in L_1, g_2\in L_2} \frac{\|g_1\|}{\|g_1 + g_2\|} = \sup_{g_1\in L_1} \frac{\|g_1\|}{\operatorname{dist}(g_1, L_2)} = \frac{1}{\gamma(L_1, L_2)}.$$

□

Lemma 7.1.1. *Let L_1, L_2 be closed subspaces of a Hilbert space. For γ defined as in Proposition 2.3.4 we have*

$$\gamma(L_1, L_2)^2 = 1 - \|\mathcal{P}_2\mathcal{P}_1\|^2.$$

Proof.

$$\begin{aligned}
\gamma(L_1, L_2)^2 &= \inf_{g_1 \in L_1, \|g_1\|=1} \|g_1 - \mathcal{P}_2 g_1\| \\
&= \inf_{g_1 \in L_1, \|g_1\|=1} \langle g_1 - \mathcal{P}_2 g_1, g_1 - \mathcal{P}_2 g_1 \rangle \\
&= \inf_{g_1 \in L_1, \|g_1\|=1} \langle g_1, g_1 \rangle - \langle \mathcal{P}_2 g_1, \mathcal{P}_2 g_1 \rangle \\
&= 1 - \sup_{g_1 \in L_1, \|g_1\|=1} \langle \mathcal{P}_2 g_1, \mathcal{P}_2 g_1 \rangle \\
&= 1 - \sup_{g \in L, \|g\|=1} \langle \mathcal{P}_2 \mathcal{P}_1 g, \mathcal{P}_2 \mathcal{P}_1 g \rangle \\
&= 1 - \|\mathcal{P}_2 \mathcal{P}_1\|^2.
\end{aligned}$$

□

Proof of Proposition 2.3.6. Let $\mathcal{P}_j$ be the orthogonal projection onto L_j. Following Lemma 7.1.1 we have

$$1 - \|\mathcal{P}_2 \mathcal{P}_1\|^2 = \gamma(L_1, L_2)^2.$$

Using Proposition 2.3.4, proving $\|\mathcal{P}_2 \mathcal{P}_1\| < 1$ implies that L is closed. Observe that $\|\mathcal{P}_2 \mathcal{P}_1\| \leq 1$ because for $g \in L$

$$\|\mathcal{P}_j g\|^2 = \langle \mathcal{P}_j g, \mathcal{P}_j g \rangle = \langle g, \mathcal{P}_j g \rangle \leq \|g\| \|\mathcal{P}_j g\|,$$

which yields $\|\mathcal{P}_i\| \leq 1$ for $i = 1, 2$. To show the strict inequality, note that if $\mathcal{P}_2\big|_{L_1}$ is compact, so is $\|\mathcal{P}_2 \mathcal{P}_1\|$ since the composition of two operators is compact if at least one is compact. Thus, for every $\varepsilon > 0$, $\mathcal{P}_2 \mathcal{P}_1$ has at most a finite number of eigenvalues greater than ε. Since 1 is clearly not an eigenvalue, we conclude $\|\mathcal{P}_1 \mathcal{P}_2\| < 1$. □

7.2 Optimal Convergence Rates of Deep Neural Networks in a Classification Setting

7.2.1 General Convergence Results

The proof of Proposition 3.2.1 is similar to the proof of Theorem 2 in [91]. For the sake of completion, we provide the entire argumentation here anyway.

Proof of Proposition 3.2.1. Let $n \geq N_0$. Without loss of generality, we may assume that $\tau_n \leq n^{\frac{1}{\rho + 2\kappa - 1}}$, since otherwise the conditions are also satisfied when using $\bar{\tau}_n = n^{\frac{1}{\rho + 2\kappa - 1}}$. We begin by proving the assertion for the first term. The idea is to bound

$$\mathbb{P}\big(d_{f_{\mathbb{Q}}}(\widehat{G}_n, G^*_{\mathbb{Q}}) > t\tau_n^{-\kappa}\big)$$

for some $t > 0$. First, observe that for any $G \in \mathcal{N}_n$

$$\begin{aligned}
&R_n(G) - R_n(G^*_{\mathbb{Q}}) - d_{f_{\mathbb{Q}}}(G, G^*_{\mathbb{Q}}) \\
&= \frac{1}{n}\sum_{i=1}^n \big(Y_i - 1(X_i \in G)\big)^2 - \frac{1}{n}\sum_{i=1}^n \big(Y_i - 1(X_i \in G^*_{\mathbb{Q}})\big)^2 \\
&\quad - \Big(\mathbb{E}\Big[\big(Y - 1(X \in G)\big)^2\Big] - \mathbb{E}\Big[\big(Y - 1(X \in G^*_{\mathbb{Q}})\big)^2\Big]\Big) \\
&= \frac{1}{n}\sum_{i=1}^n h_G(X_i, Y_i) - \mathbb{E}\big[h_G(X_i, Y_i)\big] =: \frac{1}{n}\sum_{i=1}^n U_i(G)
\end{aligned}$$

holds, where

$$h_G : \mathbb{R}^d \times \{0,1\} \to \mathbb{R},\ h_G(x,y) = \big(y - 1(x \in G)\big)^2 - \big(y - 1(x \in G^*_{\mathbb{Q}})\big)^2.$$

Regarding (iii), for every $n \in \mathbb{N}$ there exists a $G_n \in \mathcal{N}_n$ such that

$$d_{f_{\mathbb{Q}}}(G_n, G^*_{\mathbb{Q}}) \le c_2 \tau_n^{-\kappa}.$$

For $t > 0$, define

$$\Xi_t := \Big\{G \in \mathcal{N}_n \,\Big|\, d_{f_{\mathbb{Q}}}(G, G^*_{\mathbb{Q}}) \ge t\tau_n^{-\kappa}\Big\}.$$

Then, for $t \ge 4c_2$ and $G \in \Xi_t$ we have

$$\frac{1}{2} d_{f_{\mathbb{Q}}}(G, G^*_{\mathbb{Q}}) - d_{f_{\mathbb{Q}}}(G_n, G^*_{\mathbb{Q}}) \ge c_2 \tau_n^{-\kappa}. \tag{7.2.1}$$

Recall that by definition $\widehat{G}_n$ minimizes $R_n(\cdot)$. Therefore, in view of the calculations above, for $t \ge 4c_2$

$$\begin{aligned}
&\mathbb{P}\big(d_{f_{\mathbb{Q}}}(\widehat{G}_n, G^*_{\mathbb{Q}}) > t\tau_n^{-\kappa}\big) \\
&\le \mathbb{P}\big(\exists G \in \Xi_t :\ R_n(G) - R_n(G_n) \le 0\big) \\
&= \mathbb{P}\Big(\exists G \in \Xi_t :\ R_n(G) - R_n(G^*_{\mathbb{Q}}) - \big(R_n(G_n) - R_n(G^*_{\mathbb{Q}}) \le 0\Big) \\
&= \mathbb{P}\Bigg(\exists G \in \Xi_t :\ d_{f_{\mathbb{Q}}}(G, G^*_{\mathbb{Q}}) + \frac{1}{n}\sum_{i=1}^n U_i(G) \\
&\qquad\qquad - d_{f_{\mathbb{Q}}}(G_n, G^*_{\mathbb{Q}}) - \frac{1}{n}\sum_{i=1}^n U_i(G_n) \le 0\Bigg)
\end{aligned}$$

holds. Using inequality (7.2.1) in the third row yields

$$\begin{aligned}
&\mathbb{P}\Bigg(\exists G \in \Xi_t:\ d_{f_{\mathbb{Q}}}(G, G^*_{\mathbb{Q}}) + \frac{1}{n}\sum_{i=1}^n U_i(G) \\
&\qquad\qquad - d_{f_{\mathbb{Q}}}(G_n, G^*_{\mathbb{Q}}) - \frac{1}{n}\sum_{i=1}^n U_i(G_n) \le 0\Bigg) \\
&= \mathbb{P}\Bigg(\exists G \in \Xi_t:\ \Big(\frac{1}{2} d_{f_{\mathbb{Q}}}(G, G^*_{\mathbb{Q}}) + \frac{1}{n}\sum_{i=1}^n U_i(G)\Big) \\
&\qquad\qquad + \Big(\frac{1}{2} d_{f_{\mathbb{Q}}}(G, G^*_{\mathbb{Q}}) - d_{f_{\mathbb{Q}}}(G_n, G^*_{\mathbb{Q}}) - \frac{1}{n}\sum_{i=1}^n U_i(G_n)\Big) \le 0\Bigg) \\
&\le \mathbb{P}\Bigg(\exists G \in \Xi_t:\ \frac{1}{2} d_{f_{\mathbb{Q}}}(G, G^*_{\mathbb{Q}}) + \frac{1}{n}\sum_{i=1}^n U_i(G) \le 0\Bigg) \\
&\qquad\qquad + \mathbb{P}\Bigg(c_2 \tau_n^{-\kappa} - \frac{1}{n}\sum_{i=1}^n U_i(G_n) \le 0\Bigg) \\
&\le \mathbb{P}\Bigg(\exists G \in \Xi_t:\ \frac{1}{n}\sum_{i=1}^n U_i(G) \le -\frac{1}{2} d_{f_{\mathbb{Q}}}(G, G^*_{\mathbb{Q}})\Bigg) \\
&\qquad\qquad + \mathbb{P}\Bigg(c_2 \tau_n^{-\kappa} \le \frac{1}{n}\sum_{i=1}^n U_i(G_n)\Bigg)
\end{aligned}$$

and thus

$$\begin{aligned}
&\mathbb{P}\big(d_{f_{\mathbb{Q}}}(\widehat{G}_n, G^*_{\mathbb{Q}}) > t\tau_n^{-\kappa}\big) \\
&\le \mathbb{P}\Bigg(\exists G \in \Xi_t:\ \frac{1}{n}\sum_{i=1}^n U_i(G) \le -\frac{1}{2} d_{f_{\mathbb{Q}}}(G, G^*_{\mathbb{Q}})\Bigg) \\
&\qquad\qquad + \mathbb{P}\Bigg(c_2 \tau_n^{-\kappa} \le \frac{1}{n}\sum_{i=1}^n U_i(G_n)\Bigg).
\end{aligned}$$

It remains to find upper bounds for the two terms above. In order to bound the first term, note that for $(x, y) \in \mathbb{R}^d \times \{0, 1\}$ and any $G \in \mathcal{N}_n$ we have

$$\begin{aligned}
|h_G(x, y)| &= \begin{cases} \big|1 - 1(x \in G) - \big(1 - 1(x \in G^*_{\mathbb{Q}})\big)\big|, & \text{for } y = 1, \\ \big|1(x \in G) - 1(x \in G^*_{\mathbb{Q}})\big|, & \text{for } y = 0 \end{cases} \\
&= 1\big(x \in G \Delta G^*_{\mathbb{Q}}\big).
\end{aligned}$$

For all $i = 1, \ldots, n$ this implies $|U_i(G)| \le 2$ and

$$\begin{aligned}
\mathbb{E}\big[U_i(G)^2\big] &\le \mathbb{E}\big[h_G(X_i, Y_i)^2\big] = \mathbb{E}\big[1\big(x \in G\Delta G^*_{\mathbb{Q}}\big)\big] \\
&= d_{\Delta}(G, G^*_{\mathbb{Q}}) \le c_1^{-\frac{1}{\kappa}} d_{f_{\mathbb{Q}}}(G, G^*_{\mathbb{Q}})^{\frac{1}{\kappa}}
\end{aligned}$$

where the last inequality follows from (ii). By Bernstein's inequality, for all $a > 0$

$$\mathbb{P}\left(\left|\frac{1}{n}\sum_{i=1}^{n} U_i(G)\right| \geq a\right) \leq 2\exp\left(-\frac{k_1 n a^2}{a + c_1^{-\frac{1}{\kappa}} d_{f_{\mathbb{Q}}}(G, G^*_{\mathbb{Q}})^{\frac{1}{\kappa}}}\right)$$

holds, where $k_1 > 0$ is a constant. By setting $a = \frac{1}{2} d_{f_{\mathbb{Q}}}(G, G^*_{\mathbb{Q}})$ and observing that $d_{f_{\mathbb{Q}}}(G, G^*_{\mathbb{Q}}) \leq 1$, we have

$$\mathbb{P}\left(\left|\frac{1}{n}\sum_{i=1}^{n} U_i(G)\right| \geq \frac{1}{2} d_{f_{\mathbb{Q}}}(G, G^*_{\mathbb{Q}})\right) \leq 2\exp\Big(-k_2 n d_{f,g}(G, G^*_{\mathbb{Q}})^{\frac{2\kappa-1}{\kappa}}\Big)$$

for some constant $k_2 > 0$. Noting that by definition $\tau_n \leq n^{\frac{1}{\rho+2\kappa-1}}$ and $\kappa \geq 1$, by (iv) we have

$$\begin{aligned}
&\mathbb{P}\left(\exists G \in \Xi_t : \left|\frac{1}{n}\sum_{i=1}^{n} U_i(G)\right| \geq \frac{1}{2} d_{f_{\mathbb{Q}}}(G, G^*_{\mathbb{Q}})\right)\\
&\leq 2\exp\big(c_3 n^{\frac{\rho}{\rho+2\kappa-1}}\big)\exp\big(-k_2 n t^{\frac{2\kappa-1}{\kappa}} \tau_n^{1-2\kappa}\big)\\
&\leq 2\exp\big(c_3 n^{\frac{\rho}{\rho+2\kappa-1}}\big)\exp\big(-k_2 t^{\frac{2\kappa-1}{\kappa}} n^{\frac{1-2\kappa}{\rho+2\kappa-1}+1}\big)\\
&\leq 2\exp\Big(\big(c_3 - k_2 t^{\frac{2\kappa-1}{\kappa}}\big) n^{\frac{\rho}{\rho+2\kappa-1}}\Big)\\
&\leq 2\exp\Big(-c_3 \tau_n^{\rho}\Big)
\end{aligned}$$

for all $t \geq \left(\frac{2c_3}{k_2}\right)^{\frac{\kappa}{2\kappa-1}}$. To bound the second term we use Bernstein's inequality with $a = c_2 \tau_n^{-\kappa}$ and receive

$$\begin{aligned}
\mathbb{P}\left(c_2\tau_n^{-\kappa} \leq \frac{1}{n}\sum_{i=1}^{n} U_i(G_n)\right) &\leq \exp\left(-\frac{k_1 n c_2^2 \tau_n^{-2\kappa}}{c_2\tau_n^{-\kappa} + c_1^{-\frac{1}{\kappa}} d_{f_{\mathbb{Q}}}(G^*_{\mathbb{Q}}, G_n)^{\frac{1}{\kappa}}}\right)\\
&\leq \exp\big(-k_3 n \tau_n^{-2\kappa+1}\big)\\
&\leq \exp\big(-k_3 \tau_n^{\rho}\big)
\end{aligned}$$

for some constant $k_3 > 0$. Therefore, for $t \geq \max\left\{4c_2, \left(\frac{2c_3}{k_2}\right)^{\frac{\kappa}{2\kappa-1}}\right\}$ we find an upper bound

$$\begin{aligned}
&\mathbb{P}\big(d_{f_{\mathbb{Q}}}(\widehat{G}_n, G^*_{\mathbb{Q}}) > t\tau_n^{-\kappa}\big)\\
&\leq \mathbb{P}\left(\exists G \in \Xi_t : \frac{1}{n}\sum_{i=1}^{n} U_i(G) \leq -\frac{1}{2} d_{f_{\mathbb{Q}}}(G, G^*_{\mathbb{Q}})\right)\\
&\qquad\qquad + \mathbb{P}\left(c_2\tau_n^{-\kappa} \leq \frac{1}{n}\sum_{i=1}^{n} U_i(G_n)\right)\\
&\leq 2\exp\Big(-c_3\tau_n^{\rho}\Big) + \exp\big(-k_3\tau_n^{\rho}\big).
\end{aligned}$$

Observing that $d_{f_{\mathbb{Q}}}(\widehat{G}_n, G^*_{\mathbb{Q}}) \leq 1$ we conclude

$$\begin{aligned}
&\mathbb{E}\big[d^p_{f_{\mathbb{Q}}}(\widehat{G}_n, G^*_{\mathbb{Q}})\big] \\
&\leq \mathbb{E}\big[1\big(d_{f_{\mathbb{Q}}}(\widehat{G}_n, G^*_{\mathbb{Q}}) > t\tau_n^{-\kappa}\big)\big] + t\tau_n^{-p\kappa}\mathbb{E}\big[1\big(d_{f_{\mathbb{Q}}}(\widehat{G}_n, G^*_{\mathbb{Q}}) \leq t\tau_n^{-\kappa}\big)\big] \\
&\leq 2\exp\big(-c_3\tau_n^{\rho}\big) + \exp\big(-k_3\tau_n^{\rho}\big) + t\tau_n^{-\kappa p}
\end{aligned}$$

and thus

$$\begin{aligned}
&\lim_{n\to\infty}\sup \sup_{\mathbb{Q}\in\mathfrak{Q}} \tau_n^{\kappa p}\mathbb{E}\big[d^p_{f_{\mathbb{Q}}}(\widehat{G}_n, G^*_{\mathbb{Q}})\big] \\
&\leq \lim_{n\to\infty}\sup \tau_n^{\kappa p}\Big(2\exp\big(-c_3\tau_n^{\rho}\big) + \exp\big(-k_3\tau_n^{\rho}\big) + t\tau_n^{-\kappa p}\Big) \\
&< \infty.
\end{aligned}$$

Proving that the second term in the assertion is finite follows directly, since regarding (ii) for all $\mathbb{Q} \in \mathfrak{Q}$ and sets $G \in \mathcal{N}_n$ it holds hat

$$d_{f_{\mathbb{Q}}}(G, G^*_{\mathbb{Q}}) \geq c_1 d^{\kappa}_{\Delta}(G, G^*_{\mathbb{Q}}).$$

□

Proof of Proposition 3.2.2. Let $n \geq N_0$. Without loss of generality, we may assume that $\tau_n \leq n^{\frac{1}{\rho+2\kappa-1}}$, since otherwise the conditions are also satisfied when using $\tau_n = n^{\frac{1}{\rho+2\kappa-1}}$. The idea is to bound

$$\mathbb{P}\big(d_{f_{\mathbb{Q}}}(\widehat{G}_n, G^*_{\mathbb{Q}}) > t\tau_n^{-1}\big)$$

for some $t > 0$. First, observe that for any $G \in \mathcal{N}_n$

$$\begin{aligned}
&R_n(G) - R_n(G^*_{\mathbb{Q}}) - d_{f_{\mathbb{Q}}}(G, G^*_{\mathbb{Q}}) \\
&= \frac{1}{n}\sum_{i=1}^{n}\big(Y_i - 1(X_i \in G)\big)^2 - \frac{1}{n}\sum_{i=1}^{n}\big(Y_i - 1(X_i \in G^*_{\mathbb{Q}})\big)^2 \\
&\quad - \Big(\mathbb{E}\Big[\big(Y - 1(X \in G)\big)^2\Big] - \mathbb{E}\Big[\big(Y - 1(X \in G^*_{\mathbb{Q}})\big)^2\Big]\Big) \\
&= \frac{1}{n}\sum_{i=1}^{n} h_G(X_i, Y_i) - \mathbb{E}\big[h_G(X_i, Y_i)\big] =: \frac{1}{n}\sum_{i=1}^{n} U_i(G)
\end{aligned}$$

holds, where

$$h_G : \mathbb{R}^d \times \{0,1\} \to \mathbb{R},\ h_G(x,y) = \big(y - 1(x \in G)\big)^2 - \big(y - 1(x \in G^*_{\mathbb{Q}})\big)^2.$$

Regarding (ii), for every $n \in \mathbb{N}$ there exists a $G_n \in \mathcal{N}_n$ such that

$$d_{f_{\mathbb{Q}}}(G_n, G^*_{\mathbb{Q}}) \leq c_2\tau_n^{-1}.$$

For $t > 0$, define

$$\Xi_t := \Big\{ G \in \mathcal{N}_n \;\Big|\; d_{f_{\mathbb{Q}}}(G, G^*_{\mathbb{Q}}) \geq t\tau_n^{-1} \Big\}.$$

Then, for $t \geq 4c_2$ and $G \in \Xi_t$ we have

$$\frac{1}{2} d_{f_{\mathbb{Q}}}(G, G^*_{\mathbb{Q}}) - d_{f_{\mathbb{Q}}}(G_n, G^*_{\mathbb{Q}}) \geq c_2 \tau_n^{-1}. \tag{7.2.2}$$

Recall that by definition $\widehat{G}_n$ minimizes $R_n(\cdot)$. Therefore, in view of the calculations above, for $t \geq 4c_2$

$$\begin{aligned}
&\mathbb{P}\big(d_{f_{\mathbb{Q}}}(\widehat{G}_n, G^*_{\mathbb{Q}}) > t\tau_n^{-1}\big)\\
&\leq \mathbb{P}\big(\exists G \in \Xi_t :\; R_n(G) - R_n(G_n) \leq 0\big)\\
&= \mathbb{P}\Big(\exists G \in \Xi_t :\; R_n(G) - R_n(G^*_{\mathbb{Q}}) - \big(R_n(G_n) - R_n(G^*_{\mathbb{Q}}\big) \leq 0\Big)\\
&= \mathbb{P}\Bigg(\exists G \in \Xi_t :\; d_{f_{\mathbb{Q}}}(G, G^*_{\mathbb{Q}}) + \frac{1}{n}\sum_{i=1}^n U_i(G)\\
&\qquad\qquad - d_{f_{\mathbb{Q}}}(G_n, G^*_{\mathbb{Q}}) - \frac{1}{n}\sum_{i=1}^n U_i(G_n) \leq 0\Bigg)
\end{aligned}$$

holds. Using inequality (4.2.4) in the third row yields

$$\begin{aligned}
&\mathbb{P}\Bigg(\exists G \in \Xi_t :\; d_{f_{\mathbb{Q}}}(G, G^*_{\mathbb{Q}}) + \frac{1}{n}\sum_{i=1}^n U_i(G)\\
&\qquad\qquad - d_{f_{\mathbb{Q}}}(G_n, G^*_{\mathbb{Q}}) - \frac{1}{n}\sum_{i=1}^n U_i(G_n) \leq 0\Bigg)\\
&= \mathbb{P}\Bigg(\exists G \in \Xi_t :\; \bigg(\frac{1}{2} d_{f_{\mathbb{Q}}}(G, G^*_{\mathbb{Q}}) + \frac{1}{n}\sum_{i=1}^n U_i(G)\bigg)\\
&\qquad\qquad + \bigg(\frac{1}{2} d_{f_{\mathbb{Q}}}(G, G^*_{\mathbb{Q}}) - d_{f_{\mathbb{Q}}}(G_n, G^*_{\mathbb{Q}}) - \frac{1}{n}\sum_{i=1}^n U_i(G_n)\bigg) \leq 0\Bigg)\\
&\leq \mathbb{P}\Bigg(\exists G \in \Xi_t :\; \frac{1}{2} d_{f_{\mathbb{Q}}}(G, G^*_{\mathbb{Q}}) + \frac{1}{n}\sum_{i=1}^n U_i(G) \leq 0\Bigg)\\
&\qquad\qquad + \mathbb{P}\Bigg(c_2\tau_n^{-1} - \frac{1}{n}\sum_{i=1}^n U_i(G_n) \leq 0\Bigg)\\
&\leq \mathbb{P}\Bigg(\exists G \in \Xi_t :\; \frac{1}{n}\sum_{i=1}^n U_i(G) \leq -\frac{1}{2} d_{f_{\mathbb{Q}}}(G, G^*_{\mathbb{Q}})\Bigg)\\
&\qquad\qquad + \mathbb{P}\Bigg(c_2\tau_n^{-1} \leq \frac{1}{n}\sum_{i=1}^n U_i(G_n)\Bigg)
\end{aligned}$$

and thus

$$\begin{aligned}
&\mathbb{P}\big(d_{f_{\mathbb{Q}}}(\widehat{G}_n, G^*_{\mathbb{Q}}) > t\tau_n^{-1}\big)\\
&\le \mathbb{P}\left(\exists G \in \Xi_t : \ \frac{1}{n}\sum_{i=1}^n U_i(G) \le -\frac{1}{2} d_{f_{\mathbb{Q}}}(G, G^*_{\mathbb{Q}})\right)\\
&\qquad\qquad + \mathbb{P}\left(c_2\tau_n^{-1} \le \frac{1}{n}\sum_{i=1}^n U_i(G_n)\right)
\end{aligned}$$

It remains to find upper bounds for the two terms above. In order to bound the first term, note that for $(x, y) \in \mathbb{R}^d \times \{0, 1\}$ and any $G \in \mathcal{N}_n$ we have

$$\begin{aligned}
|h_G(x,y)| &= \begin{cases} \big|1 - 1(x \in G) - \big(1 - 1(x \in G^*_{\mathbb{Q}})\big)\big|, & \text{for } y = 1,\\ \big|1(x \in G) - 1(x \in G^*_{\mathbb{Q}})\big|, & \text{for } y = 0 \end{cases}\\
&= \begin{cases} \big|1(x \in G^*_{\mathbb{Q}}) - 1(x \in G)\big|, & \text{for } y = 1,\\ \big|1(x \in G) - 1(x \in G^*_{\mathbb{Q}})\big|, & \text{for } y = 0 \end{cases}\\
&= 1\big(x \in G\Delta G^*_{\mathbb{Q}}\big).
\end{aligned}$$

For all $i = 1, \dots, n$ this implies $|U_i(G)| \le 2$ and

$$\mathbb{E}\big[U_i(G)^2\big] \le \mathbb{E}\big[h_G(X_i, Y_i)^2\big] = \mathbb{E}\big[1\big(x \in G\Delta G^*_{\mathbb{Q}}\big)\big] = d_\Delta(G, G^*_{\mathbb{Q}}) \le 1.$$

By Bernstein's inequality, for all $a > 0$

$$\mathbb{P}\left(\left|\frac{1}{n}\sum_{i=1}^n U_i(G)\right| \ge a\right) \le 2\exp\left(-\frac{k_1 n a^2}{a+1}\right)$$

holds, where $k_1 > 0$ is a constant. By setting $a = \frac{1}{2} d_{f_{\mathbb{Q}}}(G, G^*_{\mathbb{Q}})$, we have

$$\mathbb{P}\left(\left|\frac{1}{n}\sum_{i=1}^n U_i(G)\right| \ge \frac{1}{2} d_{f_{\mathbb{Q}}}(G, G^*_{\mathbb{Q}})\right) \le 2\exp\Big(-k_2 n d_{f,g}(G, G^*_{\mathbb{Q}})^2\Big)$$

for some constant $k_2 > 0$. Noting that by definition $\tau_n \le n^{\frac{1}{\rho+2}}$, by (iii) we have

$$\begin{aligned}
&\mathbb{P}\left(\exists G \in \Xi_t : \ \left|\frac{1}{n}\sum_{i=1}^n U_i(G)\right| \ge \frac{1}{2} d_{f_{\mathbb{Q}}}(G, G^*_{\mathbb{Q}})\right)\\
&\quad\le 2\exp\big(c_3 n^{\frac{\rho}{\rho+2}}\big)\exp\big(-k_2 n t^2 \tau_n^{-2}\big)\\
&\quad\le 2\exp\big(c_3 n^{\frac{\rho}{\rho+2}}\big)\exp\big(-k_2 t^2 n^{-\frac{2}{\rho+2}+1}\big)\\
&\quad\le 2\exp\Big(\big(c_3 - k_2 t^2\big) n^{\frac{\rho}{\rho+2}}\Big)\\
&\quad\le 2\exp\Big(-c_3 \tau_n^{\rho}\Big)
\end{aligned}$$

for all $t \geq \sqrt{\frac{2c_3}{k_2}}$. To bound the second term we use Bernstein's inequality with $a = c_2\tau_n^{-1}$ and receive

$$\begin{aligned}
\mathbb{P}\left(c_2\tau_n^{-1} \leq \frac{1}{n}\sum_{i=1}^n U_i(G_n)\right) &\leq \exp\left(-\frac{k_1 n c_2^2 \tau_n^{-2}}{c_2\tau_n^{-1}+1}\right)\\
&\leq \exp\big(-k_3 n \tau_n^{-2}\big)\\
&\leq \exp\big(-k_3 \tau_n^{\rho}\big)
\end{aligned}$$

for a constant $k_3 > 0$. Therefore, for $t \geq \max\left\{4c_2, \sqrt{\frac{2c_3}{k_2}}\right\}$ we find an upper bound

$$\begin{aligned}
&\mathbb{P}\big(d_{f_{\mathbb{Q}}}(\widehat{G}_n, G^*_{\mathbb{Q}}) > t\tau_n^{-1}\big)\\
&\leq \mathbb{P}\left(\exists G \in \Xi_t : \ \frac{1}{n}\sum_{i=1}^n U_i(G) \leq -\frac{1}{2} d_{f_{\mathbb{Q}}}(G, G^*_{\mathbb{Q}})\right)\\
&\qquad\qquad + \mathbb{P}\left(c_2\tau_n^{-1} \leq \frac{1}{n}\sum_{i=1}^n U_i(G_n)\right)\\
&\leq 2\exp\Big(-c_3\tau_n^{\rho}\Big) + \exp\big(-k_3\tau_n^{\rho}\big).
\end{aligned}$$

Observing that $d_{f_{\mathbb{Q}}}(\widehat{G}_n, G^*_{\mathbb{Q}}) \leq 1$, we conclude

$$\begin{aligned}
&\mathbb{E}\big[d^p_{f_{\mathbb{Q}}}(\widehat{G}_n, G^*_{\mathbb{Q}})\big]\\
&\leq \mathbb{E}\big[1\big(d_{f_{\mathbb{Q}}}(\widehat{G}_n, G^*_{\mathbb{Q}}) > t\tau_n^{-1}\big)\big] + t\tau_n^{-p}\mathbb{E}\big[1\big(d_{f_{\mathbb{Q}}}(\widehat{G}_n, G^*_{\mathbb{Q}}) \leq t\tau_n^{-1}\big)\big]\\
&\leq 2\exp\Big(-c_3\tau_n^{\rho}\Big) + \exp\big(-k_3\tau_n^{\rho}\big) + t\tau_n^{-p}
\end{aligned}$$

and thus

$$\begin{aligned}
&\lim_{n\to\infty}\sup\ \sup_{f\in\mathcal{F}}\ \tau_n^p \mathbb{E}\big[d^p_{f_{\mathbb{Q}}}(\widehat{G}_n, G^*_{\mathbb{Q}})\big]\\
&\leq \lim_{n\to\infty}\sup\ \tau_n^p\Big(2\exp\Big(-c_3\tau_n^{\rho}\Big) + \exp\big(-k_3\tau_n^{\rho}\big) + t\tau_n^{-p}\Big)\\
&< \infty.
\end{aligned}$$

□

7.2.2 Convergence Rates for Neural Networks

The first goal of this section is to prove Theorem 3.3.4. We then follow this up by proving Lemma 3.3.5 and Lemma 3.3.8.

Proof of the Main Result

In order to simplify the approximation results below, we introduce a lemma considering the parallelization and concatenation of two networks Φ_1 and Φ_2. Since these results have been shown in many other articles e.g. [104, 109], we omit the proof.

Lemma 7.2.1. *Let $R(\Phi_1) : \mathbb{R}^{d_1} \to \mathbb{R}^{d_3}$ and $R(\Phi_2) : \mathbb{R}^{d_2} \to \mathbb{R}^{d_4}$ be realizations of neural networks with L_1, L_2 layers, sparsity s_1, s_2 and weights in $\mathcal{W}_{c_1}, \mathcal{W}_{c_2}$, respectively.*

- *If $d_4 = d_1$, the concatenation of the functions $R(\Phi_1) \circ R(\Phi_2)$ can be realized by a neural network with $L = L_1 + L_2 + 1$ layers, sparsity $s \leq 2s_1 + 2s_2$ and weights in $\mathcal{W}_{\max\{c_1,c_2\}}$.*
- *If $d_1 = d_2$, the parallelization of the functions $P(R(\Phi_1), R(\Phi_2)) : \mathbb{R}^{d_1} \to \mathbb{R}^{d_3+d_4}$ given by*

  $$P(R(\Phi_1), R(\Phi_2))(x) := (R(\Phi_1)(x), R(\Phi_2)(x))$$

  can be realized by a neural network with $L = \max\{L_1, L_2\}$ layers, sparsity $s \leq s_1 + s_2 + 2dL$ and weights in $\mathcal{W}_{\max\{c_1,c_2\}}$.

Note that we are only using weights $|w| \leq 1$. In order to approximate high numbers, we use the following lemma.

Lemma 7.2.2. *Let $c, M \in \mathbb{N}$. Then there exist neural networks Φ_1, Φ_2 with input dimensions $z_0 = 1$, at most $L = M + 1$ layers, sparsity $s \leq 4M + 1$ and weights in $\mathcal{W}_c$ such that*

$$\begin{aligned} R(\Phi_1)(x) &= 2^M x, \\ R(\Phi_2)(x) &= 2^M \end{aligned}$$

for all $x \in [0, 1]$.

Proof. The network Φ_1 is given by

$$\Phi_1 := \big(W_1, b_1, \ldots, W_{M+1}, b_{M+1}\big)$$

where $W_{M+1} = W_1^T = (1\ 1)$,

$$W_i = \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}$$

for $i = 2, \ldots, M$, $b_i = (0, 0)$ for $i \leq M$ and $b_{M+1} = 0$. The other network is

$$\Phi_2 := (W_0, b_0, \ldots, W_{M+1}, b_{M+1}) = (W_0, b_0) \times \Phi_1$$

where $W_0 = 0$ and $b_0 = 1$.

The layers of both networks are bounded by $M + 1$. Sparsity of both networks can be bounded by

$$s \leq 2 * 2 + 4(M - 1) + 1 = 4M + 1.$$

□

Next, we construct a neural network for each $G \in \mathcal{K}^{\mathcal{F}}_{\mathbb{Q},\beta,B,\epsilon_1,\epsilon_2,r,d}$ which approximates G well with respect to the metric $d_{f_{\mathbb{Q}}}$. The rough idea for the construction of the network is similar to ideas used in [104]. However, the precise construction in order to adapt to the metric in question differs substantially. The proof of the following theorem is one our the main contributions.

Theorem 7.2.3. *Let $\beta \geq 0$, $B, \rho > 0$ and $d \in \mathbb{N}$ with $d \geq 2$. Let $\mathcal{F}$ be a set of functions*

$$\gamma : [0,1]^{d-1} \to \mathbb{R}$$

such that the following holds. There exist $\epsilon_0, C_1, C_2 > 0$ and $C_3, C_4 \in \mathbb{N}$ such that for any $\gamma \in \mathcal{F}$ and any $\epsilon \in (0, \epsilon_0)$ there is a neural network Φ with $L \leq L_0(\epsilon) := C_1 \lceil \log(\epsilon^{-1}) \rceil$ layers, sparsity $s \leq s_0(\epsilon) := C_2 \epsilon^{-\rho} \log(\epsilon^{-1})$ and weights in $\mathcal{W}_c$ with $c = c_0(\epsilon) := C_3 + C_4 \lceil \log(\epsilon^{-1}) \rceil$ such that

$$\|R(\Phi)(x) - f\|_\infty \leq \epsilon.$$

Define $\kappa = 1 + \beta$ and let $\mathfrak{Q}$ be a class of potential joint distributions $\mathbb{Q}$ of (X, Y) such that the following conditions hold.

(a) There is a constant $M > 1$ such that for all $\mathbb{Q} \in \mathfrak{Q}$ the marginal distribution of $\mathbb{Q}_X$ has a Lebesgue density bounded by M.

(b) There are constants $r \in \mathbb{N}$ and $\epsilon_1, \epsilon_2 > 0$ such that for all $\mathbb{Q} \in \mathfrak{Q}$ the bayes rule satisfies $G^*_{\mathbb{Q}} \in \mathcal{K}^{\mathcal{F}}_{\mathbb{Q},\beta,B,\epsilon_1,\epsilon_2,r,d}$.

Let

$$\tau_n := \frac{n^{\frac{1}{2\kappa+\rho-1}}}{\log^{\frac{2}{\rho}}(n)}.$$

Then there exist constants $C_1', C_2' > 0$ and $C_3' \in \mathbb{N}$ such that the set

$$\mathcal{N}_n = \mathcal{N}_{C_1' L_0(\tau_n^{-1}), C_2' s_0(\tau_n^{-1}), C_3' c_0(\tau_n^{-1})}$$

satisfies the following property. There is a constants $c_2 > 0$ and $N_0 \in \mathbb{N}$ such that for all $n \geq N_0$ and $\mathbb{Q} \in \mathfrak{Q}$ there is a $G \in \mathcal{N}_n$ with

$$d_{f_{\mathbb{Q}}}(G, G^*_{\mathbb{Q}}) \leq c_2 \tau_n^{-\kappa}.$$

Proof. Set $\epsilon_0 := \min\{\epsilon_1, \frac{\epsilon_2}{4}\}$. Choose N_0 large enough such that $\tau_{N_0} \geq \epsilon_0^{-1}$. The proof is outlined as follows. We first construct a candidate set G using neural networks. Then, we show that it satisfies the desired properties.

Let $n \geq N_0$, $\mathbb{Q} \in \mathfrak{Q}$ and

$$G^*_{\mathbb{Q}} = H_1 \cup \cdots \cup H_u$$

as in Definition 3.3.3 with $u \leq r$. We begin with the construction of the candidate set G. The idea is to define a network which approximates $G^*_{\mathbb{Q}}$ well on each set H_ν separately. Define

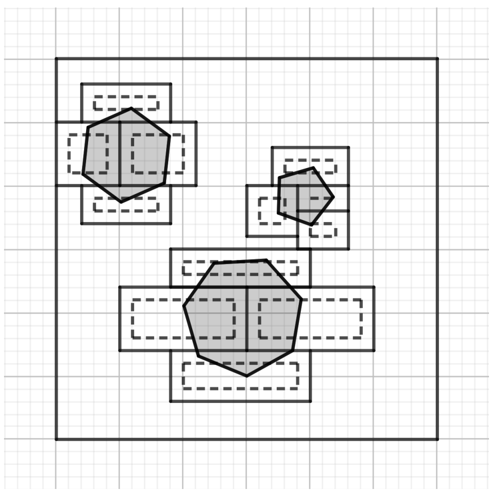


Figure 7.1: The collection of sets $\tilde{D}_\nu$ when considering the example from Figure 3.1. The dotted lines are the boarders of the sets $D_1, \dots, D_{12}$. Note that δ is quite large in this example and observe, that the distance between two sets $\tilde{D}_{\nu_1}, \tilde{D}_{\nu_2}$ is at least δ.

$\iota_\nu, j_\nu, a_i^\nu, b_i^\nu, D_\nu, \gamma_\nu$ and $g_{\nu,x}$ as in Definition 3.3.3. First, for each $\nu = 1, \dots, u$ we consider a set $\tilde{D}_\nu$ with boarders lying on a grid. The advantage of using $\tilde{H}_\nu = \tilde{D}_\nu \cap H_\nu$ instead of H_ν is twofold. On the one hand, the grid and parameters of $\mathcal{N}_n$ are defined such that the boarders of $\tilde{D}_\nu$ can be constructed precisely. On the other, using the grid, two sets $\tilde{H}_{\nu_1}$, $\tilde{H}_{\nu_2}$ have a minimum distance for $\nu_1 \neq \nu_2$, which is important for our method to work. For $\delta > 0$ let

$$h_\delta = \max\left\{h = 2^{-c} \middle| \, h \le \delta, \ c \in \mathbb{N}\right\}.$$

Set $\epsilon := \tau_n^{-1}$. Define $I := \left\{0, h_{\epsilon^\kappa}, 2h_{\epsilon^\kappa} \dots, 1 - h_{\epsilon^\kappa}\right\}$ and let

$$\begin{aligned} \tilde{a}_j^\nu &:= \min\{a \in I \mid a > a_j^\nu\}, \\ \tilde{b}_j^\nu &:= \min\{b \in I \mid b < b_j^\nu\}, \end{aligned}$$

for $\nu = 1, \dots, u$, $j = 1, \dots, d$. Now, set

$$\tilde{D}_\nu := \begin{cases} \prod_{j=1}^d \left[\tilde{a}_j^\nu, \tilde{b}_j^\nu\right], & \text{if } \forall j = 1, \dots, d \ \tilde{a}_j^\nu < \tilde{b}_j, \\ \emptyset, & \text{otherwise.} \end{cases}$$

Note that $\tilde{b}_{j_\nu}^\nu - \tilde{a}_{j_\nu}^\nu \ge 2\epsilon$ for all $\nu = 1, \dots, u$ by the choice of ϵ_0. Figure 7.1 shows the collection of sets $\tilde{D}_\nu$ in the example considered in Figure 3.1. Obviously we have $\tilde{D}_\nu \subseteq D_\nu$. Let

$$\tilde{H}_\nu = \tilde{D}_\nu \cap \{x \in [0,1]^d \mid \iota_\nu x_{j_\nu} \le \gamma_\nu(x_{-j_\nu})\}.$$

The idea is to construct a neural network for every $\nu = 1, \dots, u$ with $\tilde{D}_\nu \neq \emptyset$ which approximates $\tilde{H}_\nu$. We obtain the final neural network by parallelizing and adding up these networks. More specifically, we construct a network that approximates the product of $\mathbb{1}(x \in \tilde{D}_\nu)$ and $\mathbb{1}(\iota_\nu x_{j_\nu} \le$

$\gamma_\nu(x_{-j_\nu}))$. The latter is approximated by a network Φ_{γ_ν} which is the concatenation of a network approximating the heaviside function $\mathbb{1}(x_{j_\nu} > 0)$ and a network approximating

$$\tilde{\gamma}_\nu(x) := \big(x_1, \dots, x_{j_\nu - 1}, \iota_\nu x_{j_\nu} - \gamma_\nu(x_{-j_\nu}), x_{j_\nu+1}, \dots, x_d\big).$$

For $\nu = 1, \dots, u$, from the prerequisites given in the Theorem we obtain a network $\Phi^1_{\gamma_\nu}$ with $L^1_{\gamma_\nu} \le C^1_1 \lceil \log \epsilon^{-1} \rceil$, $s^1_{\gamma_\nu} \le C^1_2 \epsilon^{-\rho} \lceil \log \epsilon^{-1} \rceil$ and weights in $\mathcal{W}_{c^1}$ with $c^1 := C^1_3 + C^1_4 \lceil \log \epsilon^{-1} \rceil$, such that

$$\| R(\Phi^1_{\gamma_\nu}) - \gamma_\nu \|_\infty \le \frac{\epsilon}{4}.$$

For technical reasons, we need to slightly change the realisations in order to handle the behaviour of the approximations at the boarder of $\tilde{D}_\nu$. Define $\Phi^2_{\gamma_\nu}$ by its realization

$$\begin{aligned} R(\Phi^2_{\gamma_\nu})(x) &:= \frac{\tilde{b}^\nu_{j_\nu} + a^{\tilde{\nu}}_{j_\nu}}{2} + \sigma\left(R(\Phi^1_{\gamma_\nu})(x) + h_{\frac{\epsilon}{2}} - \frac{\tilde{b}^\nu_{j_\nu} + a^{\tilde{\nu}}_{j_\nu}}{2} \right) \\ &\qquad - \sigma\left(\frac{\tilde{b}^\nu_{j_\nu} + a^{\tilde{\nu}}_{j_\nu}}{2} - R(\Phi^1_{\gamma_\nu})(x) + h_{\frac{\epsilon}{2}} \right) \\ &= \begin{cases} R(\Phi^1_{\gamma_\nu})(x) + h_{\frac{\epsilon}{2}}, & \text{if } R(\Phi^1_{\gamma_\nu})(x) \ge \frac{\tilde{b}^\nu_{j_\nu} + a^{\tilde{\nu}}_{j_\nu}}{2} + h_{\frac{\epsilon}{2}}, \\ R(\Phi^1_{\gamma_\nu})(x) - h_{\frac{\epsilon}{2}}, & \text{if } R(\Phi^1_{\gamma_\nu})(x) \le \frac{\tilde{b}^\nu_{j_\nu} + a^{\tilde{\nu}}_{j_\nu}}{2} - h_{\frac{\epsilon}{2}}, \\ 2R(\Phi^1_{\gamma_\nu})(x) - \frac{\tilde{b}^\nu_{j_\nu} + a^{\tilde{\nu}}_{j_\nu}}{2}, & \text{otherwise.} \end{cases} \end{aligned}$$

Let $\widehat{\gamma}_\nu := R(\Phi^2_{\gamma_\nu})$. Note that

$$\| \widehat{\gamma}_\nu - \gamma_\nu \|_\infty \le \| R(\Phi^1_{\gamma_\nu}) - \gamma_\nu \|_\infty + h_{\frac{\epsilon}{2}} \le \frac{\epsilon}{4} + \frac{\epsilon}{2} \le \epsilon$$

as well as

$$\begin{aligned} \widehat{\gamma}_\nu(x) \ge \gamma_\nu(x) \text{ for } x : \gamma_\nu(x) \ge \tilde{b}^\nu_{j_\nu}, \\ \widehat{\gamma}_\nu(x) \le \gamma_\nu(x) \text{ for } x : \gamma_\nu(x) \le \tilde{a}^\nu_{j_\nu}. \end{aligned}$$

Using parallelization and concatenation of Lemma 7.2.1, the function

$$R(\Phi^3_{\gamma_\nu})(x) := \big(x_1, \dots, x_{j_\nu - 1}, \iota_\nu x_{j_\nu} - \widehat{\gamma}_\nu(x_{-j_\nu}), x_{j_\nu+1}, \dots, x_d\big)$$

is a realization of a neural network with at most $L^3_{\gamma_\nu} \le L^1_{\gamma_\nu} + 3$ layers, sparsity

$$s^3_{\gamma_\nu} \le 2 s^1_{\gamma_\nu} + 14 + 6d + 2d L^1_{\gamma_\nu},$$

and weights in $\mathcal{W}_{c^3}$ with $c^3 = c^1 + 2$. Now let

$$R(\Phi_H)(x) := \sigma(x_{j_\nu} + 1) - \sigma(x_{j_\nu}) = \begin{cases} 0, & \text{for } x_{j_\nu} \leq -1, \\ x_{j_\nu} + 1, & \text{for } -1 < x_{j_\nu} < 0, \\ 1, & \text{for } x_{j_\nu} \geq 0. \end{cases}$$

Note that $R(\Phi_H)(x) \in (0,1)$ if $x_{j_\nu} \in (-1,0)$. Define $\Phi_{\gamma_\nu} := \Phi_H \circ \Phi^3_{\gamma_\nu}$ as in Lemma 7.2.1 concatenation. The network Φ_{γ_ν} has $L_{\gamma_\nu} \leq C_1^{\gamma_\nu} \lceil \log \epsilon^{-1} \rceil$ layers, sparsity $s_{\gamma_\nu} \leq C_2^{\gamma_\nu} \epsilon^{-\rho} \lceil \log(\epsilon^{-1}) \rceil$ and weights in $\mathcal{W}_{c^{\gamma_\nu}}$ with $c^{\gamma_\nu} := C_3^{\gamma_\nu} + C_4^{\gamma_\nu} \lceil \log(\epsilon^{-1}) \rceil$ for some constants $C_1^{\gamma_\nu}, C_2^{\gamma_\nu} > 0$, $C_3^{\gamma_\nu}, C_4^{\gamma_\nu} \in \mathbb{N}$. Then $R(\Phi_{\gamma_\nu})(x) = 1$ if $\iota_\nu x_{j_\nu} \leq \widehat{\gamma}_\nu(x_{-j_\nu})$ and $0 \leq R(\Phi_{\gamma_\nu})(x) < 1$ otherwise. Next, for $\nu = 1, \dots, u$ with $\tilde{H}_\nu \neq \emptyset$ and $i \in \{1, \dots, d\}$, define the network $\Phi_{\nu,i}$ with realization

$$\begin{aligned} R(\Phi_{\nu,i})(x) &:= 2h_{\epsilon^\kappa}^{-1} \bigg(\sigma\left(x_i - \tilde{a}_i^\nu + \frac{h_{\epsilon^\kappa}}{2} \right) - \sigma\left(x_i - \tilde{a}_i^\nu \right) - \sigma\left(x_i - \tilde{b}_i^\nu \right) \\ &\qquad + \sigma\left(x_i - \tilde{b}_i^\nu - \frac{h_{\epsilon^\kappa}}{2} \right) \bigg) \\ &= \begin{cases} 0, & \text{for } x_i \leq \tilde{a}_i^\nu - \frac{h_{\epsilon^\kappa}}{2}, \\ 2h_{\epsilon^\kappa}^{-1} \left(x_i - \tilde{a}_i^\nu + \frac{h_{\epsilon^\kappa}}{2} \right), & \text{for } \tilde{a}_i^\nu - \frac{h_{\epsilon^\kappa}}{2} < x_i < \tilde{a}_i^\nu, \\ 1, & \text{for } \tilde{a}_i^\nu \leq x_i \leq \tilde{b}_i \nu, \\ 1 - 2h_{\epsilon^\kappa}^{-1}(x_i - \tilde{b}_i^\nu), & \text{for, } \tilde{b}_i^\nu < x_i < \tilde{b}_i^\nu + \frac{h_{\epsilon^\kappa}}{2}, \\ 0, & \text{for } x_i \geq \tilde{b}_i^\nu + \frac{h_{\epsilon^\kappa}}{2}. \end{cases} \end{aligned}$$

Note that $\Phi_{\nu,i}$ is a concatenation of two neural networks, since $2h_{\epsilon^\kappa}^{-1} \geq 1$. By Lemma 7.2.2, we can realize the function $x \mapsto 2h_{\epsilon^\kappa}^{-1} x$ using a neural network Φ_ϵ with $L_\epsilon \leq 1 + \lceil \kappa \rceil c^{\gamma_\nu}$ layers, sparsity $s_\epsilon \leq 4\lceil \kappa \rceil c^{\gamma_\nu} + 5$ and weights in $\mathcal{W}_{c^{\gamma_\nu}}$. Thus, $\Phi_{\nu,i}$ has $L_{\nu_i} \leq 4 + \lceil \kappa \rceil c^{\gamma_\nu}$ layers, sparsity $s_{\nu,i} \leq 32\lceil \kappa \rceil c^{\gamma_\lambda} + 32$ and weights in $\mathcal{W}_{\lceil \kappa \rceil c^{\gamma_\nu}}$. We then define

$$R(\Phi_\nu)(x) := \sigma\bigg(\sum_{i=1}^d R(\Phi_{\nu,i})(x) + R(\Phi_{\gamma_\nu})(x) - d \bigg).$$

For $x \in [0,1]^d$ we have $R(\Phi_\nu)(x) = 1$ if $x \in \tilde{D}_\nu$ and $\iota_\nu x_{j_\nu} \leq \widehat{\gamma}_\nu(x_{-j_\nu})$. Otherwise $0 \leq R(\Phi_\nu(x)) < 1$ holds. Note that by regarding the construction of $\tilde{D}$, we have $R(\Phi_{\nu_1})R(\Phi_{\nu_2}) = 0$ for $\nu_1 \neq \nu_2$. In order to construct the sum, we used a parallelization of the networks $\Phi_{\nu,1}, \dots, \Phi_{\nu,d}$ and Φ_{γ_ν}. Thus, the network Φ_ν has $L_\nu \leq C_1^\nu \lceil \log \epsilon^{-1} \rceil$ layers, sparsity $s_\nu \leq C_2^\nu \epsilon^{-\rho} \lceil \log \epsilon^{-1} \rceil$ and weights in $\mathcal{W}_{c^\nu}$ with $c^\nu = C_3^\nu + C_4^\nu \lceil \log \epsilon^{-1} \rceil$ for some constants $C_1^\nu, C_2^\nu > 0$, $C_3^\nu, C_4^\nu \in \mathbb{N}$[1]. Note that

[1]Note that the notation suggests that the constants differ depending on $\nu = 1, \dots, u$. However, due to the construction, this is not the case. The superscripts in the notations above are given in order to describe where the constant comes from. For our analysis below, it does not make a difference if the constants change with ν or not.

Lemma 7.2.2 was used to construct $d \geq 1$. The realization of the final network is given by

$$R(\Phi)(x) = \sum_{\nu : \tilde{D}_\nu \neq \emptyset} R(\Phi_\nu)(x).$$

Define $G := R(\Phi)^{-1}(1)$. We now verify the desired properties.

We begin by finding constants $C_1', C_2', C_3' > 0$ such that

$$G \in \mathcal{N}_{C_1' L_0(\tau_n^{-1}), C_2' s_0(\tau_n^{-1}), C_3' c_0(\tau_n^{-1})}.$$

Clearly, this realization $R(\Phi)$ can be achieved with

$$L \leq \max_{\nu : \tilde{D}_\nu \neq \emptyset} (C_1^\nu) \lceil \log \epsilon^{-1} \rceil + 1 = \max_{\nu : \tilde{D}_\nu \neq \emptyset} (C_1^\nu + 1) \lceil \log \tau_n \rceil =: C_1' L_0(\tau_n^{-1})$$

layers, sparsity

$$s \leq 2u(C_2^\nu \epsilon^{-\rho} \lceil \log \epsilon^{-1} \rceil + Ld) \leq 2r(C_2^\nu \epsilon^{-\rho} \lceil \log \epsilon^{-1} \rceil + Ld) = C_2' s_0(\tau_n^{-1})$$

and weights in $\mathcal{W}_{C_3' c_0(\tau_n)}$ with

$$C_3' c_0(\tau_n^{-1}) \geq C_3^\nu + C_4^\nu \lceil \log \epsilon^{-1} \rceil.$$

with suitably chosen $C_1', C_2' > 0$, $C_3' \in \mathbb{N}$. Note that the constants do not depend on u.

Next, we show that the set $G := R(\Phi)^{-1}(1)$ satisfies the desired approximation property $d_{f_\mathbb{Q}}(G, G_\mathbb{Q}^*) \leq \tau_n^{-\kappa}$. First, for $\nu = 1, \ldots, u$ define E_ν as follows. Let

$$E_\nu := \bigcup_{j=1}^{d} \left(\prod_{i=1}^{j-1} [0,1] \right) \times \left(\left[a_j^\nu, \tilde{a_j}^\nu \right] \cup \left[\tilde{b_j}^\nu, b_j^\nu \right] \right) \times \left(\prod_{i=j+1}^{d} [0,1] \right).$$

It is easy to see that $\tilde{D}_\nu = \emptyset$ implies $D_\nu \subseteq E_\nu$. Set $E := \bigcup_{\nu=1}^u E_\nu \cup \tilde{D}_\nu$. Figure 7.2 shows E in the example considered in figures 3.1 and 7.2. Clearly $G_\mathbb{Q}^*, G \subseteq E$. Thus

$$\begin{aligned} d_{f_\mathbb{Q}}(G_\mathbb{Q}, G) &= \int_{G_\mathbb{Q}^* \Delta G} |2 f_\mathbb{Q}(x) - 1| \mathbb{Q}_X(\mathrm{d}x) \\ &\leq M \left(\sum_{\nu=1}^u \int_{E_\nu} 1 \mathrm{d}x + \sum_{\nu=1}^u \int_{(G_\mathbb{Q}^* \Delta G) \cap \tilde{D}_\nu} |2 f_\mathbb{Q}(x) - 1| \mathrm{d}x \right) \\ &=: M\big((I) + (II)\big). \end{aligned}$$

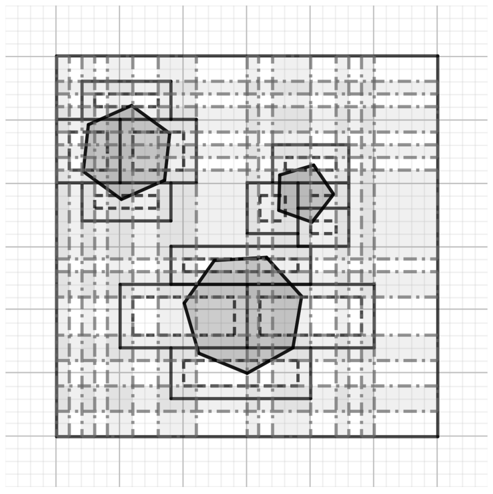


Figure 7.2: The set E when considering the example from figures 3.1 and 7.1. The light grey set represents E. Note that E covers a majority of the space, since $\epsilon = \tau_n^{-1}$ is quite large in this example. Observe that $G_{\mathbb{Q}}^*, G \subseteq E$.

We need to bound both terms. For (I) we observe that by construction $h_{\epsilon^\kappa} \leq 2\tau_n^\kappa$ we have

$$(I) \leq \sum_{\nu=1}^{u}\sum_{j=1}^{d} h_{\epsilon^\kappa} \leq 2rd\tau_n^\kappa.$$

The calculations for the second term are a bit more involved. First, observe that by construction of G, for all $\nu = 1, \dots, u$ we have

$$\begin{aligned}&\int_{(G_{\mathbb{Q}}^* \Delta G)\cap \tilde{D}_\nu} |2f_{\mathbb{Q}}(x) - 1|\mathrm{d}x \\ &= \int_{\prod_{i\neq j_\nu}[\tilde{a}_i^\nu, \tilde{b}_i^\nu]} \int_{a_\nu(x_{-j_\nu})}^{b_\nu(x_{-j_\nu})} |2f_{\mathbb{Q}}(x) - 1|\mathrm{d}x_{j_\nu}\mathrm{d}x_{-j_\nu}.\end{aligned}$$

where

$$b_\nu(x_{-j_\nu}) := \begin{cases} \tilde{a}_{j_\nu}^\nu, & \text{if } \widehat{\gamma}_\nu(x_{-j_\nu}), \gamma_\nu(x_{-j_\nu}) < \tilde{a}_{j_\lambda}^\nu, \\ \tilde{b}_{j_\nu}^\nu, & \text{if } \tilde{b}_{j_\nu}^\nu < \widehat{\gamma}_\nu(x_{-j_\nu}), \gamma_\nu(x_{-j_\nu}), \\ \max\{\widehat{\gamma}_\nu(x_{-j_\nu}), \gamma_\nu(x_{-j_\nu})\}, & \text{otherwise,} \end{cases}$$

$$a_\nu(x_{-j_\nu}) := \begin{cases} \tilde{a}_{j_\nu}^\nu, & \text{if } \widehat{\gamma}_\nu(x_{-j_\nu}), \gamma_\nu(x_{-j_\nu}) < \tilde{a}_{j_\lambda}^\nu, \\ \tilde{b}_{j_\nu}^\nu, & \text{if } \tilde{b}_{j_\nu}^\nu < \widehat{\gamma}_\nu(x_{-j_\nu}), \gamma_\nu(x_{-j_\nu}), \\ \min\{\widehat{\gamma}_\nu(x_{-j_\nu}), \gamma_\nu(x_{-j_\nu})\}, & \text{otherwise.} \end{cases}$$

Let $x_{-j_\nu} \in \prod_{i\neq j_\nu}[\tilde{a}_i^\nu, \tilde{b}_i^\nu]$ be fixed. We have the following cases.

- Assume $\gamma_\nu(x_{-j_\nu}) \geq \tilde{b}^\nu_{j_\nu}$. Then by construction we have

$$\widehat{\gamma}_\nu(x_{-j_\nu}) \geq \gamma_\nu(x_{-j_\nu}) \geq \tilde{b}^\nu_{j_\nu}$$

and thus

$$\int_{a_\nu(x_{-j_\nu})}^{b_\nu(x_{-j_\nu})} |2f_{\mathbb{Q}}(x) - 1| \mathrm{d}x_{j_\nu} = \int_{\tilde{b}^\nu_{j_\nu}}^{\tilde{b}^\nu_{j_\nu}} |2f_{\mathbb{Q}}(x) - 1| \mathrm{d}x_{j_\nu} = 0.$$

- Assume $\gamma_\nu(x_{-j_\nu}) \leq \tilde{a}^\nu_{-j_\nu}$. Then by construction we have

$$\widehat{\gamma}_\nu(x_{-j_\nu}) \geq \gamma_\nu(x_{-j_\nu}) \geq \tilde{a}^\nu_{j_\nu}$$

and thus

$$\int_{a_\nu(x_{-j_\nu})}^{b_\nu(x_{-j_\nu})} |2f_{\mathbb{Q}}(x) - 1| \mathrm{d}x_{j_\nu} = \int_{\tilde{a}^\nu_{j_\nu}}^{\tilde{a}^\nu_{j_\nu}} |2f_{\mathbb{Q}}(x) - 1| \mathrm{d}x_{j_\nu} = 0.$$

- Assume $\gamma_\nu(x_{-j_\nu}) \in (\tilde{a}^\nu_{j_\nu}, \tilde{b}^\nu_{j_\nu})$, by construction

$$x^*_: = (x_1, \ldots, x_{j_\nu - 1}, \gamma_\nu(x_{-j_\nu}), x_{j_\nu + 1}, \ldots, x_d) \in \partial G^*_{\mathbb{Q}}.$$

Consider $\gamma_\nu(x_{-j_\nu}) \leq x_{j_\nu} \leq b_\nu(x_{-j_\nu})$. Let

$$x = (x_1, \ldots, x_{j_\nu - 1}, x_{j_\nu}, x_{j_\nu + 1}, \ldots, x_d).$$

Now, for $\beta = 0$ we have

$$\begin{aligned}\int_{a_\nu(x_{-j_\nu})}^{b_\nu(x_{-j_\nu})} |2f_{\mathbb{Q}}(x) - 1| \mathrm{d}x_{j_\nu} &\leq \int_{\gamma_\nu(x_{-j_\nu})}^{\widehat{\gamma}_\nu(x_{-j_\nu})} 1 \mathrm{d}x_{j_\nu} \\ &= \widehat{\gamma}_\nu(x_{-j_\nu}) - \gamma_\nu(x_{-j_\nu}) \\ &\leq \tau_n^{-\kappa}.\end{aligned}$$

For $\beta > 0$, by definition of $\mathcal{K}^{\mathcal{F}}_{\mathbb{Q},\epsilon_1,\epsilon_2,,r,d}$ we have

$$|2f_{\mathbb{Q}}(x) - 1| \leq \left| g_{\nu,x^*}\left(x_{j_\nu} - \gamma_\nu(x_{-j_\nu})\right) \right|$$

Let $m := \max\{k \in \mathbb{N} \mid k < \beta\}$ and $\omega := \beta - m$. Using a Taylor expansion, there exists

$y_{j_\nu} \in \big(0, x_{j_\nu} - \gamma_\nu(x_{-j_\nu})\big)$ such that

$$\begin{aligned} g_{\nu,x^*}\big(x_{j_\nu} - \gamma_\nu(x_{-j_\nu})\big) &= g_{\nu,x^*}\big(x_{j_\nu} - \gamma_\nu(x_{-j_\nu})\big) - g_{\nu,x^*}(0) \\ &= \sum_{i=1}^{m-1} \frac{1}{i!}\partial^i_{j_\nu} g_{\nu,x^*}(0)\big(x_{j_\nu} - \gamma_\nu(x_{-j_\nu})\big)^i \\ &\qquad + \frac{1}{m!}\partial^m_{j_\nu} g_{\nu,x^*}(y_{j_\nu})\big(x_{j_\nu} - \gamma_\nu(x_{-j_\nu})\big)^m \\ &= \frac{1}{m!}\partial^m_{j_\nu} g_{\nu,x^*}(y_{j_\nu})\big(x_{j_\nu} - \gamma_\nu(x_{-j_\nu})\big)^m. \end{aligned}$$

Note that we used the definition of $\mathcal{H}_{\beta,B}$ in the last equality for all $i \le m < \beta$. Thus

$$\begin{aligned} &|2f_{\mathbb{Q}}(x) - 1| \\ &\le \big|g_{\nu,x^*}\big(x_{j_\nu} - \gamma_\nu(x_{-j_\nu})\big)\big| \\ &\le \frac{1}{m!}\frac{|\partial^m_{j_\nu} g_{\nu,x^*}(y_{j_\nu}) - \partial^m_{j_\nu} g_{\nu,x^*}(0)|}{(y_{j_\nu} - 0)^\omega}\big(x_{j_\nu} - \gamma_\nu(x_{-j_\nu})\big)^\beta \\ &\le \frac{B}{m!}\big(x_{j_\nu} - \gamma_\lambda(x_{-j_\nu})\big)^\beta. \end{aligned}$$

Similarly, for $a_\nu(x_{-j\nu}) \le x'_{j_\nu} \le \gamma_\nu(x_{-j_\nu})$ we obtain

$$|2f_{\mathbb{Q}}(x) - 1| \le \frac{B}{m!}\big(x_{j_\nu} - \gamma_\lambda(x_{-j_\nu})\big)^\beta.$$

This implies

$$\begin{aligned} &\int_{a_\nu(x_{-j_\nu})}^{b_\nu(x_{-j_\nu})} |2f_{\mathbb{Q}}(x) - 1| \mathrm{d}x_{j_\nu}\mathrm{d}x_{-j_\nu} \\ &\le \int_{\gamma_\nu(x_{-j_\nu})}^{\widehat{\gamma}_\nu(x_{-j_\nu})} \frac{B}{m!}\big(x_{j_\nu} - \gamma_\nu(x_{-j_\nu})\big)^\beta \mathrm{d}x_{j_\nu}\mathrm{d}x_{-j_\nu} \\ &= \frac{B}{m!(\beta+1)}\big(\widehat{\gamma}_\nu(x_{-j_\nu}) - \gamma_\nu(x_{-j_\nu})\big)^{\beta+1} \\ &\le \frac{B}{m!(\beta+1)}\tau_n^{-\kappa}. \end{aligned}$$

Therefore, we have

$$\int_{a_\nu(x_{-j_\nu})}^{b_\nu(x_{-j_\nu})} |2f_{\mathbb{Q}}(x) - 1| \mathrm{d}x_{j_\nu}\mathrm{d}x_{-j_\nu} \le \max\left\{\frac{B}{m!(\beta+1)}, 1\right\}\tau_n^{-\kappa}.$$

for all $\nu = 1, \dots, u$ and $x_{-j_\nu} \in \prod_{i \ne j_\nu}[\tilde{a}^\nu_i, \tilde{b}^\nu_i]$ which yields

$$(II) \le \max\left\{\frac{B}{m!(\beta+1)}, 1\right\}\tau_n^{-\kappa}.$$

Thus

$$d_f(G, G^*_{\mathbb{Q}}) \leq \left(2rd + \max\left\{\frac{B}{m!(\beta+1)}, 1\right\}\right) \tau_n^{-\kappa}$$

which concludes the proof. □

The remainder of the proof of our main result is now simple.

Proof of Theorem 3.3.4. We check the requirements in Proposition 3.2.1.

Conditions (i) and (ii) are clear.

Condition (iii) follows from Theorem 7.2.3.

Lastly, we need to prove (iv). Let $n \geq N_0$ where N_0 is defined in Theorem 7.2.3. By Lemma 3.3.2 we have

$$\begin{aligned}|\mathcal{N}_n| &= \left|\mathcal{N}_{C_1' L_0(\tau_n), C_2' s_0(\tau_n), C_3' c_0(\tau_n^{-1})}\right| \\ &\leq \Big((dC_2' s_n(\tau_n) \\ &\qquad + \min\{C_1' L_n(\tau_n), C_2' s_0(\tau_n^{-1})\}(C_2' s_0(\tau_n) + 1)^2)2^{C_3' c_0(\tau_n^{-1})+2}\Big)^{C_2' s_0(\tau_n)}.\end{aligned}$$

Inserting all variables yields

$$|\mathcal{N}_n| \leq \left(k_1 \tau_n^{k_2} \log^2(\tau_n)\right)^{k_3 \tau_n^{\rho} \log(\tau_n)}$$

for some constants $k_1, k_2, k_3 > 0$. Thus

$$\log\left(|\mathcal{N}_n|\right) \leq k_4 \tau_n^{\rho} \log^2(\tau_n)$$

for some constant $k_4 > 0$. By setting

$$\tau_n := \frac{n^{\frac{1}{2\kappa+\rho-1}}}{\log^{\frac{2}{\rho}}(n)}$$

we obtain assumption (iv)

$$\log|\mathcal{N}_n| \leq c_3 n^{\frac{\rho}{2\kappa+\rho-1}}.$$

□

Proofs for Regular Boundaries

Next, we prove Lemma 3.3.5. We first provide the corresponding statement from [109]. Lemma 3.3.5 is a reformulated version.

Theorem 7.2.4. *(Theorem 5 in [109])*
For any function $f \in \mathcal{F}_{\beta,B,d}$ and any integers $m \geq 1$ and $N \geq \max\left\{(\beta+1)^d, B+1\right\}$ there exists

a neural network Φ *with*

$$L = 8 + (m+5)(1 + \lceil d \log_2 d \rceil)$$

layers, sparcity

$$s \leq 94 d^2 (\beta+1)^{2d} N (m+6)(1 + \lceil \log_2 d \rceil)$$

and weights $|w| \leq 1$ *such that*

$$\|R(\Phi) - f\|_\infty \leq (2B+1) 3^{d+1} N 2^{-m} + B 2^\beta N^{-\frac{\beta}{d}}.$$

Proof. Theorem 5 in [109]. □

Proof of Lemma 3.3.5. Let N be the smallest integer satisfying

$$N \geq k_1 \epsilon^{-\frac{d}{\beta}} \geq \max\left\{ \left(B 2^{\beta+2} \epsilon^{-1}\right)^{\frac{d}{\beta}}, (\beta+1)^d, B+1 \right\},$$

where $k_1 := \max\left\{ \left(B 2^{\beta+2}\right)^{\frac{d}{\beta}}, (\beta+1)^d, B+1 \right\}$. Let k_2 be the smallest integer satisfying

$$k_2 \geq \log(k_1+1) + \left(\frac{d}{\beta} + 1\right) + \log((2B+1)3^{d+1}) + 1$$

and define $m := k_2 \lceil \log(\epsilon^{-1}\rceil \in \mathbb{N}$. Since $\epsilon < 3^{-1}$ and thus $\log(\epsilon^{-1}) \geq 1$ we have

$$\begin{aligned} m &\geq \left(\log(k_1+1) + \left(\frac{d}{\beta}+2\right) + \log((2B+1)3^{d+1}) + 1 \right) \log(\epsilon^{-1}) \\ &\geq \log(k_1+1) + \left(\frac{d}{\beta}+1\right) \log(\epsilon^{-1}) + \log((2B+1)3^{d+1}) + 2 \\ &\geq N + \log((2B+1)3^{d+1}) + \log(\epsilon^{-1}) + 2. \end{aligned}$$

Theorem 7.2.4 implies the following. For any $f \in \mathcal{F}_{\beta,B,d}$ there exists a network $\tilde{\Phi}$ with

$$L = 8 + (k_2 \lceil \log \epsilon^{-1} \rceil + 5)(1 + \lceil \log_2 d \rceil)$$

layers, sparsity

$$s \leq 94 d^2 (\beta+1)^{2d} k_1 \epsilon^{-\frac{d}{\beta}} (k_2 \lceil \log \epsilon^{-1} \rceil + 6)(1 + \lceil \log_2 d \rceil)$$

and weights $|w_i| \leq 1$ such that

$$\|R(\tilde{\Phi}) - f\|_\infty \leq \frac{\epsilon}{2}.$$

Note that

$$\begin{aligned} L &\leq (8 + (2k_2+5)(1 + \lceil \log_2 d \rceil) \log \epsilon^{-1} =: c_1 \log \epsilon^{-1}, \\ s &\leq \left(94 d^2 (\beta+1)^{2d} k_1 (2k_2+6)(1 + \lceil \log_2 d \rceil\right) \epsilon^{-\frac{d}{\beta}} \log \epsilon^{-1} =: c_2 \epsilon^{-\frac{d}{\beta}} \log \epsilon^{-1}. \end{aligned}$$

Let V be defined as in the proof of Lemma 3.3.2. Following the proof of Lemma 12 of [109], we see that for any $g \leq \frac{\epsilon}{4(L+1)V}$ there is a neural network Φ with L layers and sparsity s such that

$$\|R(\Phi) - R(\tilde{\Phi})\|_\infty \leq \frac{\epsilon}{2}.$$

where the nonzero weights of Φ are discretized with grid size g. Now, define

$$\begin{aligned}
&\frac{\epsilon}{4(L+1)V} \\
&= \frac{\epsilon}{4(L+1)\big(ds + L(s+1)^2\big)} \\
&\geq \frac{\epsilon}{4(c_1\lceil\log\epsilon^{-1}\rceil + 1)\big(dc_2\epsilon^{-\frac{d}{\beta}}\lceil\log\epsilon^{-1}\rceil + c_1\lceil\log\epsilon^{-1}\rceil(c_2\epsilon^{-\frac{d}{\beta}}\lceil\log\epsilon^{-1}\rceil + 1)^2\big)} \\
&\geq \frac{1}{4(c_1+1)(dc_2 + c_1(c_2+1)^2}\epsilon^{2+2\frac{d}{\beta}} \\
&\geq 2^{-\left(c_3 + c_4\lceil\log(\epsilon^{-1})\rceil\right)} =: g.
\end{aligned}$$

with

$$\begin{aligned}
c_3 &:= \lceil\log\big(4(c_1+1)(dc_2 + c_1(c_2+1)^2\big)\rceil, \\
c_4 &:= \left\lceil 2 + 2\frac{d}{\beta}\right\rceil.
\end{aligned}$$

Therefore, all weights are elements of $\mathcal{W}_c$ with $c = c_3 + c_4\lceil\log\epsilon^{-1}\rceil$ and

$$\|R(\Phi) - f\|_\infty \leq \|R(\Phi) - R(\tilde{\Phi})\|_\infty + \|R(\tilde{\Phi}) - f\|_\infty \leq \frac{\epsilon}{2} + \frac{\epsilon}{2} = \epsilon.$$

□

Lastly, we prove Lemma 3.3.8. The extension to this case is similar to the extension in [109].

Proof of Lemma 3.3.8. Let

$$\gamma = \gamma_r \circ \cdots \circ \gamma_1 \in \mathcal{G}_{r,t,\beta,B,d}$$

with $\gamma_i = (\gamma_{ij} \circ \iota_{ij})_{j=1}^{d_i+1}$. We first construct a candidate network Φ for γ. Then, we show that it approximates gamma well and satisfies the required properties.

In order to construct a network that approximates γ well, we first approximate γ_{ij} and τ_{ij} using neural networks. The final network is constructed using concatenation and parallelization.
Let $i = 1, \dots, r$, $j = 1, \dots, d_i + 1$ and $\epsilon_i > 0$. Using Lemma 3.3.5 there exist constants $\epsilon_0^i, c_1^i, c_2^i > 0, c_3^i, c_4^i \in \mathbb{N}$ such that there exists a neural network Φ_{ij} with $L^{ij} \leq c_1^i\lceil\log(\epsilon_i^{-1})\rceil$ layers, sparsity $s^{ij} \leq c_2^i\epsilon_i^{-\frac{t_i}{\beta_i}}\log(\epsilon_i^{-1})$ and weights in $\mathcal{W}_{c^i}$ with $c^i := c_3^i + c_4^i\lceil\log(\epsilon_i^{-1})\rceil$ such that

$$\|R(\Phi_{ij})(x) - \gamma_{ij}\|_\infty \leq \epsilon_i$$

if $\epsilon_i < \epsilon_0^i$. Let $\widehat{\gamma}_{ij} := R(\Phi_{ij})(x)$. Additionally, the function ι_{ij} is the realization of a network

with 0 Layers and sparsity t_i.
Since concatenating and parallelizing networks using Lemma 7.2.1 leads to linear transformations on the upper bounds on the Layers, sparsity and the constant c', there exist constants $c_1', c_2' > 0$, $c_3', c_4' \in \mathbb{N}$ such that the function

$$\widehat{\gamma} = \widehat{\gamma}_r \circ \cdots \circ \widehat{\gamma}_1$$

with $\widehat{\gamma}_i = (\widehat{\gamma}_{ij} \circ \iota_{ij})_{j=1}^{d-1}$ is the realization of a network with

$$\begin{aligned} L &\leq c_1' \max_{i=1,\dots,r} \lceil \log(\epsilon_i^{-1}) \rceil \text{ layers,} \\ s &\leq c_2' \max_{i=1,\dots,r} \epsilon_i^{-\frac{t_i}{\beta_i}} \log(\epsilon_i^{-1}) \text{ sparsity,} \\ c' &:= c_3' + c_4' \max_{i=1,\dots,r} \lceil \log(\epsilon_i^{-1}) \rceil \end{aligned}$$

and weights in $\mathcal{W}_{c'}$.

Now, let $\epsilon > 0$ be small enough. We show that $\widehat{\gamma}$ approximates γ well for suitabily chosen ϵ_i. Following Lemma 9 in [109] we have

$$\begin{aligned} \|\gamma - \widehat{\gamma}\|_\infty &\leq C \sum_{i=1}^{r} \| \max_{j=1,\dots,d_{i+1}} |\gamma_{ij} - \widehat{\gamma}_{ij}| \|_\infty^{\prod_{k=i+1}^{r} \min\{\beta_k, 1\}} \\ &\leq C \sum_{i=1}^{r} \epsilon_i^{\prod_{k=i+1}^{r} \min\{\beta_k, 1\}} \\ &\leq Cr \max_{i=1,\dots,r} \epsilon_i^{\prod_{k=1}^{i+1} \min\{\beta_k, 1\}} \end{aligned}$$

for some constant $C > 0$. Set

$$\epsilon_i := \left(\frac{\epsilon}{Cr}\right)^{\frac{1}{\prod_{k=1}^{i+1} \min\{\beta_k, 1\}}}.$$

First, this implies

$$\|\gamma - \widehat{\gamma}\|_\infty \leq \epsilon.$$

Additionally, the network Φ has

$$\begin{aligned} L &\leq c_1' \max_{i=1,\dots,r} \lceil \log(\epsilon_i^{-1}) \rceil \leq c_1 \lceil \log(\epsilon^{-1}) \rceil \text{ layers,} \\ s &\leq c_2' \max_{i=1,\dots,r} \epsilon_i^{-\frac{t_i}{\beta_i}} \log(\epsilon_i^{-1}) = c_2 \max_{i=1,\dots,r} \epsilon^{-\frac{t_i}{\beta_i \prod_{k=1}^{i+1} \min\{\beta_k, 1\}}} \log(\epsilon^{-1}) \\ &= c_2 \epsilon^{-\rho} \log(\epsilon^{-1}) \text{ sparsity,} \\ c' &= c_3' + c_4' \max_{i=1,\dots,r} \lceil \log(\epsilon_i^{-1}) \rceil \leq c_3 + c_4 \lceil \log(\epsilon^{-1}) \rceil := c \end{aligned}$$

and weights in $\mathcal{W}_c$ for some constants $c_1, c_2 > 0$, $c_3, c_4 \in \mathbb{N}$. □

7.2.3 Lower Bound

We first prove Theorem 3.4.1. The outline of the proof is similar to the proof of Theorem 3 in [91]. However, the setting of Theorem 3.3.4 differs substantially from theirs. This leads to a new situation and new technical challenges to overcome in the proof of Theorem 3.4.1 .

Proof of Theorem 3.4.1. By Hoelders inequality and condition (c) it is enough to consider the case $p = 1$ and the first inequality. Let $\mathfrak{Q}_1 \subseteq \mathfrak{Q}$ be a finite set of potential probability measures of $(X_1, Y_1), \dots, (X_n, Y_n)$. Then

$$\sup_{\mathbb{Q}\in\mathfrak{Q}} \mathbb{E}[d_\Delta(G_n, G^*_{\mathbb{Q}})] \geq \frac{1}{\#\mathfrak{Q}_1} \sum_{\mathbb{Q}\in\mathfrak{Q}_1} \mathbb{E}[d_\Delta(G_n, G^*_{\mathbb{Q}})]$$

Hence, it suffices to show that for any estimator G_n we have

$$\frac{1}{\#\mathfrak{Q}_1} \sum_{\mathbb{Q}\in\mathfrak{Q}_1} \mathbb{E}[d_\Delta(G_n, G^*_{\mathbb{Q}})] \geq cn^{-\frac{1}{2\kappa-1+\rho}} \quad \text{a.s.,} \tag{7.2.3}$$

for some constant $c > 0$. We now define the set $\mathfrak{Q}_1$. Then, we prove $\mathfrak{Q}_1 \subseteq \mathfrak{Q}$. Lastly, we show that $\mathfrak{Q}_1$ satisfies (7.2.3).

Let $\phi : \mathbb{R} \to [0, 1]$ be an infinitely many times differentiable function with the following properties:

- $\phi(t) = 0$ for $|t| \geq 1$,
- $\phi(0) = 1$.

Let $K \geq 2$ be an integer. For $i \in \{1, \dots, K\}^{d-1}$ define

$$\phi_i : [0,1]^{d-1} \to [0,1],\ \phi_i(y) = k_1 K^{-\beta_2} \prod_{j=1}^{d-1} \phi\left(K\left(y_j - \frac{2i_j - 1}{K}\right)\right)$$

for some $0 < k_1$ small enough. Define

$$W := \prod_{i\in\{1,\dots,K\}^{d-1}} \{0,1\}.$$

For $w \in W$ let

$$\gamma_w : [0,1]^{d-1} \to [0,1],\ \gamma_w(y) = \sum_{i\in\{1,\dots,K\}^{d-1}} w_i\phi_i(y).$$

Now, for $w \in W$ define $\mathbb{Q}_w$ as follows. The marginal distribution $\mathbb{Q}_{w,X}$ is the uniform distribution on $[0,1]^d$ and

$$\begin{aligned} f_{\mathbb{Q}_w}(x) := & \frac{1}{2}\left(1 + k_2\left(\gamma_w(x_{-1}) - x_1\right)^{\beta_1}\right)\mathbb{1}\left(x_1 \le \gamma_w(x_{-1})\right) \\ & + \frac{1}{2}\left(1 - k_2 x_1^{\beta_1}\right)\mathbb{1}\left(0 < x_1 \le \gamma_{1-w}(x_{-1})\right) \\ & + \frac{1}{2}\left(1 - k_3\left(x_1 - \gamma_1(x_{-1})\right)^{\beta_1}\right)\mathbb{1}(\gamma_1(x_{-1}) < x_1) \end{aligned}$$

for some $k_2, k_3 > 0$. Finally, let

$$\mathfrak{Q}_1 := \left\{\mathbb{Q}_w \mid w \in W\right\}.$$

We now show that $\mathfrak{Q}_1 \subseteq \mathfrak{Q}$ by properly selecting the constants c_1, k_1, k_2, k_3 such that $f_{\mathbb{Q}_w}$ is well defined for all $w \in W$ and showing that $\mathfrak{Q}_1$ satisfies the conditions (a),(b),(c).

First of all, we choose k_1, k_3 small enough and (given $k_2 > 0$) K_0 large enough such that for all $K \ge K_0$ we have

$$\frac{1}{4} \le f_{\mathbb{Q}_w}(x) \le 1$$

for all $x \in [0,1]^d$ and $w \in W$.

(a) Clearly, for all $w \in W$ the marginal distribution of $\mathbb{Q}_w$ with respect to X has a Lebesgue density which bounded by $1 \le M$.

(b) We need to show

$$G^*_{\mathbb{Q}_w} = \left\{x \in [0,1]^d \middle| \, f_{\mathbb{Q}_w}(x) \ge \frac{1}{2}\right\} \in \mathcal{K}^{\mathcal{F}_{\beta_2,B_2,d-1}}_{\mathbb{Q},\beta,B,\epsilon_1,\epsilon_2,r,d}$$

for all $w \in W$.

1. Clearly, by selecting $\nu = u = 1$, $j = j_\nu = 1$, $\iota_2 = 1$, $D_\nu = [0,1]^d$ and $\gamma = \gamma_w$ we have

$$G^*_{\mathbb{Q}_w} = H_1 = D_\nu \cap \left\{x \in [0,1]^d \mid \iota_2 x_1 \le \gamma(x_{-1})\right\}.$$

For k_1 small enough we also have $\gamma \in \mathcal{F}_{\beta_2,B_2,d-1}$ for all $w \in W$.

2. clear.
3. If $\beta_1 > 0$, for $w \in W$ and $x \in \partial G^*_{\mathbb{Q}_w}$ we have $x_1 = \gamma_w(x_{-1})$. Let

$$g_{\nu,x} : [0,1] \to \mathbb{R}, \; g_{\nu,x}(y) = \max\{k_2, k_3\} y^{\beta_1}.$$

Note that for k_2, k_3 small enough we have $g_{\nu,x} \in \mathcal{H}_{\beta_1,B_1}$. Additionally, we have

$$\begin{aligned} |2f_{\mathbb{Q}_w}(y) - 1| &\le g_{\nu,x}(y - x_1), \text{ for } y \ge x_1, \\ |2f_{\mathbb{Q}_w}(y) - 1| &\le g_{\nu,x}(x_1 - y), \text{ for } y \le x_1. \end{aligned}$$

4. clear.

This implies the assertion.

(c) Let $w \in W$. For $\beta_1 = 0$ we have

$$d_\Delta^\kappa(G, G^*_{\mathbb{Q}_w}) = d_\Delta(G, G^*_{\mathbb{Q}_w}) = \frac{1}{\min\{k_2, k_3\}} d_{f_{\mathbb{Q}_w}}(G, G^*_{\mathbb{Q}_w})$$

For $\beta_1 > 0$, there is an $\eta_0 > 0$ such that for all $0 < \eta \leq \eta_0$ we have

$$\begin{aligned}
&\lambda\Big(\{x \in [0,1]^d \mid |2f_{\mathbb{Q}_w}(x) - 1| \leq \eta\}\Big) \\
&\leq \lambda\Bigg(\Big\{x \in [0,1]^d \;\Big|\; x_1 \leq \gamma_w(x_{-1}),\ k_2(\gamma_w(x_{-1}) - x_1)^{\beta_1} \leq \eta\Big\} \\
&\quad \cup \Big\{x \in [0,1]^d \;\Big|\; x_1 \leq \gamma_{1-w}(x_{-1}),\ k_2 x_1^{\beta_1} \leq \eta\Big\} \\
&\quad \cup \Big\{x \in [0,1]^d \;\Big|\; \gamma_1(x_{-1}) \leq x_1,\ k_3\Big(x_1 - \gamma_1(x_{-1})\Big)^{\beta_1} \leq \eta\Big\}\Bigg) \\
&\leq \lambda\Bigg(\left\{x \in [0,1]^d \;\middle|\; \gamma_2(x_{-1}) - \frac{1}{k_2^{\frac{1}{\beta_1}}}\eta^{\frac{1}{\beta_1}} \leq x_1 \leq \gamma_w(x_{-1})\right\} \\
&\quad \cup \left\{x \in [0,1]^d \;\middle|\; x_1 \leq \frac{1}{k_2^{\frac{1}{\beta_1}}}\eta^{\frac{1}{\beta_1}}\right\} \\
&\quad \cup \left\{x \in [0,1]^d \;\middle|\; \gamma_1(x_{-1}) \leq x_1,\ \gamma_1(x_{-1}) + \frac{1}{k_3^{\frac{1}{\beta_1}}}\eta^{\frac{1}{\beta_1}}\right\}\Bigg) \\
&\leq \left(\frac{2}{k_2^{\frac{1}{\beta_1}}} + \frac{1}{k_3^{\frac{1}{\beta_1}}}\right)\eta^{\frac{1}{\beta_1}}.
\end{aligned}$$

Following Proposition 1 in of [126] there exists $\tilde{c}_1, \tilde{\eta}_0 > 0$ such that

$$d_\Delta^\kappa(G, G^*_{\mathbb{Q}_w}) \leq \tilde{c}_1 d_{f_{\mathbb{Q}_w}}(G, G^*_{\mathbb{Q}_w})$$

for all G such that $d_\Delta(G, G^*_{\mathbb{Q}_w}) \leq \tilde{\eta}_0$. If $\tilde{\eta}_0 \geq 1$ this implies the assertion with $c_1 := \tilde{c}_1$. If not, the assertion is implied by setting $c_1 := \frac{\tilde{c}_1}{\tilde{\eta}_0^\kappa}$.

Next we prove Inequality (7.2.3). For $w \in W$ write $\mathbb{Q}_w^n$ for the probability measure of the distribution of $(X_1, Y_1), \ldots, (X_n, Y_n)$ when the underlying distribution is $\mathbb{Q}_w$. Define the product measure $\psi = \zeta \times \lambda$, where ζ is the counting measure on $\{0,1\}$. Note that $\mathbb{Q}_w$ has a density with respect to ψ which is given by

$$\mathrm{d}\mathbb{Q}_w = \frac{\mathrm{d}\mathbb{Q}_w}{\mathrm{d}\psi}(y, x) := \mathbb{1}(y = 1) f_w(x) + \mathbb{1}(y = 0) \cdot (1 - f_w(x)).$$

Assume $w_1, w_2 \in W$ differ by only 1 entry. We obtain

$$\int \min\{\mathrm{d}\mathbb{Q}^n_{w_1}, \mathrm{d}\mathbb{Q}^n_{w_2}\}\mathrm{d}\psi = \int \min\{\mathrm{d}\mathbb{Q}^n_0, \mathrm{d}\mathbb{Q}^n_1\}\mathrm{d}\psi,$$

where for $s = 0, 1$ we write $\mathbb{Q}_s^n = \mathbb{Q}_{w^s}^n$ with

$$w_i^s := \begin{cases} s, & \text{for } i_1 = \cdots = i_{d-1} = 1, \\ 0, & \text{otherwise} \end{cases}$$

for $i \in \{1, \ldots, K\}^{d-1}$. Then, using Assouad's Lemma we get

$$\begin{aligned} &\frac{1}{\#\mathfrak{Q}_1} \sum_{\mathbb{Q} \in \mathfrak{Q}_1} \mathbb{E}[d_\Delta(G_n, G_{\mathbb{Q}}^*)] \\ &\geq \frac{1}{2} K^{d-1} \lambda\left(\left\{ x \in [0,1]^d \;\middle|\; x_1 \leq \phi_1(x_{-1}) \right\} \right) \int \min\{\mathrm{d}\mathbb{Q}_0^n, \mathrm{d}\mathbb{Q}_1^n\} \mathrm{d}\psi \\ &= \frac{1}{2} k_1 K^{d-1-\beta_2} \int_{\mathbb{R}^{d-1}} \prod_{j=1}^{d-1} \phi(K x_{j+1}) \mathrm{d}x_{-1} \int \min\{\mathrm{d}\mathbb{Q}_0^n, \mathrm{d}\mathbb{Q}_1^n\} \mathrm{d}\psi. \end{aligned}$$

We first bound the term $\int \min\{\mathrm{d}\mathbb{Q}_0^n, \mathrm{d}\mathbb{Q}_1^n\} \mathrm{d}\psi$. By using the fact that

$$\int \mathrm{d}\mathbb{Q}_0 \mathrm{d}\psi = 1$$

and Hoelders inequality in the fourth row we calculate

$$\begin{aligned} &\int \min\{\mathrm{d}\mathbb{Q}_0^n, \mathrm{d}\mathbb{Q}_1^n\} \mathrm{d}\psi \\ &= 1 - \frac{1}{2} \int |\mathrm{d}\mathbb{Q}_0^n - \mathrm{d}\mathbb{Q}_1^n| \mathrm{d}\psi \\ &= 1 - \frac{1}{2} \int \left| \sqrt{\mathrm{d}\mathbb{Q}_0^n} - \sqrt{\mathrm{d}\mathbb{Q}_1^n} \right| \cdot \left| \sqrt{\mathrm{d}\mathbb{Q}_0^n} + \sqrt{\mathrm{d}\mathbb{Q}_1^n} \right| \mathrm{d}\psi \\ &\geq 1 - \frac{1}{2} \left(\int \left(\sqrt{\mathrm{d}\mathbb{Q}_0^n} - \sqrt{\mathrm{d}\mathbb{Q}_1^n} \right)^2 \mathrm{d}\psi \right)^{\frac{1}{2}} \left(\int \left(\sqrt{\mathrm{d}\mathbb{Q}_0^n} + \sqrt{\mathrm{d}\mathbb{Q}_1^n} \right)^2 \mathrm{d}\psi \right)^{\frac{1}{2}}. \end{aligned}$$

By repeatedly using the fact that $\int \mathrm{d}\mathbb{Q}_0 \mathrm{d}\psi = 1$, this implies

$$\begin{aligned}
&\int \min\{\mathrm{d}\mathbb{Q}_0^n, \mathrm{d}\mathbb{Q}_1^n\}\mathrm{d}\psi \\
&= 1 - \frac{1}{2}\left(2\left(1 - \int \sqrt{\mathrm{d}\mathbb{Q}_0^n \mathrm{d}\mathbb{Q}_1^n}\mathrm{d}\psi\right)\right)^{\frac{1}{2}}\left(2\left(1 + \int \sqrt{\mathrm{d}\mathbb{Q}_0^n \mathrm{d}\mathbb{Q}_1^n}\mathrm{d}\psi\right)\right)^{\frac{1}{2}} \\
&= 1 - \left(1 - \left(\int \sqrt{\mathrm{d}\mathbb{Q}_0^n \mathrm{d}\mathbb{Q}_1^n}\mathrm{d}\psi\right)^2\right)^{\frac{1}{2}} \\
&\geq 1 - \left(1 - \left(\int \sqrt{\mathrm{d}\mathbb{Q}_0^n \mathrm{d}\mathbb{Q}_1^n}\mathrm{d}\psi\right)^2 + \frac{1}{4}\left(\int \sqrt{\mathrm{d}\mathbb{Q}_0^n \mathrm{d}\mathbb{Q}_1^n}\mathrm{d}\psi\right)^4\right)^{\frac{1}{2}} \\
&= 1 - \left(1 - \frac{1}{2}\left(\int \sqrt{\mathrm{d}\mathbb{Q}_0^n \mathrm{d}\mathbb{Q}_1^n}\mathrm{d}\psi\right)^2\right) \\
&= \frac{1}{2}\left(\int \sqrt{\mathrm{d}\mathbb{Q}_0^n \mathrm{d}\mathbb{Q}_1^n}\mathrm{d}\psi\right)^2.
\end{aligned}$$

By independence we have

$$\int \sqrt{\mathrm{d}\mathbb{Q}_0^n \mathrm{d}\mathbb{Q}_1^n}\mathrm{d}\psi = \left(\int \sqrt{\mathrm{d}\mathbb{Q}_0 \mathrm{d}\mathbb{Q}_1}\mathrm{d}\psi\right)^n.$$

Additionally, observe that

$$\begin{aligned}
&\int \sqrt{\mathrm{d}\mathbb{Q}_0 \mathrm{d}\mathbb{Q}_1}\mathrm{d}\psi \\
&= \frac{1}{2}\int \mathrm{d}\mathbb{Q}_0 \mathrm{d}\psi + \frac{1}{2}\int \mathrm{d}\mathbb{Q}_1 \mathrm{d}\psi - \frac{1}{2}\int \left(\sqrt{\mathrm{d}\mathbb{Q}_0} - \sqrt{\mathrm{d}\mathbb{Q}_1}\right)^2 \mathrm{d}\psi \\
&= 1 - \frac{1}{2}\int \left(\sqrt{\mathrm{d}\mathbb{Q}_0} - \sqrt{\mathrm{d}\mathbb{Q}_1}\right)^2 \mathrm{d}\psi
\end{aligned}$$

and

$$\begin{aligned}
&\int \left(\sqrt{\mathrm{d}\mathbb{Q}_0} - \sqrt{\mathrm{d}\mathbb{Q}_1}\right)^2 \mathrm{d}\psi \\
&\leq \int \left(\sqrt{f_{w^0}(x)} - \sqrt{f_{w^1}(x)}\right)^2 \mathrm{d}x + \int \left(\sqrt{1 - f_{w^0}(x)} - \sqrt{1 - f_{w^1}(x)}\right)^2 \mathrm{d}x \\
&\leq 2\int \left(f_{w^0}(x) - f_{w^1}(x)\right)^2 \mathrm{d}x + 2\int \left(1 - f_{w^0}(x) - \left(1 - f_{w^1}(x)\right)\right)^2 \mathrm{d}x \\
&= 4\int \left(f_{w^0}(x) - f_{w^1}(x)\right)^2 \mathrm{d}x
\end{aligned}$$

where we used that $f_w(x) \geq \frac{1}{4}$ for all $x \in [0,1]^d$ and $w \in W$. Next, we calculate

$$\begin{aligned}
&\int \big(f_{w^0}(x) - f_{w^1}(x)\big)^2 \mathrm{d}x \\
&\geq \frac{1}{4} \int_{[0,1]^{d-1}} \int_0^{\phi_j(x_{-1})} \Big(k_2 \left(\phi_j(x_{-1}) - x_1\right)^{\beta_1} + k_2 x_1^{\beta_1}\Big)^2 \mathrm{d}x_1 \mathrm{d}x_{-1} \\
&\geq \frac{k_2^2}{4} \int_{[0,1]^{d-1}} \int_0^{\phi_j(x_{-1})} \left(\phi_j(x_{-1}) - x_1\right)^{2\beta_1} \mathrm{d}x_1 \mathrm{d}x_{-1} \\
&\quad + \frac{k_2^2}{4} \int_{[0,1]^{d-1}} \int_0^{\phi_j(x_{-1})} x_1^{2\beta_1} \mathrm{d}x_1 \mathrm{d}x_{-1} \\
&= \frac{k_2^2}{4}(I_1 + I_2).
\end{aligned}$$

We need to control the terms I_1 and I_2. For the first we obtain

$$\begin{aligned}
I_1 &:= \int_{[0,1]^{d-1}} \int_0^{\phi_j(x_{-1})} \left(\phi_j(x_{-1}) - x_1\right)^{2\beta_1} \mathrm{d}x_1 \mathrm{d}x_{-1} \\
&= \int_{[0,1]^{d-1}} \int_0^{\phi_j(x_{-1})} x_1^{2\beta_1} \mathrm{d}x_1 \mathrm{d}x_{-1} \\
&= \frac{1}{1+2\beta_1} \int_{[0,1]^{d-1}} \phi_j(x_{-1})^{1+2\beta_1} \mathrm{d}x_{-1} \\
&\leq \frac{k_1^{1+2\beta_1}}{1+2\beta_1} K^{-\beta_2(1+2\beta_1)} \int_{\mathbb{R}^{d-1}} \prod_{j=1}^{d-1} \phi(Kx_{j+1})^{1+2\beta_1} \mathrm{d}x_{-1} \\
&\leq 2 \frac{k_1^{1+2\beta_1}}{1+2\beta_1} K^{-\beta_2(1+2\beta_1)-(d-1)} \\
&= 2 \frac{k_1^{1+2\beta_1}}{1+2\beta_1} K^{-\beta_2(2\kappa-1+\rho)}
\end{aligned}$$

and similarly

$$\begin{aligned}
I_2 &:= \int_{[0,1]^{d-1}} \int_0^{\phi_j(x_{-1})} x_1^{2\beta_1} \mathrm{d}x_1 \mathrm{d}x_{-1} \\
&\leq 2 \frac{k_1^{1+2\beta_1}}{1+2\beta_1} K^{-\beta_2(2\kappa-1+\rho)}.
\end{aligned}$$

This implies

$$\int \min\{\mathrm{d}\mathbb{Q}_0^n, \mathrm{d}\mathbb{Q}_1^n\} \mathrm{d}\psi \geq \frac{1}{2}\left(1 - c^* K^{-\beta_2(2\kappa-1+\rho)}\right)^{2n}$$

for some constant $c^* > 0$. By setting $K := n^{\frac{1}{\beta_2}\frac{1}{2\kappa-1+\rho}}$ we obtain

$$\int \min\{\mathrm{d}\mathbb{Q}_0^n, \mathrm{d}\mathbb{Q}_1^n\} \mathrm{d}\psi \geq \frac{1}{2}\left(1 - c^* \frac{1}{n}\right)^{2n} > c'$$

for some constant $c' > 0$ for n large enough. Thus

$$\begin{aligned}
&\frac{1}{\#\mathfrak{Q}_1}\sum_{\mathbb{Q}\in\mathfrak{Q}_1}\mathbb{E}[d_\Delta(G_n, G^*_{\mathbb{Q}})]\\
&= \frac{1}{2}k_1K^{d-1-\beta_2}\int_{\mathbb{R}^{d-1}}\prod_{j=1}^{d-1}\phi(Kx_{j+1})\mathrm{d}x_{-1}\int\min\{\mathrm{d}\mathbb{Q}_0^n, \mathrm{d}\mathbb{Q}_1^n\}\mathrm{d}\psi\\
&\geq \frac{1}{2}k_1K^{-\beta_2}\int_{\mathbb{R}^{d-1}}\prod_{j=1}^{d-1}\phi(x_{j+1})\mathrm{d}x_{-1}\cdot c'\\
&\geq cn^{-\frac{1}{2\kappa-1+\rho}}
\end{aligned}$$

for some constant $c > 0$. This concludes the proof. $\square$

Lastly, the proof of Theorem 3.4.2 is provided. The ideas used in the proof are very similar to those used in the proof of Theorem 3.4.1 above. We therefore only focus on the differences.

Proof of Theorem 3.4.2. As in the proof of Theorem 3.4.1 the strategy is to show that for any estimator G_n we have

$$\frac{1}{\#\mathfrak{Q}_1}\sum_{\mathbb{Q}\in\mathfrak{Q}_1}\mathbb{E}[d_\Delta(G_n, G^*_{\mathbb{Q}})] \geq cn^{\frac{1}{2\kappa-1+\rho}} \quad \text{a.s.},$$

for some constant $c > 0$ and some finite set $\mathfrak{Q}_1 \subseteq \mathfrak{Q}$. Let $K \geq 2$ be an integer and let

$$i_{\text{opt}} := \arg\max_{i=1,\dots,r_2}\frac{t_i}{\beta^*_{2,i}}.$$

As in the proof of Theorem 3.4.1 define $\phi : \mathbb{R} \to [0,1]$ to be an infinitely many times differentiable function with the following two properties:

- $\phi(t) = 0$ for $|t| \geq 1$,
- $\phi(0) = 1$.

Note that ϕ^α also fulfills both properties for any $\alpha > 0$, though it may not be infinitely many times differentiable. For $i \in \{1,\dots,K\}^{t_{i_{\text{opt}}}}$ define

$$\phi_i : [0,1]^{d_1-1} \to [0,1],\ \phi_i(y) = k_1K^{-\beta^*_{2,i_{\text{opt}}}}\prod_{j=1}^{t_{i_{\text{opt}}}}\phi^\alpha\left(K\left(y_j - \frac{2i_j-1}{K}\right)\right)$$

for $\alpha := \prod_{k=i_{\text{opt}}}^{r_2}\min\{\beta_k, 1\}$ and some $0 < k_1$ small enough. Define

$$W := \prod_{i\in\{1,\dots,K\}^{d-1}}\{0,1\}.$$

For $w \in W$ let

$$\gamma_w : [0,1]^{d_1-1} \to [0,1],\ \gamma_w(y) = \sum_{i \in \{1,\dots,K\}^{t_{i_{\text{opt}}}}} w_i \phi_i(y).$$

Now, for $w \in W$ we define $\mathbb{Q}_w$ as before. The marginal distribution $\mathbb{Q}_X$ is the uniform distribution on $[0,1]^d$ and

$$\begin{aligned} f_{\mathbb{Q}_w}(x) := &\frac{1}{2}\left(1 + k_2\left(\gamma_w(x_{-1}) - x_1\right)^{\beta_1}\right)\mathbb{1}\left(x_1 \le \gamma_w(x_{-1})\right) \\ &+ \frac{1}{2}\left(1 - k_2 x_1^{\beta_1}\right)\mathbb{1}\left(0 < x_1 \le \gamma_{1-w}(x_{-1})\right) \\ &+ \frac{1}{2}\left(1 - k_3\left(x_1 - \gamma_1(x_{-1})\right)^{\beta_1}\right)\mathbb{1}(\gamma_1(x_{-1}) < x_1) \end{aligned}$$

for some $k_2, k_3 > 0$. Note that for $k_1 > 0$ small enough $\gamma := \gamma_w \in \mathcal{G}_{r_2,t,\beta_2,B_2,d'}$ by defining

$$\begin{aligned} \gamma_i(y) &= (y_1, \dots, y_{t_i}, 0, \dots, 0),\ \text{for } i < i_{\text{opt}}, \\ \gamma_i(y) &= (\psi(y), 0, \dots, 0),\ \text{for } i = i_{\text{opt}}, \\ \gamma_i(y) &= \left(k_1^{\alpha_i} y_1^{\min\{\beta_i,1\}}, 0, \dots, 0\right),\ \text{for } i > i_{\text{opt}}, \end{aligned}$$

where

$$\begin{aligned} \psi(y) &:= \sum_{i \in \{1,\dots,K\}^{t_{i_{\text{opt}}}}} w_i k_1^{\alpha_{i_{\text{opt}}}} K^{-\beta_{2,i_{\text{opt}}}} \prod_{j=1}^{t_{i_{\text{opt}}}} \phi\left(K\left(y_j - \frac{2i_j - 1}{K}\right)\right), \\ \alpha_i &:= \frac{1}{(r_2 - i_{\text{opt}} + 1)\prod_{k=i+1}^{r_2} \min\{\beta_k, 1\}} \end{aligned}$$

for $i_{\text{opt}} \le i \le r_2$. The rest of the proof is analogous to the proof of Theorem 3.4.1. □

7.3 Random Planted Forest: a directly interpretable tree ensemble

Proof of Theorem 4.5.1 In the proof we denote different constants by C. The meaning of C may change, also in the same formula. In this proof we omit the index v in the notation.

We rewrite (4.5.1) as

$$\widehat{m}^s(x) = \widehat{\overline{m}}^s(x_{t_s}) + \widehat{\Psi}^s \widehat{m}^{s-1}(x), \tag{7.3.1}$$

where $\widehat{\overline{m}}^s = \widehat{\overline{m}}^s_{t_s}$, $\widehat{\Psi}^s = \widehat{\Psi}^s_{t_s}$ and where

$$\begin{aligned} \widehat{\Psi}^s_t m(x) &= m(x) - m_t(x_t) \\ &- \sum_{t' \in T_r, t' \neq t} \int_{(0,1]^{|t' \setminus t|}} \frac{\widehat{p}^s_{t \cup t'}(x_t, u_{t' \setminus t})}{\widehat{p}^s_t(x_t)} m_{t'}(x_{t' \cap t}, u_{t' \setminus t}) \mathrm{d}u_{t' \setminus t} \end{aligned}$$

for an additive function $m(x) = \sum_{t \in T_r} m_t(x_t)$. Iterative application of (7.3.1) gives for $1 \leq S_1 < S_2 \leq S$

$$\begin{aligned} \widehat{m}^{S_2} &= \widehat{\widehat{m}}^{S_2} + \widehat{\Psi}^{S_2} \widehat{\widehat{m}}^{S_2-1} + \widehat{\Psi}^{S_2} \widehat{\Psi}^{S_2-1} \widehat{\widehat{m}}^{S_2-2} \\ &\quad + ... + \widehat{\Psi}^{S_2} ... \widehat{\Psi}^{S_1+1} \widehat{\widehat{m}}^{S_1} + \widehat{\Psi}^{S_2} ... \widehat{\Psi}^{S_1} \widehat{m}^{S_1-1}. \end{aligned} \tag{7.3.2}$$

We use this equality in the proofs of the following lemmas for the choices $S_1 = 1, S_2 = S^*$ and $S_1 = S^*, S_2 = S$ with $S^* = S - J'$.

Lemma 7.3.1. *It holds that*

$$\frac{1}{n} \sum_{i=1}^{n} \widehat{m}^{S^*-1}(X_i)^2 \quad \leq \quad S^{1/2} \frac{1}{n} \sum_{i=1}^{n} Y_i^2.$$

Proof. It can be easily shown that for $t \in T_r$

$$\frac{1}{n} \sum_{i=1}^{n} \widehat{\widehat{m}}_t(X_i)^2 \leq \frac{1}{n} \sum_{i=1}^{n} Y_i^2. \tag{7.3.3}$$

Furthermore, for any function m we have

$$\frac{1}{n} \sum_{i=1}^{n} \left(\widehat{\Psi}^s m \right) (X_i)^2 \leq \frac{1}{n} \sum_{i=1}^{n} m(X_i)^2. \tag{7.3.4}$$

For a proof of (7.3.4) consider the space of functions $m : [0,1]^d \to \mathbb{R}$ endowed with the pseudo metric

$$\|m\|_{n,2}^2 = \frac{1}{n} \sum_{i=1}^{n} m(X_i)^2.$$

Then $I - \widehat{\Psi}^s$ is the orthogonal projection onto the space of piecewise constant functions that are constant on the rectangles $\mathcal{I}_{t,l}^s : l = 1, ..., L_{t,s}$. Furthermore, $\widehat{\Psi}^s$ is the orthogonal projection onto the orthogonal complement of this space. This shows (7.3.4). The statement of the lemma now follows from (7.3.3)–(7.3.4) by application of (7.3.2) with $S_1 = 1$ and $S_2 = S^* - 1$. □

This lemma can be used to show a bound for the $L_2(p)$-norm of $\widehat{m}^{S^*-1}$ which we denote by $\|\widehat{m}^{S^*-1}\|$

Lemma 7.3.2. *It holds that with probability tending to one*

$$\|\widehat{m}^{S^*-1}\|^2 = \int \widehat{m}^{S^*-1}(x)^2 p(x) \mathrm{d}x \quad \leq \quad (1 - C\eta_{1,n})^{-1} S^{1/2} \frac{1}{n} \sum_{i=1}^{n} Y_i^2.$$

Proof. For $t \in T_r$ we get by application of (A6) that

$$\begin{aligned}\int \widehat{m}_t^{S^*-1}(x_t)^2 p(x)\mathrm{d}x &\leq \int \widehat{m}_t^{S^*-1}(x_t)^2 \widehat{p}_t^{S^*-1}(x_t)\mathrm{d}x_t \\ &\quad + \int \widehat{m}_t^{S^*-1}(x_t)^2 |\widehat{p}_t^{S^*-1} - p_t|(x_t)\mathrm{d}x_t \\ &\leq \frac{1}{n}\sum_{i=1}^{n} \widehat{m}_t^{S^*-1}(X_{t,i})^2 \\ &\quad + C\eta_{1,n} \int \widehat{m}_t^{S^*-1}(x_t)^2 p_t(x_t)\mathrm{d}x_t.\end{aligned}$$

The statement of the lemma now follows by application of Lemma 7.3.1. □

Lemma 7.3.3. *It holds for some* $C > 0$ *and* $0 < \rho < 1$ *that with probability tending to one*

$$\|\widehat{\Psi}^S \circ ... \circ \widehat{\Psi}^{S^*}\| \leq C\rho^{C_{A3}\log n}.$$

Here for an operator Ψ mapping $L_2(p)$ into $L_2(p)$ we denote the operator norm $\|\Psi\| = \sup\{\|\Psi m\| : m \in L_2(p), \|m\| = 1\}$ by $\|\Psi\|$. Before we come to the proof of this lemma let us discuss its implications. Using the representation (7.3.1) we get from Lemmas 7.3.2, 7.3.3 that for all $R > 0$ we can choose a constant C_R such that if C_{A3} is large enough with probability tending to one

$$\|\widehat{m}^S - \widehat{m}^{1,S}\| \leq C_R n^{-R}, \tag{7.3.5}$$

where

$$\widehat{m}^{1,S} = \widehat{\widehat{m}}^S + \widehat{\Psi}^S \widehat{\widehat{m}}^{S-1} + ... + \widehat{\Psi}^S ... \widehat{\Psi}^{S^*+1}\widehat{\widehat{m}}^{S^*}.$$

Note that $\widehat{m}^{1,S}$ only depends on the growth history of the tree in the last $C'_J(\log n)^2$ steps. Thus we have shown that approximately the same holds for the tree estimators $\widehat{m}^S$. Below we go a step further and show that the tree family estimator approximately only depends on the data averages in the terminal leaves and in particular not on the growth of the tree family in the past. Before we come to this refinement we first give a proof of Lemma 7.3.3. For this purpose we introduce population analogues Ψ_{t_s} of the operators $\widehat{\Psi}^s$ and we discuss some theory on backfitting estimators in additive interaction models. We consider the following subspaces of functions $m : [0,1]^d \to \mathbb{R}$ with $E[m^2(X_i)] < \infty$ for $t \in T_r$

$$\begin{aligned}\mathcal{H} &= \{m : [0,1]^d \to \mathbb{R} \mid E[m^2(X_i)] < \infty\}, \\ \mathcal{H}_t &= \{m \in \mathcal{H} \mid m(x) = m_t(x_t) \text{ for some function } m_t : [0,1]^{|t|} \to \mathbb{R}\}, \\ \mathcal{H}_{add} &= \{m \in \mathcal{H} \mid m(x) = \sum_{t \in T_r} m_t(x_t) \text{ for some functions} \\ &\qquad m_t \in \mathcal{H}_t, t \in T_r\}.\end{aligned}$$

In particular, $\mathcal{H}_\emptyset$ is the subspace of $\mathcal{H}$ that contains only constant functions. In abuse of notation for a function $m_t \in \mathcal{H}_t$ we also write $m_t(x)$ with $x \in [0,1]^d$ instead of $m_t(x_t)$. Thus m_t can be interpreted as a function with domain $[0,1]^{|t|}$ or with domain $[0,1]^d$. The projection of $\mathcal{H}$ onto $\mathcal{H}_t$ is denoted by Π_t. For $\Psi_t = I - \Pi_t$ and $m = \sum_{t\in T_r} m_t$ with $m_t \in \mathcal{H}_t$ $(t \in T_r)$ one gets

$$\Psi_t m(x) = \sum_{w\in T_r\setminus\{t\}} m_w(x_w) + m_t^*(x_t)$$

with

$$m_t^*(x_t) = -\sum_{w\in T_r, w\neq t} \int \frac{p_{t\cup w}(x_{t\cup w})}{p_t(x_t)} m_w(x_w)\mathrm{d}x_{w\setminus t}.$$

We now consider operators K of the form $\Psi_{t^1} \circ \ldots \circ \Psi_{t^{k_K}}$ with $\{t^1, ..., t^{k_K}\} \supseteq T_r$. We call these operators complete. We argue that there exists a constant $\gamma < 1$ such that for all complete operators K of this form we have

$$\|K\|_{add} = \sup\{\|Km\| : m \in \mathcal{H}_{add}, \|m\| \leq 1\} < \gamma. \tag{7.3.6}$$

Note that in particular γ does not depend on the order of $t^1, ..., t^{k_K}$. Our notation is a little bit sloppy because in the representation $\sum_{t\in T_r} m_t$ of m the summands are not uniquely defined because $t\cap t'$ may be nonempty for some $t, t' \in T_r$, $t \neq t'$. A more appropriate notation would be to define K as an operator mapping $\prod_{t\in T_r} \mathcal{H}_t$ into $\prod_{t\in T_r} \mathcal{H}_t$ and to endow this space with the pseudo norm $\|\sum_{t\in T_r} m_t\|$. For a proof of (7.3.6) we show that $\sum_{t\in T} \mathcal{H}_t$ are closed subspaces of $\mathcal{H}$ for all choices of $T \subset T_r$. In particular, by a result of Deutsch (1985), see also Appendix A.4 in Bickel, Klaassen, Ritov and Wellner (1993), this implies that $\rho(\mathcal{H}_{t_j}, \mathcal{H}_{t_j+1} + ... + \mathcal{H}_{t_{k^K}}) < 1$ for $1 \leq j \leq t_{k^K-1}$, where for two linear subspaces L_1 and L_2 of $\mathcal{H}$ the quantity $\rho(L_1.L_2)$ is the cosine of the minimal angle between L_1 and L_2, i.e. $\rho(L_1.L_2) = \sup\{\int h_1(x)h_2(x)p(x)\mathrm{d}x : h_j \in L_j \cap (L_1 \cap L_2)^\perp, \int h_j^2(x)p(x)\mathrm{d}x \leq 1 \text{ for } j = 1,2\}$.
According to a result of Smith, Solomon and Wagner (1977) this implies that for an operator K of the above form we have that

$$\|K\|_{add} \leq 1 - \prod_{j=1}^{k^K} \sin^2(\alpha_j),$$

where α_j is chosen such that $\cos(\alpha_j) = \rho(\mathcal{H}_{t_j}, \mathcal{H}_{t_{j+1}} + ... + \mathcal{H}_{t_{k^K}})$. We remark that this bound is also valid if the space $\mathcal{H}_{t_j}$ is identical to the same space $\mathcal{H}_t$ for several choices of j. In this case we have $\sin^2(\alpha_j) = 1$ for all such values of the index j with the exception of the last appearance of $\mathcal{H}_t$. Because there are only finitely many ways to order $|T_r|$ elements we get by the last remark that (7.3.6) holds with some $\gamma < 1$ for all operators K. For a proof of (7.3.6) it remains to show that $\sum_{t\in T} \mathcal{H}_t$ are closed subspaces of $\mathcal{H}$ for all choices of $T \subset T_r$. For this claim we argue that $\sum_{t\in T} \bar{\mathcal{H}}_t$ are closed subspaces of $\mathcal{H}$ for all choices of $T \subset T_r$, where $\bar{\mathcal{H}}_t = \{h \in \mathcal{H}_t : \int h(x)\mathrm{d}x_j = 0 \text{ for } j \in t\}$. According to Proposition 2 in the supplement material A.4 of Bickel, Klaassen, Ritov and Wellner (1993) this follows if there exists some $c > 0$ such that $\int(\sum_{t\in T} h_t(x_t))^2 p(x)\mathrm{d}x \geq 1$ implies that $\int h_t(x_t)^2 p(x)\mathrm{d}x \geq c$ for some $t \in T$. This can be easily verified with $c = C(\max_{x\in[0,1]^d} p(x))^{-1}|T_r|^{-1}$ by noting that for $h_t \in \bar{\mathcal{H}}_t$ it holds that

$$\begin{aligned}\int\left(\sum_{t\in T} h_t(x_t)\right)^2 p(x)\mathrm{d}x &\leq \max_{x\in[0,1]^d} p(x)\int\left(\sum_{t\in T} h_t(x_t)\right)^2 \mathrm{d}x\\ &= \max_{x\in[0,1]^d} p(x)\sum_{t\in T}\int h_t(x_t)^2\mathrm{d}x\\ &\leq C\max_{x\in[0,1]^d} p(x)\sum_{t\in T}\int h_t(x_t)^2 p(x)\mathrm{d}x\end{aligned}$$

In particular, one can use (7.3.6) to show that with

$$\bar{m}_t(x) = E[Y_i|X_{i,t} = x_t] = \sum_{t'\in T_r}\int m_{t'}(x_{t'})\frac{p_{t'\cup t}(x_{t'\cup t})}{p_t(x_t)}\mathrm{d}x_{t'\setminus t}$$

we have for $\mu^* \in \mathcal{H}_{add}$ with some $0 < \gamma < 1$, and some $C > 0$

$$\|m - \mu\| \leq C\gamma^{S^*}(\|\mu^*\| + 1),$$

where

$$\mu = \bar{m}_{t^*_{v,1}} + \Psi_{t^*_{v,1}}\bar{m}_{t^*_{v,2}} + \Psi_{t^*_{v,1}}\Psi_{t^*_{v,2}}\bar{m}_{t^*_{v,3}} + ... + \Psi_{t^*_{v,1}} \circ ... \circ \Psi_{t^*_{v,S^*-1}}\mu^*$$

with $t^*_{v,s} = t_{v,S-s+1}$. Note that

$$m - \mu = \Psi_{t^*_{v,1}} \circ ... \circ \Psi_{t^*_{v,S^*-1}}(m - \mu^*).$$

Proof of Lemma 7.3.3. By Assumption (A6) we have by application of Cauchy-Schwarz inequality that

$$\|\widehat{\Psi}^s - \Psi_{t_s}\| \leq C\delta_{2,n} \tag{7.3.7}$$

with probability tending to one. This implies that, with probability tending to one,

$$\|\widehat{\Psi}^s\| \leq 1 + C\delta_{2,n}$$

and we get by a telescope argument that, with probability tending to one, for $1 \leq j \leq J \leq C_{A3}\log n + 1$

$$\begin{aligned}\|\widehat{\Psi}^{s_j} \circ ... \circ \widehat{\Psi}^{s_{j-1}+1} - \Psi_{t_{s_j}} \circ ... \circ \Psi_{t_{s_{j-1}+1}}\| &\leq CJ'(1 + C\delta_{2,n})^{J'-1}\delta_{2,n}\\ \leq CC'_{A3}(\log n)^2(1 + C\delta_{2,n})^{C'_{A3}(\log n)^2}\delta_{2,n} &\leq C(\log n)^2\delta_{2,n}.\end{aligned}$$

Because by Assumption (A3) $\Psi_{t_{s_j}} \circ ... \circ \Psi_{t_{s_{j-1}+1}}$ is a complete operator with high probability, we get from (7.3.6) that, with probability tending to one, for $1 \leq j \leq J \leq C_{A3} \log n + 1$

$$\|\widehat{\Psi}^{s_j} \circ ... \circ \widehat{\Psi}^{s_{j-1}+1}\| \quad \leq \quad \gamma + C(\log n)^2 \delta_{2,n}.$$

This inequality can be used to show the bound on $\|\widehat{\Psi}^{S} \circ ... \circ \widehat{\Psi}^{S^*}\|$, claimed in Lemma 7.3.3. $\Box$

As discussed above, Lemma 7.3.3 implies (7.3.5). We now approximate $\widehat{m}^{1,S}$ by $\widehat{m}^{2,S}$ where

$$\widehat{m}^{2,S} \quad = \quad \widehat{\widehat{m}}^{S}_{t_S} + \widehat{\Psi}^{S}_{t_S} \widehat{\widehat{m}}^{S}_{t_{S-1}} + \widehat{\Psi}^{S}_{t_S} \widehat{\Psi}^{S}_{t_{S-1}} \widehat{\widehat{m}}^{S}_{t_{S-2}} + ... + \widehat{\Psi}^{S}_{t_S} ... \widehat{\Psi}^{S}_{t_{S^*+1}} \widehat{\widehat{m}}^{S}_{t_{S^*}}.$$

Note that $\widehat{m}^{2,S}$ differs from $\widehat{m}^{1,S}$ by having always the superindex S for the operators Ψ and the functions $\widehat{\widehat{m}}$. In the following lemma we compare $\widehat{m}^{S}$ and $\widehat{m}^{2,S}$. The bound can be shown by similar arguments as in the proof of (7.3.5). In this proof one uses Assumption (A5) instead of (7.3.7). One gets that for all $R > 0$ we can choose a constant C_R such that if C_{A3} in Assumption (A3) is large enough with probability tending to one $\|\widehat{m}^{S} - \widehat{m}^{2,S}\| \leq C_R n^{-R} + CC'_{A3}(1+C\delta_{2,n})^{C'_{A3}(\log n)^2} \delta_{1,n} \leq C_R n^{-R} + C\delta_{1,n}$. If R is chosen large enough we get the following lemma, see Assumption (A4).

Lemma 7.3.4. *If C_{A3} in Assumption (A3) is large enough it holds that with probability tending to one*

$$\|\widehat{m}^{S} - \widehat{m}^{2,S}\| \leq C\delta_{1,n}.$$

We now define $\widehat{m}^{3,S} = \sum_{t \in T_r} \widehat{m}^{3,S}_t$ as a minimizer of

$$\sum_{i=1}^{n} \left(Y_i - \sum_{t \in T_r} m_t(X_{i,t}) \right)^2$$

over all function $m_t : (0,1]^{|t|} \to \mathbb{R}$ that are piecewise constant on the rectangles $\mathcal{I}^{S}_{t,l} : l = 1, ..., L_{t,S}$. Then we have $\widehat{\Psi}^{S}_t \widehat{m}^{3,S} + \widehat{\widehat{m}}^{S}_t = \widehat{m}^{3,S}$ for all $t \in T_r$ which implies that

$$\widehat{m}^{3,S} \quad = \quad \widehat{\widehat{m}}^{S}_{t_S} + \widehat{\Psi}^{S}_{t_S} \widehat{\widehat{m}}^{S}_{t_{S-1}} + \widehat{\Psi}^{S}_{t_S} \widehat{\Psi}^{S}_{t_{S-1}} \widehat{\widehat{m}}^{S}_{t_{S-2}} + ... + \widehat{\Psi}^{S}_{t_S} ... \widehat{\Psi}^{S}_{t_{S^*+1}} \widehat{m}^{3,S}_{t_{S^*}}. \qquad (7.3.8)$$

This shows that

$$\widehat{m}^{3,S} - \widehat{m}^{2,S} \quad = \quad \widehat{\Psi}^{S}_{t_S} ... \widehat{\Psi}^{S}_{t_{S^*+1}} (\widehat{m}^{3,S}_{t_{S^*}} - \widehat{\widehat{m}}^{S}_{t_{S^*}}).$$

This equation can be used to prove the following result:

Lemma 7.3.5. *If C_{A3} in Assumption (A3) is large enough it holds that with probability tending to one*

$$\|\widehat{m}^{S} - \widehat{m}^{3,S}\| \leq C\delta_{1,n}.$$

We now decompose $\widehat{m}^{3,S}$ into a stochastic and a bias term

$$\widehat{m}^{3,S} - m^0 = \widehat{m}^{A,S} + \widehat{m}^{B,S} - m^0,$$

where $\widehat{m}^{A,S} = \sum_{t \in T_r} \widehat{m}_t^{A,S}$ minimizes

$$\sum_{i=1}^n \left(\varepsilon_i - \sum_{t \in T_r} m_t(X_{i,t}) \right)^2$$

over all function $m_t : (0,1]^{|t|} \to \mathbb{R}$ that are piecewise constant on the rectangles $\mathcal{I}_{t,l}^S : l = 1, ..., L_{t,S}$ and $\widehat{m}^{B,S} = \sum_{t \in T_r} \widehat{m}_t^{B,S}$ minimizes

$$\sum_{i=1}^n \left(m^0(X_i) - \sum_{t \in T_r} m_t(X_{i,t}) \right)^2$$

over the same class of piecewise constant functions. We see that $\widehat{m}^{A,S}$ is the projection of ε onto an $(S+1)$-dimensional linear subspace of $\mathbb{R}^n$. We conclude that

$$E\left[\frac{1}{n} \sum_{i=1}^n \left| \sum_{t \in T_r} \widehat{m}_t^{A,S}(X_{i,t}) \right|^2 \right] \quad \leq \quad (S+1) n^{-1} \sigma^2. \tag{7.3.9}$$

For the study of the bias term define $\bar{m}(x) = \sum_{t \in T_r} \bar{m}_t(x_t)$ with

$$\bar{m}_t(x_t) = \frac{1}{n |\mathcal{I}_{t,l}^S|_n} \sum_{i: X_{i,t} \in \mathcal{I}_{t,l}^S} m_t^0(X_{i,t})$$

for $x_t \in \mathcal{I}_{t,l}^S$. Now, by definition of $\widehat{m}^{B,S}$, we have that

$$\frac{1}{n} \sum_{i=1}^n \left| \sum_{t \in T_r} \widehat{m}_t^{B,S}(X_{i,t}) - m^0(X_i) \right|^2 \leq \frac{1}{n} \sum_{i=1}^n \left| \bar{m}(X_i) - m^0(X_i) \right|^2.$$

Furthermore, we have by an application of Assumption (A4) that

$$\frac{1}{n} \sum_{i=1}^n \left| \bar{m}(X_i) - m^0(X_i) \right|^2 \leq C \frac{1}{n} \sum_{i=1}^n \sum_{t \in T_r} \sum_{k \in t} \left(b_{t,k,l_t(X_{i,t})}^S - a_{t,k,l_t(X_{i,t})}^S \right)^2 \leq C \delta_{1,n}^2.$$

Using the same arguments as in the proof of Lemma 7.3.2 one gets the same bound with empirical norm replaced by the $L_2(P)$-norm $\| \cdot \|$. This concludes the proof of the theorem.

Proof of Theorem 4.5.2 We now introduce the super index v again which denotes the number of the respective tree family. Note that the bound of Lemma 7.3.5 holds uniformly over $1 \leq v \leq$

V. We can decompose $\widehat{m}^{3,S,v}$ into a stochastic and a bias term

$$\widehat{m}^{3,S,v} - m^0 = \widehat{m}^{A,S,v} + \widehat{m}^{B,S,v} - m^0$$

and we get from (7.3.9) that

$$E\left[\frac{1}{n}\sum_{i=1}^{n}\left|\sum_{t\in T_r}\frac{1}{V}\sum_{v=1}^{V}\widehat{m}_t^{A,S,v}(X_{i,t})\right|^2\right] \;\leq\; (S+1)n^{-1}\sigma^2.$$

For the treatment of the averaged bias term note that for $t \in T_r$

$$\widehat{m}_t^{B,S,v}(x_t) = \widehat{\widehat{m}}_t^{B,S,v}(x_t) - \sum_{t'\neq t}\int \frac{\widehat{p}_{t\cup t'}^{S,v}(x_{t\cup t'})}{\widehat{p}_t^{S,v}(x_t)}\widehat{m}_{t'}^{B,S,v}(x_{t'})\mathrm{d}x_{t'\setminus t}$$

where $\widehat{\widehat{m}}_t^{B,S,v}(x_t) = \frac{1}{n|\mathcal{I}_{t,l}^{S,v}|_n}\sum_{i:X_{i,t}\in\mathcal{I}_{t,l}^{S,v}} m^0(X_i)$ for $x_t \in \mathcal{I}_{t,l}^{S,v}$. Now by subtracting $m_t^0(x_t)$ we get

$$\begin{aligned}\widehat{m}_t^{B,S,v}(x_t) - m_t^0(x_t) &= \widehat{\widehat{m}}_t^{B,S,v}(x_t) - \bar{m}_t^0(x_t)\\ &-\sum_{t'\neq t}\int\left(\frac{\widehat{p}_{t\cup t'}^{S,v}(x_{t\cup t'})}{\widehat{p}_t^{S,v}(x_t)}\widehat{m}_{t'}^{B,S,v}(x_{t'}) - \frac{p_{t\cup t'}(x_{t\cup t'})}{p_t(x_t)}m_{t'}^0(x_{t'})\right)\mathrm{d}x_{t'\setminus t}\end{aligned}$$

where $\bar{m}_t^0(x_t) = \int \frac{p(x)}{p_t(x_t)}m^0(x)\mathrm{d}x_{\{1,\dots,d\}\setminus t}$. This can be rewritten as

$$\begin{aligned}\widehat{m}_t^{B,S,v}(x_t) - m_t^0(x_t) &= \widehat{\widehat{m}}_t^{B,S,v}(x_t) - \bar{m}_t^0(x_t) + \Delta_{1,t}^v(x_t)\\ &-\sum_{t'\neq t}\int\frac{p_{t\cup t'}(x_{t\cup t'})}{p_t(x_t)}\left(\widehat{m}_{t'}^{B,S,v}(x_{t'}) - m_{t'}^0(x_{t'})\right)\mathrm{d}x_{t'\setminus t} + \Delta_{2,t}^v(x_t),\end{aligned}$$

where

$$\begin{aligned}\Delta_{1,t}^v(x_t) = &-\left(\widehat{p}_t^{S,v}(x_t) - p_t(x_t)\right)\sum_{t'\neq t}\int\frac{p_{t\cup t'}(x_{t\cup t'})}{p_t^2(x_t)}m_{t'}^0(x_{t'})\mathrm{d}x_{t'\setminus t}\\ &-\sum_{t'\neq t}\int\frac{\widehat{p}_{t\cup t'}^{S,v}(x_{t\cup t'}) - p_{t\cup t'}(x_{t\cup t'})}{p_t(x_t)}m_{t'}^0(x_{t'})\mathrm{d}x_{t'\setminus t},\end{aligned}$$

and

$$\begin{aligned}\Delta_{2,t}^{v}(x_t) = &\sum_{t' \neq t} \int \left(\frac{\widehat{p}_{t\cup t'}^{S,v}(x_{t\cup t'})}{\widehat{p}_t^{S,v}(x_t)} - \frac{p_{t\cup t'}(x_{t\cup t'})}{p_t(x_t)} \right) \left(\widehat{m}_{t'}^{B,S,v}(x_{t'}) - m_{t'}^0(x_{t'}) \right) \mathrm{d}x_{t'\setminus t} \\ &- \frac{\left(\widehat{p}_t^{S,v}(x_t) - p_t(x_t)\right)^2}{p_t^2(x_t)\widehat{p}_t^{S,v}(x_t)} \sum_{t' \neq t} \int \widehat{p}_{t\cup t'}^{S,v}(x_{t\cup t'}) \widehat{m}_{t'}^{B,S,v}(x_{t'}) \mathrm{d}x_{t'\setminus t} \\ &+ \frac{\left(\widehat{p}_t^{S,v}(x_t) - p_t(x_t)\right)}{p_t^2(x_t)} \sum_{t' \neq t} \int p_{t\cup t'}(x_{t\cup t'}) (\widehat{m}_{t'}^{B,S,v}(x_{t'}) - m_{t'}^0(x_{t'})) \mathrm{d}x_{t'\setminus t}.\end{aligned}$$

By averaging the integral equations over v we get

$$\begin{aligned}\widehat{m}_t^{B,S,+}(x_t) - m_t^0(x_t) = &\,\widehat{\bar{m}}_t^{B,S,+}(x_t) - \bar{m}_t^0(x_t) + \Delta_{1,t}(x_t) \\ &- \sum_{t' \neq t} \int \frac{p_{t\cup t'}(x_{t\cup t'})}{p_t(x_t)} \left(\widehat{m}_{t'}^{B,S,+}(x_{t'}) - m_{t'}^0(x_{t'}) \right) \mathrm{d}x_{t'\setminus t} + \Delta_{2,t}(x_t),\end{aligned} \tag{7.3.10}$$

where $m_t^{B,S,+} = V^{-1}\sum_{v=1}^V m_t^{B,S,v}$, $\widehat{\bar{m}}_t^{B,S,+} = V^{-1}\sum_{v=1}^V \widehat{\bar{m}}_t^{B,S,v}$, $\Delta_{1,t} = V^{-1}\sum_{v=1}^V \Delta_{1,t}^v$ and $\Delta_{2,t} = V^{-1}\sum_{v=1}^V \Delta_{2,t}^v$.

We now argue that with probability tending to one

$$\|\Delta_{1,t}\|_1 \leq C\delta_{4,n}, \tag{7.3.11}$$

$$\|\Delta_{2,t}\|_1 \leq C(\delta_{2,n}(\delta_{1,n} + S^{1/2}n^{-1/2}) + \delta_{3,n}^2), \tag{7.3.12}$$

$$\|\widehat{\bar{m}}_t^{B,S,+} - \bar{m}_t^0\|_1 \leq C(\delta_{4,n} + \delta_{3,n}^2). \tag{7.3.13}$$

Note that with $\Delta_t = \Delta_{1,t} + \Delta_{2,t} + \widehat{\bar{m}}_t^{B,S,+} - \bar{m}_t^0$ we can rewrite (7.3.10) as

$$\widehat{m}^{B,S,+} - m^0 = \Delta_t + \Psi_t(\widehat{m}^{B,S,+} - m^0). \tag{7.3.14}$$

Order the elements of T_r as $t_1, ..., t_{2^r}$. Iterative application of (7.3.14) gives that

$$\widehat{m}^{B,S,+} - m^0 = \Delta_+ + \Psi_+(\widehat{m}^{B,S,+} - m^0), \tag{7.3.15}$$

where $\Delta_+ = \Delta_{t_1} + \Psi_{t_1}\Delta_{t_2} + ... + \Psi_{t_1}...\Psi_{t_{2^r-1}}\Delta_{t_{2^r}}$ and $\Psi_+ = \Psi_{t_1}...\Psi_{t_{2^r}}$. Because Ψ_+ is a complete operator we get from (7.3.6) that $\|\Psi_+\mu\| \leq \gamma\|\mu\|$ for $\mu \in \mathcal{H}_{add}$ with $\gamma < 1$. Furthermore, one can easily verify that $\|\Psi_+\mu\| \leq C\|\mu\|_1$ for $\mu \in \mathcal{H}_{add}$ with some constant $C > 0$. With the help of (7.3.11)–(7.3.13) this shows that $\|\Delta_+\|_1 \leq C\delta_n$ and $\|\Psi_+\Delta_+\| \leq C\delta_n$ with probability tending to one where $\delta_n = \delta_{2,n}(\delta_{1,n} + S^{1/2}n^{-1/2}) + \delta_{3,n}^2 + \delta_{4,n}$. From (7.3.15) we get that $\widehat{m}^{B,S,+} - m^0 = \Delta_+ + \Delta_{++}$ with $\Delta_{++} = \sum_{k=1}^\infty \Psi_+^k\Delta_+$. It holds with probability tending to one that $\|\Delta_{++}\| \leq C\delta_n$ which implies that $\|\Delta_{++}\|_1 \leq C\delta_n$. We conclude that $\|\widehat{m}^{B,S,+} - m^0\|_1 \leq C\delta_n$ with probability tending to one. This shows the statement of Theorem 4.5.2. It remains to verify (7.3.11)–(7.3.13). The first claim follows directly from Assumption (A7). For the proof of

(7.3.12) one makes use of the bounds for $\widehat{m}_{t'}^{B,S,v} - m_{t'}^0$, $\widehat{p}_{t\cup t'}^{S,v} - p_{t\cup t'}$ and $\widehat{p}_t^{S,v} - p_t$ which we have shown during the proof of Theorem 4.5.1 and which carry over to the averaged values. For the proof of (7.3.13) note that

$$\widehat{\bar{m}}_t^{B,S,+}(x_t) - \bar{m}_t^0(x_t) = \Delta_{3,t}(x_t) + \Delta_{4,t}(x_t)$$

with

$$\begin{aligned}\Delta_{3,t}(x_t) &= \frac{\widehat{p}_t^{S,+}(x_t) - p_t(x_t)}{p_t(x_t)} \sum_{t'\neq t} \int p_{t\cup t'}(x_{t\cup t'}) m_{t'}^0(x_{t'}) \mathrm{d}x_{t'\setminus t} \\ &+ \sum_{t'\neq t} \int \frac{\widehat{p}_{t\cup t'}^{S,+}(x_{t\cup t'}) - p_{t\cup t'}(x_{t\cup t'})}{p_t(x_t)} m_{t'}^0(x_{t'}) \mathrm{d}x_{t'\setminus t}, \\ \Delta_{4,t}(x_t) &= -V^{-1} \sum_{v=1}^{V} \left(\widehat{p}_t^{S,v}(x_t) - p_t(x_t) \right) \\ &\times \sum_{t'\neq t} \int \frac{\widehat{p}_{t\cup t'}^{S,v}(x_{t\cup t'}) - p_{t\cup t'}(x_{t\cup t'})}{p_t^2(x_t)} m_{t'}^0(x_{t'}) \mathrm{d}x_{t'\setminus t} \\ &+ V^{-1} \sum_{v=1}^{V} \frac{\left(\widehat{p}_t^{S,v}(x_t) - p_t(x_t) \right)^2}{p_t^2(x_t) \widehat{p}_t^{S,v}(x_t)} \sum_{t'\neq t} \int \widehat{p}_{t\cup t'}^{S,v}(x_{t\cup t'}) m_{t'}^0(x_{t'}) \mathrm{d}x_{t'\setminus t}.\end{aligned}$$

Using similar arguments as above, one can easily verify that $\|\Delta_{3,t}\|_{t,*} \leq C\delta_{4,n}$ and $\|\Delta_{4,t}\| \leq C\delta_{3,n}^2$, with probability tending to one. This concludes the proof of the theorem.

7.4 Unifying Local and Global Model Explanations by Functional Decomposition of Low Dimensional Structures

7.4.1 Lemmata

Lemma 7.4.1. *The solution m_S^* described in theorem 5.3.1 can be re-written as*

$$m_S^*(x_S) = \sum_{V\subseteq S} (-1)^{|S\setminus V|} \int m^{(0)}(x) p_{-V}(x_{-V}) \mathrm{d}x_{-V},$$

where $m^{(0)}(x) = \sum_S m_S^{(0)}(x_S)$. In particular m_S^* does not depend on the particular identification of $m^{(0)}$.

Proof. We consider a fixed $S \subseteq \{1, \ldots, d\}$. We make use of the fact that for a set $T \not\supseteq S$

$$\sum_{V\subseteq S} (-1)^{|S\setminus V|} \int m_T^{(0)}(x_T) p_{-V}(x_{-V}) \mathrm{d}x_{-V} = 0, \tag{7.4.1}$$

and for a set $T \subseteq \{1,\dots,d\}, T \supseteq S$

$$\{U : T \setminus S \subseteq U \subseteq T\} = \{T \setminus V : V \subseteq S\} \tag{7.4.2}$$

Combining (7.4.1) and (7.4.2), we get

$$\begin{aligned}
&\sum_{V\subseteq S}(-1)^{|S\setminus V|}\int m^{(0)}(x)p_{-V}(x_{-V})\mathrm{d}x_{-V}\\
&=\sum_{T\subseteq\{1,\dots,d\}}\sum_{V\subseteq S}(-1)^{|S\setminus V|}\int m_T^{(0)}(x_T)p_{-V}(x_{-V})\mathrm{d}x_{-V}\\
&=\sum_{T\supseteq S}\sum_{V\subseteq S}(-1)^{|S|-|V|}\int m_T^{(0)}(x_T)p_{T\setminus V}(x_{-T\setminus V})\mathrm{d}x_{T\setminus V}\\
&=\sum_{T\supseteq S}\sum_{T\setminus S\subseteq U\subseteq T}(-1)^{|S|-|T\setminus U|}\int m_T^{(0)}(x_T)p_U(x_U)\mathrm{d}x_U.
\end{aligned}$$

It is left to show (7.4.1) and (7.4.2). Equation (7.4.2) follows from straight forward calculations. To see 7.4.1, note

$$\begin{aligned}
&\sum_{V\subseteq S}(-1)^{|S\setminus V|}\int m_T^{(0)}(x_T)p_{-V}(x_{-V})\mathrm{d}x_{-V}\\
=&\sum_{U\subseteq S\cap T}\sum_{W\subseteq S\setminus T}(-1)^{|S\setminus\{W\cup U\}|}\int m_T^{(0)}(x_T)p_{-\{U\cup W\}}(x_{-\{U\cup W\}})\mathrm{d}x_{-\{U\cup W\}}\\
=&\sum_{U\subseteq S\cap T}\int m_T^{(0)}(x_T)p_{-U}(x_{-U})\mathrm{d}x_{-U}\sum_{W\subseteq S\setminus T}(-1)^{|S\setminus\{W\cup U\}|}\\
=&\sum_{U\subseteq S\cap T}\int m_T^{(0)}(x_T)p_{-U}(x_{-U})\mathrm{d}x_{-U}\\
&\times\left(\sum_{W\subseteq S\setminus T,|W|=\text{odd}}(-1)^{|S\setminus U|-1}+\sum_{W\subseteq S\setminus T,|W|=\text{even}}(-1)^{|S\setminus U|}\right)\\
=&0,
\end{aligned}$$

where the last equality follows from the fact that every non-empty set has an equal number of odd and even subsets. □

Lemma 7.4.2 ([111])**.** *For every $U \subseteq \{1,\dots,d\}$,*

$$\int m(x)p_{-U}(x_{-U})\mathrm{d}x_{-U} = \sum_{T\subseteq U} m_T^*(x_T)$$

Proof.

$$\begin{aligned}\sum_{T\subseteq U} m_T(x_T) &= \sum_{T\subseteq U}\sum_{V\subseteq T}(-1)^{|T\setminus V|}\int m^{(0)}(x)p_{-V}(x_{-V})\mathrm{d}x_{-V}\\ &= \sum_{V\subset U}\int m^{(0)}(x)p_{-V}(x_{-V})\mathrm{d}x_{-V}\sum_{S\subseteq\{1,\dots,|U\setminus V|\}}(-1)^{|S|}\\ &\quad+\int m^{(0)}(x)p_{-U}(x_{-U})\\ &= \int m^{(0)}(x)p_{-U}(x_{-U}),\end{aligned}$$

where the last equation follows from $\sum_{S\subseteq\{1,\dots,|U-V|\}}(-1)^{|S|}{=}0$, noting that a non-empty set has an equal number of subsets with an odd number of elements as subsets with an even number of elements. □

7.4.2 Proofs

Proof of Corrolary 5.3.4. This proof is analogue to the proof of Lemma 3.1 in [111]. From 7.4.2, We have $\int m(x)p_{-U}(x_{-U})\mathrm{d}x_{-U} = \sum_{T\subseteq U} m^*_T(x_T)$. Hence the game m with value function

$$v_m(U) = \int m(x)p_{-U}(x_{-U})\mathrm{d}x_{-U}$$

equals the game m^* with value function

$$v_{m^*}(U) = \sum_{S\subseteq\{1,\dots,d\}} m^*_S(x_S)\delta_S(U), \quad \delta_S(U) = 1(S\subseteq U).$$

We now concentrate on the function m^*_S with value function $m^*_S(x_S)\delta_S(U)$. We show that for every non-empty $S\subseteq\{1,\dots,d\}$,

$$\phi_k(x, m^*_S(x_S)\delta_S(U)) = 1(k\in S)\,|S|^{-1}\,m^*_S(x_S). \tag{7.4.3}$$

Here, $\phi_k(x,v)$ denotes the Shapley value for feature k at point x in a game with value function v. The proof is then completed by the additivity axiom, together with

$$\phi_k(x, m^*_\emptyset\delta_\emptyset(U)) = \begin{cases} m^*_\emptyset, & k=0,\\ 0 & \text{else.}\end{cases}$$

The last statement follows from the dummy axiom. To show (7.4.3), let's assume that $j,k\in S$. Then for every $U\subseteq\{1,\dots,d\}$, $m^*_S(x_S)\delta_S(U\cup j) = m^*_S(x_S)\delta_S(U\cup k)$, which, by the symmetry axiom, implies

$$\phi_j(x, m^*_S(x_S)\delta_S(U)) = \phi_k(x, m^*_S(x_S)\delta_S(U)).$$

Additionally, for $k \notin S$, we have $\phi_j(x, m^*_S(x_S)\delta_S(U)) = 0$, by the dummy axiom. Hence we conclude (7.4.3) by applying (5.3.3). □

Proof of Theorem 2.2. Lemma 7.4.2 implies that m^* is a solution. To see this, for $S \subseteq \{1, \dots, d\}$, consider the following decomposition

$$\int m^*(x)p_S(x_S)\mathrm{d}x_S = \sum_{T\cap S\neq\emptyset} \int m^*_T(x_T)p_S(x_S)\mathrm{d}x_S + \sum_{T\cap S=\emptyset} m^*_T(x_T).$$

Using Lemma 7.4.2, we have

$$\int m^*(x)p_S(x_S)\mathrm{d}x_S = \sum_{T\subseteq S^c} m^*_T(x_T) = \sum_{T\cap S=\emptyset} m^*_T(x_T),$$

which with the previous statement implies that m^* is a solution:

$$\sum_{T\cap S\neq\emptyset} \int m^*_T(x_T)p_S(x_S)\mathrm{d}x_S = 0.$$

It is left to show that the solution is unique. Note that for every $S \subseteq \{1, \dots, d\}$

$$\sum_T \int m_T(x_T)p_S(x_S)\mathrm{d}x_S = \sum_{T\cap S=\emptyset} m_T(x_T) + \sum_{T\cap S\neq\emptyset} \int m_T(x_T)p_S(x_S)\mathrm{d}x_S.$$

Hence, condition (5.3.1) is equivalent to

$$\sum_T \int m_T(x_T)p_S(x_S)\mathrm{d}x_S = \sum_{T\cap S=\emptyset} m_T(x_T). \tag{7.4.4}$$

Now assume that there are two set of functions m° and m^* that satisfy (5.3.1) with $\sum_S m^\circ_S = \sum_S m^*_S$. From (7.4.4), it follows that for all $S \subseteq \{1, \dots, d\}$

$$\sum_{T\cap S=\emptyset} m^\circ_T(x_T) = \sum_{T\cap S=\emptyset} m^*_T(x_T),$$

implying $m^\circ_T(x_T) = m^*_T(x_T)$ for all $T \subseteq \{1, \dots, d\}$. □

8 Simulation Results

8.1 Random Planted Forest: a directly interpretable tree ensemble

In this section, we provide further results from our simulations in Chapter 4.

8.1.1 Simulations Random Planted Forest

First, we provide the results which were omitted in Section 4.4. The results are given in the following Tables and further confirm the discussion of that section.

Table 8.1: For each method, we ran 40 simulations to find the optimal parameter combinations from the parameter range below, measured via sample mean squared error: $n^{-1}\sum_i(m(X_i) - \widehat{m}(X_i))^2$. The names of the parameters are drawn from their functional definitions in their respective R-packages.

Method	Parameter range
xgboost	`max.depth` $= 1$ (if additive),
	$= 2,3,4$ (if interaction(2+)),
	`eta` $= 0.005, 0.01, 0.02, 0.04, 0.08, 0.16, 0.32,$
	`nrounds` $= 100, 300, 600, 1000, 3000, 5000, 7000.$
ebm	`max_tree_splits` $= 1,$
	`learning_rate` $= 0.005, 0.01, 0.02, 0.04, 0.08, 0.16, 0.32,$
	`nrounds` $= 100, 300, 600, 1000, 3000, 5000, 7000.$
rpf	`max_interaction` $= 1$ (if additive),
	$= 2$ (if interaction(2)),
	$= \infty$ if (interaction(∞)),
	`t_try` $= 0.25, 0.5, 0.75,$
	`nsplits` $= 10, 15, 20, 25, 30, 40, 50, 60, 80, 100, 120, 200$,
	`split_try` $= 2, 5, 10, 20.$
rf	`m_try` $= \lfloor d/4 \rfloor, \lfloor d/2 \rfloor, \lfloor 3d/4 \rfloor, \rfloor 7d/8 \lfloor, \rfloor d \lfloor,$
	`min.node.size` $= 5,$
	`ntrees` $= 500,$
	`replace` = TRUE,FALSE,
sbf	`bandwidth` $= 0.1, 0.2, 0.3.$
gam	`select` =TRUE, `method` = 'REM' (in sparse models),
	default settings (in dense models).
BART	`power`(β) $= 1, 2, 3,$
	`ntree` = 50,100,150,200,250,300,
	sparsity parameter = 0.6,0.75,0.9.
MARS	`degree` $= 1, 2, 3, 4, 5, 6, 7, 8, 9, 10,$
	`penalty` $= 1, 2, 3, 4, 5, 6, 7, 8, 9, 10.$

Table 8.2: Model 4: Additive Sparse Jump Model. We report the average MSE from 100 simulations. The standard deviations are provided in brackets.

Method	dim=4	dim=10	dim=30
xgboost (`depth=1`)	0.19 (0.029)	0.282 (0.044)	0.401 (0.045)
xgboost	0.198 (0.031)	0.265 (0.053)	0.286 (0.034)
xgboost-CV	0.209 (0.028)	0.281 (0.052)	0.313 (0.058)
rpf (`max_interaction=1`)	0.159 (0.033)	0.198 (0.075)	0.179 (0.041)
rpf (`max_interaction=2`)	0.185 (0.028)	0.24 (0.066)	0.259 (0.043)
rpf	0.192 (0.026)	0.251 (0.065)	0.282 (0.043)
rpf-CV	0.169 (0.033)	0.207 (0.072)	0.183 (0.042)
rf	0.274 (0.035)	0.322 (0.05)	0.375 (0.037)
sbf	0.342 (0.049)	0.603 (0.053)	1.112 (0.138)
gam	0.41 (0.047)	0.406 (0.027)	0.431 (0.06)
BART	0.177 (0.047)	0.162 (0.038)	0.157 (0.034)
BART-CV	0.179 (0.051)	0.163 (0.041)	0.159 (0.036)
MARS	0.751 (0.136)	0.74 (0.104)	0.687 (0.123)
1-NN	2.393 (0.229)	3.029 (0.308)	3.512 (0.333)
average	1.276 (0.075)	1.25 (0.063)	1.213 (0.054)

Table 8.3: Model 7: Additive Dense Smooth Model. We report the average MSE from 100 simulations. The standard deviations are provided in brackets.

Method	dim=4	dim=10
xgboost (`depth=1`)	0.2 (0.035)	0.662 (0.059)
xgboost	0.273 (0.028)	1.233 (0.127)
xgboost-CV	0.209 (0.043)	0.673 (0.06)
rpf (`max_interaction=1`)	0.162 (0.025)	0.578 (0.068)
rpf (`max_interaction=2`)	0.191 (0.017)	0.798 (0.097)
rpf	0.222 (0.019)	1.052 (0.115)
rpf-CV	0.178 (0.03)	0.6 (0.072)
rf	0.567 (0.044)	10.527 (0.772)
sbf	0.071 (0.021)	0.183 (0.026)
gam	0.055 (0.012)	0.171 (0.045)
BART	0.155 (0.023)	0.438 (0.053)
BART-CV	0.165 (0.032)	0.465 (0.094)
MARS	0.166 (0.035)	4.4 (0.36)
1-NN	2.05 (0.108)	11.634 (0.702)
average	7.71 (0.381)	18.986 (1.391)

Table 8.4: Model 8: Hierarchical-interaction Dense Smooth Model. We report the average MSE from 100 simulations. The standard deviations are provided in brackets.

Method	dim=4	dim=10
xgboost	0.645 (0.053)	2.895 (0.271)
xgboost-CV	0.678 (0.042)	3.013 (0.338)
rpf (`max_interaction=2`)	0.414 (0.047)	3.643 (0.349)
rpf	0.385 (0.034)	3.357 (0.372)
rpf-CV	0.413 (0.033)	3.665 (0.467)
rf	0.77 (0.034)	12.265 (1.447)
BART	0.34 (0.04)	1.889 (0.324)
BART-CV	0.354 (0.059)	2.133 (0.363)
MARS	0.624 (0.114)	10.885 (0.635)
1-NN	2.516 (0.141)	17.728 (1.215)
average	10.696 (0.621)	26.502 (1.892)

Table 8.5: Model 2: Hierarchical-interaction Sparse Smooth Model. We report the average MSE from 100 simulations. The standard deviations are provided in brackets.

Method	dim=4	dim=10	dim=30
xgboost (`depth=2`)	2.435 (0.157)	2.542 (0.15)	2.587 (0.152)
xgboost	0.374 (0.035)	0.481 (0.064)	0.557 (0.089)
xgboost-CV	0.393 (0.051)	0.499 (0.058)	0.563 (0.089)
rpf (`max_interaction=1`)	2.36 (0.165)	2.43 (0.17)	2.404 (0.145)
rpf (`max_interaction=2`)	0.248 (0.038)	0.327 (0.045)	0.408 (0.07)
rpf	0.263 (0.034)	0.357 (0.044)	0.452 (0.076)
rpf-CV	0.277 (0.039)	0.366 (0.051)	0.463 (0.083)
rf	0.432 (0.039)	0.575 (0.061)	0.671 (0.08)
sbf	2.298 (0.168)	2.507 (0.181)	3.163 (0.207)
gam	2.242 (0.172)	2.311 (0.159)	2.277 (0.185)
BART	0.214 (0.03)	0.223 (0.04)	0.252 (0.037)
BART-CV	0.242 (0.043)	0.276 (0.053)	0.315 (0.047)
MARS	0.355 (0.089)	0.282 (0.038)	0.414 (0.126)
1-NN	2.068 (0.156)	5.988 (0.624)	11.059 (0.676)
average	8.366 (0.43)	8.086 (0.246)	8.207 (0.496)

Table 8.6: Model 3: Pure-interaction Sparse Smooth Model. We report the average MSE from 100 simulations. The standard deviations are provided in brackets.

Method	dim=4	dim=10	dim=30
xgboost (`depth1`)	2.176 (0.14)	2.236 (0.176)	2.183 (0.136)
xgboost	0.417 (0.082)	0.797 (0.16)	1.381 (0.234)
xgboost-CV	0.443 (0.078)	0.872 (0.136)	1.497 (0.326)
rpf (`max_interaction=1`)	2.172 (0.133)	2.236 (0.164)	2.199 (0.145)
rpf(`max_interaction=2`)	0.416 (0.082)	1.289 (0.224)	1.822 (0.208)
rpf	0.219 (0.035)	0.556 (0.143)	1.186 (0.236)
rpf-CV	0.233 (0.033)	0.603 (0.163)	1.313 (0.253)
rf	0.304 (0.047)	0.744 (0.305)	1.295 (0.317)
sbf	2.249 (0.159)	2.473 (0.181)	3.133 (0.22)
gam	2.161 (0.13)	2.222 (0.172)	2.209 (0.168)
BART	0.168 (0.022)	0.172 (0.032)	0.202 (0.021)
BART-CV	0.192 (0.03)	0.199 (0.039)	0.223 (0.025)
MARS	0.245 (0.088)	0.831 (0.728)	0.429 (0.403)
1-NN	1.323 (0.117)	2.642 (0.317)	4.173 (0.413)
average	2.187 (0.125)	2.226 (0.174)	2.177 (0.146)

Table 8.7: Model 5: Hierarchical-interaction Sparse Jump Model. We report the average MSE from 100 simulations. The standard deviations are provided in brackets.

Method	dim=4	dim=10	dim=30
xgboost(`depth=1`)	2.974 (0.112)	3.046 (0.12)	3.098 (0.223)
xgboost	1.02 (0.152)	1.28 (0.16)	1.418 (0.156)
xgboost-CV	1.049 (0.125)	1.279 (0.157)	1.475 (0.185)
rpf (`max_interaction=1`)	2.941 (0.117)	2.942 (0.123)	2.913 (0.197)
rpf (`max_interaction=2`)	0.767 (0.096)	1.082 (0.139)	1.34 (0.132)
rpf	0.745 (0.089)	1.093 (0.142)	1.307 (0.113)
rpf-CV	0.769 (0.101)	1.167 (0.152)	1.404 (0.14)
rf	0.914 (0.091)	1.237 (0.121)	1.415 (0.152)
sbf	2.791 (0.098)	2.926 (0.12)	3.756 (0.284)
gam	2.782 (0.085)	2.728 (0.105)	2.793 (0.208)
BART	0.611 (0.078)	0.644 (0.106)	0.67 (0.094)
BART-CV	0.661 (0.111)	0.772 (0.173)	0.791 (0.133)
MARS	2.306 (0.17)	2.325 (0.145)	3.374 (2.716)
1-NN	4.559 (0.409)	8.883 (0.692)	13.434 (0.674)
average	8.721 (0.334)	8.449 (0.229)	8.638 (0.412)

Table 8.8: Model 6: Pure-interaction Sparse Jump Model. We report the average MSE from 100 simulations. The standard deviations are provided in brackets.

Method	dim=4	dim=10	dim=30	
xgboost (`depth=1`)	2.662 (0.078)	2.616 (0.105)	2.565 (0.153)	
xgboost	-	1.034 (0.177)	1.723 (0.178)	2.337 (0.378)
xgboost-CV	-	1.196 (0.371)	2.056 (0.3)	2.481 (0.385)
rpf (`max_interaction=1`)	2.682 (0.076)	2.653 (0.103)	2.601 (0.155)	
rpf (`max_interaction=2`)	1.252 (0.164)	2.268 (0.13)	2.534 (0.175)	
rpf	0.834 (0.121)	1.729 (0.156)	2.337 (0.284)	
rpf-CV	0.886 (0.142)	1.939 (0.2)	2.438 (0.246)	
rf	0.805 (0.172)	1.696 (0.168)	2.276 (0.306)	
sbf	2.757 (0.094)	2.893 (0.128)	3.705 (0.282)	
gam	2.645 (0.096)	2.617 (0.095)	2.674 (0.165)	
BART	0.583 (0.074)	0.632 (0.124)	0.798 (0.29)	
BART-CV	0.608 (0.106)	0.73 (0.184)	1.16 (0.655)	
MARS	2.324 (0.14)	2.549 (0.296)	2.522 (0.291)	
1-NN	3.769 (0.323)	5.459 (0.419)	6.247 (0.434)	
average	2.637 (0.092)	2.59 (0.106)	2.55 (0.14)	

Table 8.9: Model 8: Hierarchical-interaction Dense Smooth Model. We report the average MSE from 100 simulations. The standard deviations are provided in brackets.

Method	dim=4	dim=10
xgboost (`depth=1`)	3.509 (0.266)	10.108 (0.425)
xgboost	0.645 (0.053)	2.895 (0.271)
xgboost-CV	0.678 (0.042)	3.013 (0.338)
rpf (`max_interaction=1`)	3.408 (0.237)	9.717 (0.378)
rpf (`max_interaction=2`)	0.414 (0.047)	3.643 (0.349)
rpf	0.385 (0.034)	3.357 (0.372)
rpf-CV	0.413 (0.033)	3.665 (0.467)
rf	0.77 (0.034)	12.265 (1.447)
sbf	3.42 (0.208)	9.215 (0.419)
gam	3.258 (0.227)	9.212 (0.483)
BART	0.34 (0.04)	1.889 (0.324)
BART-CV	0.354 (0.059)	2.133 (0.363)
MARS	0.624 (0.114)	10.885 (0.635)
1-NN	2.516 (0.141)	17.728 (1.215)
average	10.696 (0.621)	26.502 (1.892)

Table 8.10: Model 9: Pure-interaction Dense Smooth Model. We report the average MSE from 100 simulations. The standard deviations are provided in brackets.

Method	dim=4	dim=10
xgboost (`depth=1`)	3.108 (0.197)	8.091 (0.359)
xgboost	0.596 (0.063)	3.888 (0.411)
xgboost-CV	0.684 (0.069)	3.974 (0.508)
rpf (`max_interaction=1`)	3.119 (0.209)	8.156 (0.366)
rpf (`max_interaction=2`)	0.712 (0.101)	5.944 (0.324)
rpf	0.38 (0.049)	4.747 (0.329)
rpf-CV	0.395 (0.055)	4.789 (0.335)
rf	0.657 (0.074)	5.784 (0.409)
sbf	3.385 (0.183)	9.177 (0.479)
gam	3.109 (0.216)	8.183 (0.389)
BART	0.266 (0.034)	1.425 (0.183)
BART-CV	0.299 (0.054)	1.738 (0.254)
MARS	0.618 (0.552)	6.257 (0.824)
1-NN	1.482 (0.126)	7.358 (0.514)
average	3.156 (0.221)	8.109 (0.363)

Table 8.11: Model 10: Additive Dense Jump Model.We report the average MSE from 100 simulations. The standard deviations are provided in brackets.

Method	dim=4	dim=10
xgboost (`depth=1`)	0.325 (0.068)	1.095 (0.106)
xgboost	0.376 (0.085)	1.437 (0.153)
xgboost-CV	0.36 (0.073)	1.187 (0.145)
rpf (`max_interaction=1`)	0.321 (0.047)	1.273 (0.161)
rpf (`max_interaction=2`)	0.402 (0.059)	2.18 (0.139)
rpf	0.429 (0.067)	2.804 (0.192)
rpf-CV	0.326 (0.051)	1.303 (0.166)
rf	0.807 (0.104)	4.051 (0.186)
sbf	0.588 (0.078)	1.685 (0.185)
gam	0.923 (0.15)	4.405 (0.379)
BART	0.369 (0.079)	1.164 (0.093)
BART-CV	0.39 (0.076)	1.436 (0.253)
MARS	1.749 (0.151)	5.424 (0.304)
1-NN	3.822 (0.296)	11.278 (1.097)
average	2.5 (0.116)	6.332 (0.41)

Table 8.12: Model 11: Hierarchical-interaction Dense Jump Model. We report the average MSE from 100 simulations. The standard deviations are provided in brackets.

Method	dim=4	dim=10
xgboost(`depth=1`)	4.19 (0.238)	13.112 (0.83)
xgboost	1.666 (0.159)	9.327 (0.594)
xgboost-CV	1.87 (0.312)	9.407 (0.653)
rpf(`max_interaction=1`)	4.19 (0.253)	12.997 (0.831)
rpf (`max_interaction=2`)	1.43 (0.205)	9.238 (0.648)
rpf	1.26 (0.165)	9 (0.64)
rpf-CV	1.303 (0.171)	9.441 (0.604)
rf	1.681 (0.14)	13.6 (1.247)
sbf	3.972 (0.254)	12.234 (0.688)
gam	3.997 (0.251)	12.524 (0.858)
BART	1.01 (0.121)	7.116 (0.653)
BART-CV	1.165 (0.224)	7.897 (0.917)
MARS	3.595 (0.15)	16.307 (1.266)
1-NN	5.839 (0.649)	30.736 (1.933)
average	11.471 (0.498)	29.623 (2.046)

Table 8.13: Model 12: Pure-interaction Dense Jump Model.We report the average MSE from 100 simulations. The standard deviations are provided in brackets.

Method	dim=4	dim=10
xgboost (`depth=1`)	3.787 (0.297)	11.106 (0.546)
xgboost	1.606 (0.164)	10.005 (0.531)
xgboost-CV	1.768 (0.456)	10.868 (0.648)
rpf (`max_interaction=1`)	3.816 (0.259)	11.246 (0.596)
rpf (`max_interaction=2`)	2.005 (0.237)	10.846 (0.488)
rpf	1.441 (0.187)	10.264 (0.433)
rpf-CV	1.564 (0.214)	10.582 (0.536)
rf	1.36 (0.175)	10.235 (0.424)
sbf	3.928 (0.262)	12.129 (0.639)
gam	3.794 (0.271)	11.154 (0.573)
BART	0.972 (0.129)	6.835 (0.579)
BART-CV	1.04 (0.158)	7.208 (0.775)
MARS	3.541 (0.289)	11.03 (0.574)
1-NN	4.819 (0.475)	19.802 (1.51)
average	3.768 (0.283)	11.03 (0.574)

8.1.2 Generalized Random Planted Forest

First, we provide the results which were omitted in Subsection 4.6.6. The results are given in the following Tables and further confirm the discussion of that section.

Table 8.14: We summerize all data sets used for the results shown in Subsection 4.6.6. The parameter n denotes the number of data, d denotes the number of covariates. The data is taken from [10] and includes only binary responses with $n \cdot d \leq 100000$ without missing values.

data set	n	d
blood-transfusion-service-center	748	4
banknote-authentication	1372	4
ilpd	583	10
diabetes	768	8
tic-tac-toe	958	9
climate-model-simulation-crashes	540	18
kc2	522	21
wdbc	569	30
credit-g	1000	20
pc1	1109	21
wilt	4839	5
phoneme	5404	5
qsar-biodeg	1055	41
kc1	2109	21
pc4	1458	37
pc3	1563	37
churn	5000	20

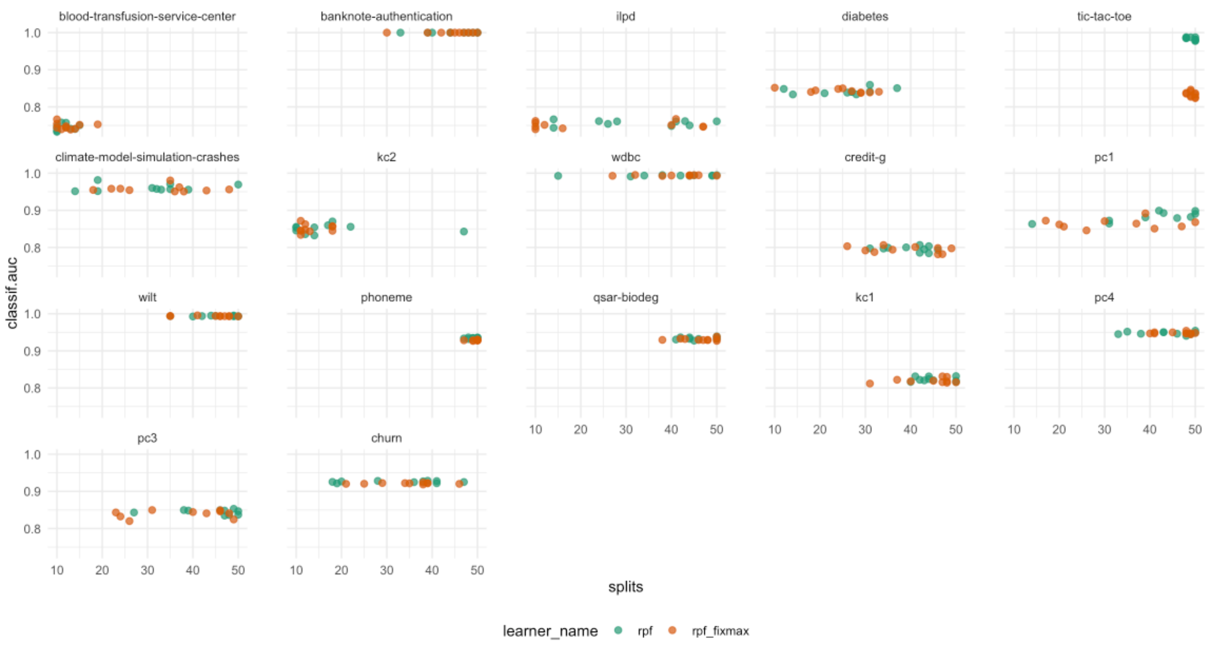


Figure 8.1: Number of splits chosen by the inner crossvalidation for each of the 17 data sets used for binary classification. Each node corresponds to one of the 10 outer crossvalidation estimates.

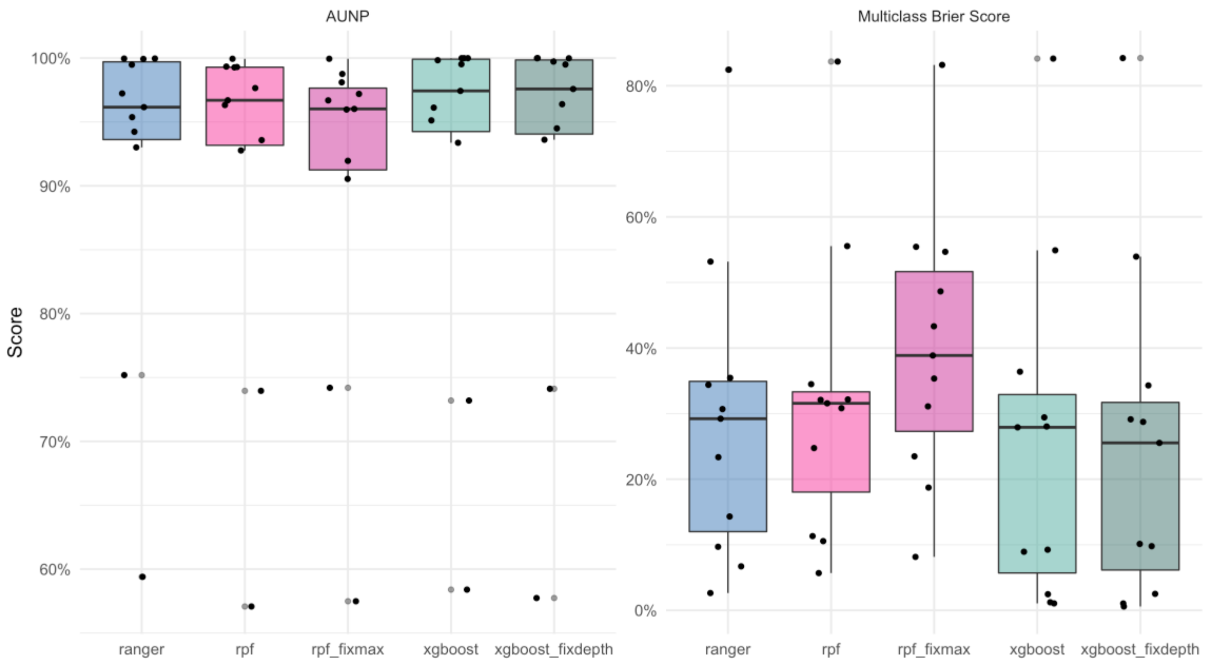


Figure 8.2: Summary of multiclass classification results over all 11 data sets. Each node corresponds to the AUC (left) or Brier Score (right) of one distinct data set. Lower and upper bounds of boxplots correspond to 25% and 75% quantiles respectively. Black lines correspond to the median.

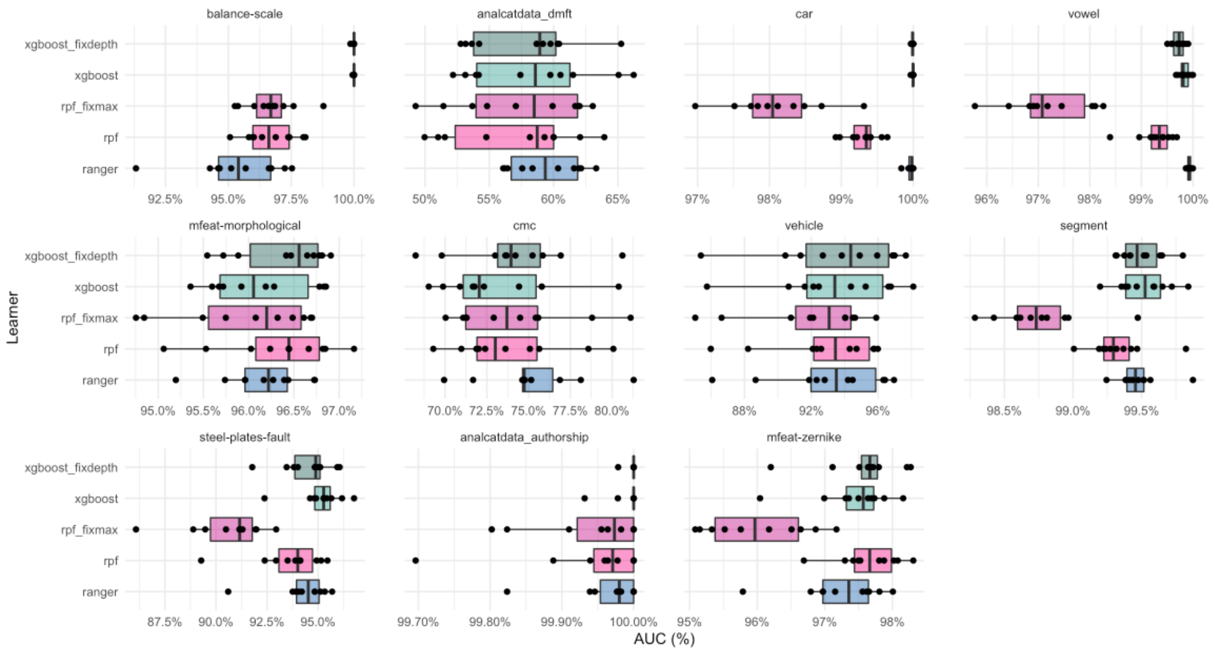


Figure 8.3: AUC results of each of the 11 data sets used for multiclass classification. Each plot contains 10 estimates obtained by splitting the data into test data and training data with a 10 fold crossvalidation. Each node corresponds to one of the 10 outer crossvalidation estimates. Lower and upper bounds of boxplots correspond to 25% and 75% quantiles respectively. Black lines correspond to the median.

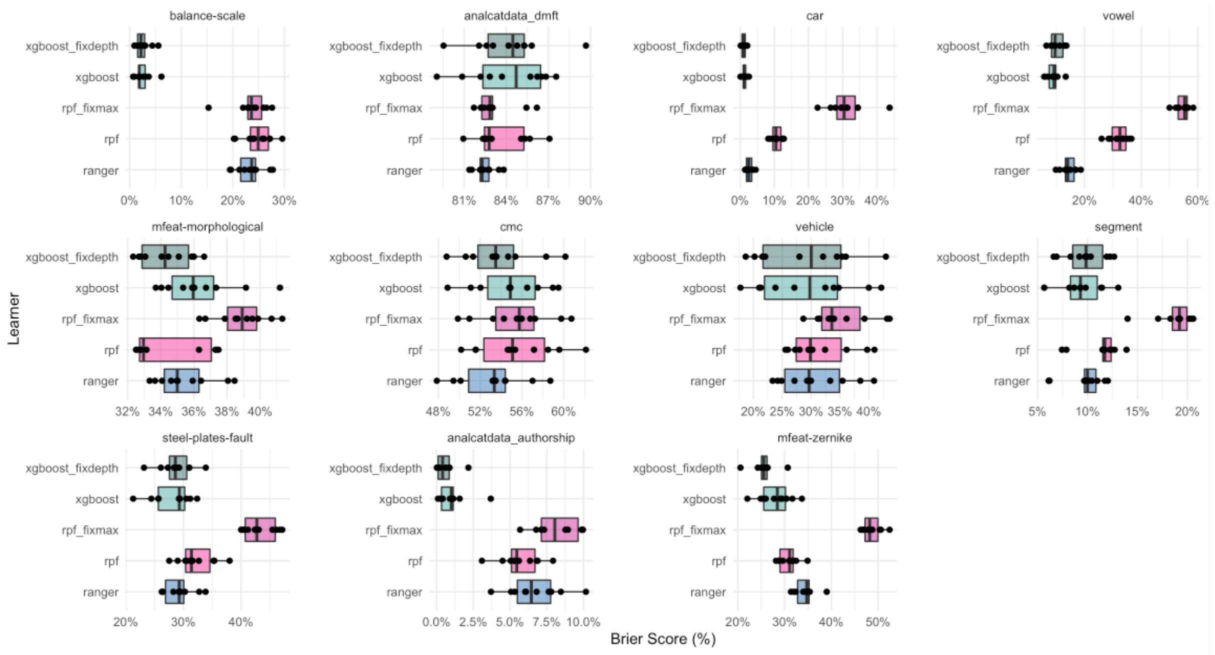


Figure 8.4: Brier scores of each of the 11 data sets used for multiclass classification. Each plot contains 10 estimates obtained by splitting the data into test data and training data with a 10 fold crossvalidation. Each node corresponds to one of the 10 outer crossvalidation estimates. Lower and upper bounds of boxplots correspond to 25% and 75% quantiles respectively. Black lines correspond to the median.

Table 8.15: For each method, we ran 40 simulations to find the optimal parameter combinations from the parameter range below, measured via sample mean squared error: $n^{-1}\sum_i(m(X_i)-\widehat{m}(X_i))^2$. The names of the parameters are drawn from their functional definitions in their respective R-packages.

Method	Parameter range
xgboost	`max.depth` $= 1,\dots,20$,
	`eta` $\in (0,1)$,
	`colsample_bytree` $\in (0.1,1)$,
	`subsample` $\in (0.1,1)$,
	`nrounds` $= 10,\dots,5000$.
xgboost_fixmax	`max.depth` $= 2$,
	`eta` $\in (0,1)$,
	`colsample_bytree` $\in (0.1,1)$,
	`subsample` $\in (0.1,1)$,
	`nrounds` $= 10,\dots,5000$.
rpf	`max_interaction` $= 1,\dots,30$,
	`t_try` $\in (0.1,0.9)$
	`nsplits` $= 10,\dots,30$,
	`ntrees` $= 50$,
	`split_try` $= 1,\dots,20$,
	Loss= L1, exponential.
rpf_fixmax	`max_interaction` $= 2$,
	`t_try` $\in (0.1,0.9)$
	`nsplits` $= 10,\dots,30$,
	`ntrees` $= 50$,
	`split_try` $= 1,\dots,20$,
	Loss= L_1, logit.
rf	`mtry.ratio` $\in (0.1,1)$,
	`min.node.size` $= 1,\dots,50$,
	`num.trees` $= 50$,
	`replace` = TRUE,FALSE,
	`sample.fraction`$\in (0.1,1]$.

Table 8.16: We summerize all data sets used for multiclass classification. The parameter n denotes the number of data, d denotes the number of covariates. The data is taken from [10] and includes only responses which take on more than two values with $n \cdot d \leq 100000$ without missing values.

data set	n	d
balance-scale	625	4
analcatdata_dmft	797	4
car	1728	6
vowel	990	12
mfeat-morphological	2000	6
cmc	1473	9
vehicle	846	18
segment	2310	16
steel-plates-fault	1941	27
analcatdata_authorship	841	70
mfeat-zernike	2000	47

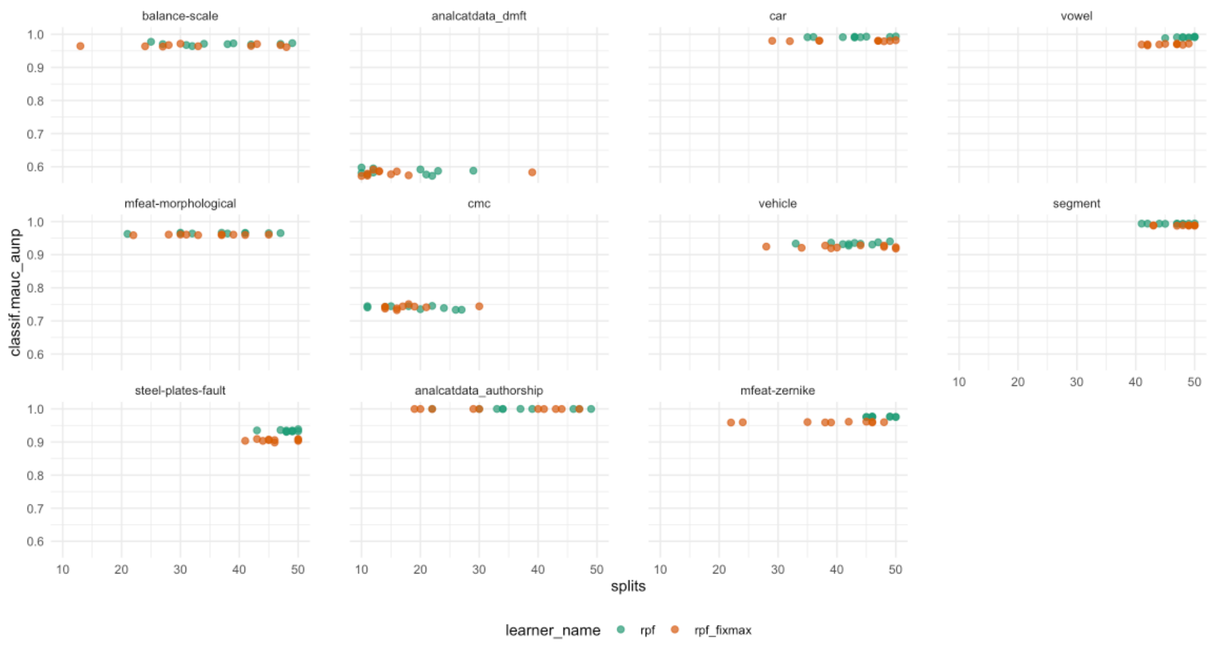


Figure 8.5: Number of splits chosen by the inner crossvalidation for each of the 11 data sets used for multiclass classification. Each node corresponds to one of the 10 outer crossvalidation estimates.

8.2 Unifying Local and Global Model Explanations by Functional Decomposition of Low Dimensional Structures

8.2.1 Motivating Example

This section shows the simulation results when using the *random planted forest algorithm* as an estimation procedure. First of all, the results considering the motivating example from Section 5.1.1 are given in Figure 8.6. The following subsections include the results of experiments which where discussed in Section 5.5.

8.2.2 Truly Global Explanations

Figure 8.7 includes the results discussed in Subsection 5.5.1 when considering the *random planted forest algorithm* instead of *xgboost*.

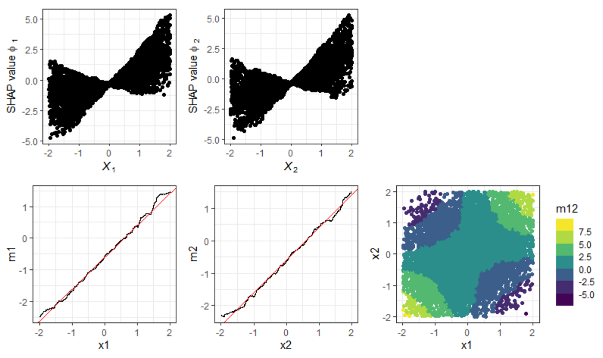


Figure 8.6: Simple example. SHAP values (top row) and functional decomposition (bottom row) of a *random planted forest* model of the function $m(x_1, x_2) = x_1 + x_2 + 2x_1x_2$. The red lines in the bottom row represent the SHAP values of the true function.

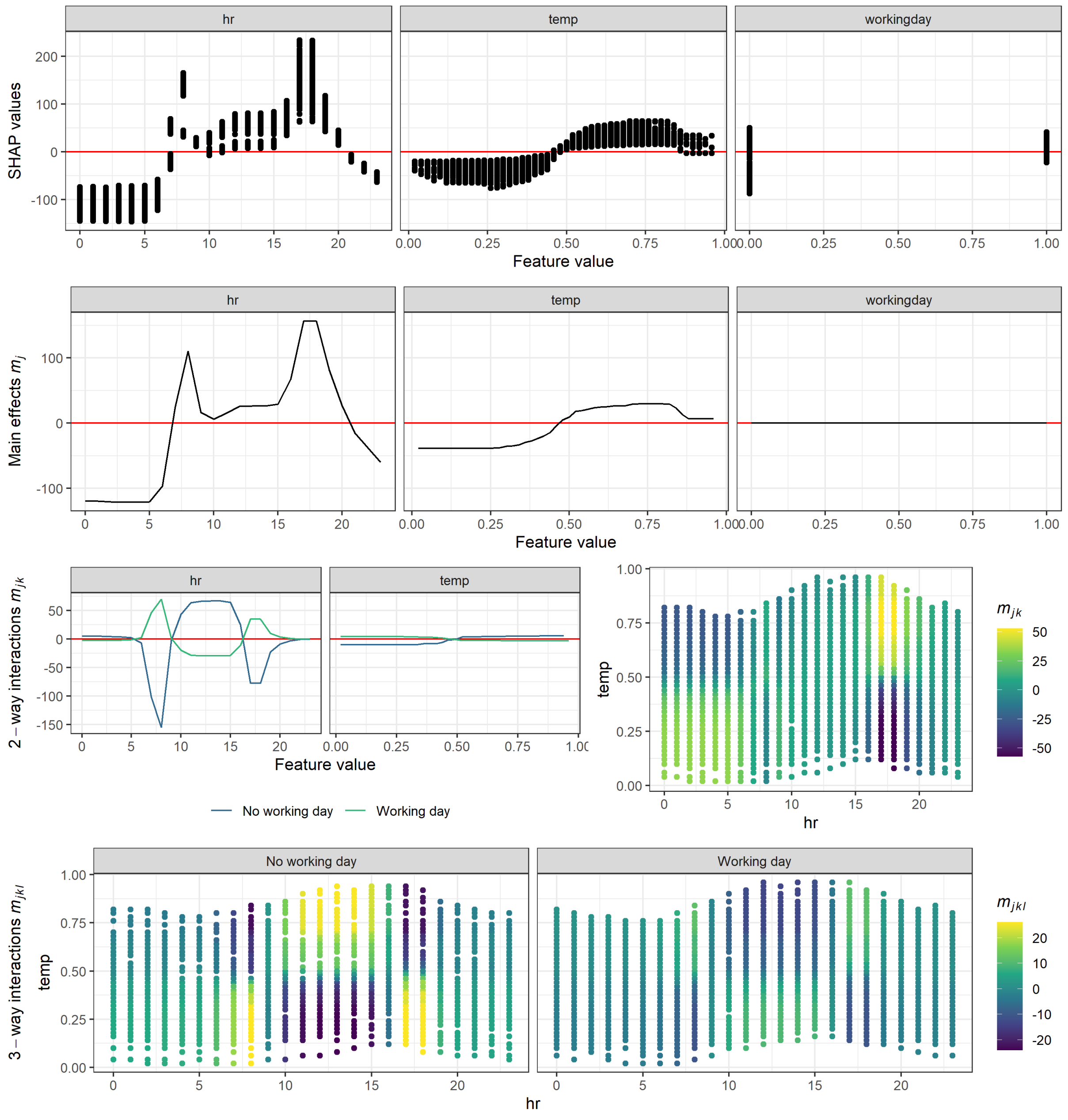


Figure 8.7: Bike sharing example (*random planted forest*). SHAP values (top row), main effects (second row), 2-way interactions (third row) and 3-way interactions (bottom row) of the features *hour of the day* (hr, 0-24 full hours), *Temperature* (temp, normalized to 0-1) and *working day* (workingday, 0=no, 1=yes) of the bike sharing data.

8.2.3 Feature Importance

Figure 8.8 includes the results discussed in Subsection 5.5.2 when considering the *random planted forest algorithm* instead of *xgboost*.

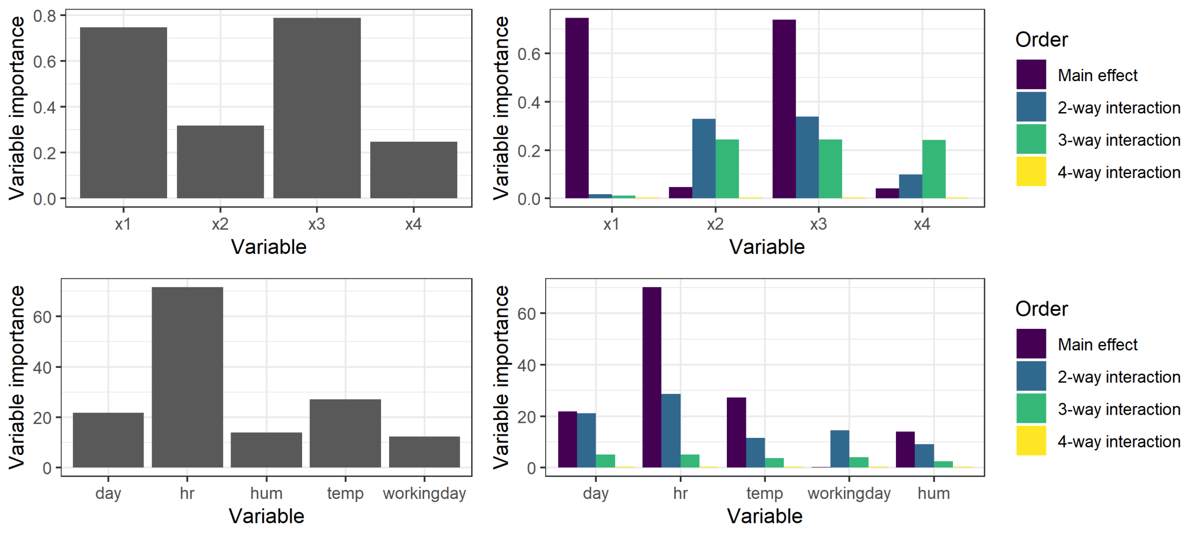
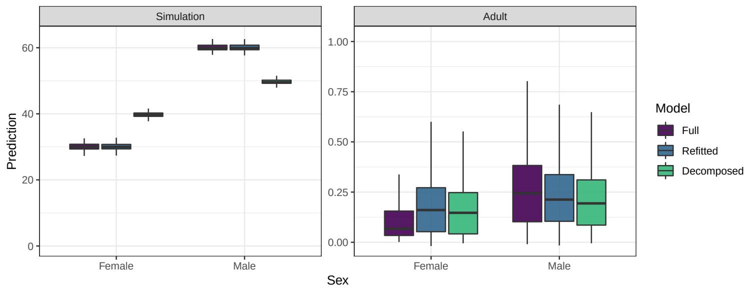


Figure 8.8: Feature importance (*random planted forest*) for the function $m(x) = x_1 + x_3 + x_2 x_3 - 2x_2 x_3 x_4$ (top row) and the bike sharing data from Section 5.5.1 (bottom row) based on SHAP values (left column) and our functional decomposition separately for main effects and interactions of different orders (right column).

8.2.4 Post-hoc Feature Removal

Figure 8.9 includes the results discussed in Section 5.5.3 when considering the *random planted forest algorithm* instead of *xgboost*.



Setting	Median difference		
	Full	Refitted	Decomposed
Simulation	29.90	29.91	9.84
Adult	0.18	0.052	0.047

Figure 8.9: Post-hoc feature removal (*random planted forest*). Predictions in a simulation (left) and the *adult* dataset for males and females of the full model, a refitted model without the protected feature *sex* and a decomposed model where the feature *sex* was removed post-hoc. The table below shows the median differences between females and males for the three models.

Bibliography

[1] Rishabh Agarwal, Nicholas Frosst, Xuezhou Zhang, Rich Caruana, and Geoffrey E Hinton. Neural additive models: Interpretable machine learning with neural nets. *arXiv preprint*, arXiv:2004.13912:1–17, 2020.

[2] Daniel W Apley and Jingyu Zhu. Visualizing the effects of predictor variables in black box supervised learning models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 82(4):1059–1086, 2020.

[3] Jean-Yves Audibert and Alexandre B Tsybakov. Fast learning rates for plug-in classifiers. *The Annals of Statistics*, 35(2):608–633, 2007.

[4] Solon Barocas, Moritz Hardt, and Arvind Narayanan. *Fairness and Machine Learning.* 2019. `http://www.fairmlbook.org`.

[5] Andrew R Barron. Approximation and estimation bounds for artificial neural networks. *Machine learning*, 14(1):115–133, 1994.

[6] Gérard Biau. Analysis of a random forests model. *The Journal of Machine Learning Research*, 13(1):1063–1095, 2012.

[7] Gérard Biau, Luc Devroye, and Gábor Lugosi. Consistency of random forests and other averaging classifiers. *Journal of Machine Learning Research*, 9(Sep):2015–2033, 2008.

[8] Gérard Biau and Erwan Scornet. A random forest guided tour. *Test*, 25(2):197–227, 2016.

[9] Peter J Bickel, Chris AJ Klaassen, Ya'acov Ritov, and Jon A Wellner. *Efficient and adaptive estimation for semiparametric models.* John Hopkins University Press, Baltimore, 1993.

[10] Bernd Bischl, Giuseppe Casalicchio, Matthias Feurer, Frank Hutter, Michel Lang, Rafael G Mantovani, Jan N van Rijn, and Joaquin Vanschoren. Openml benchmarking suites. *arXiv preprint arXiv:1708.03731*, 2017.

[11] Sebastian Bordt and Ulrike von Luxburg. From shapley values to generalized additive models and back. *arXiv preprint arXiv:2209.04012*, 2022.

[12] Thijs Bos and Johannes Schmidt-Hieber. Convergence rates of deep relu networks for multiclass classification. *Electronic Journal of Statistics*, 16(1):2724–2773, 2022.

[13] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[14] Leo Breiman and Jerome H Friedman. Estimating optimal transformations for multiple regression and correlation. *Journal of the American Statistical Association*, 80(391):580–598, 1985.

[15] Andreas Buja, Trevor Hastie, and Robert Tibshirani. Linear smoothers and additive models. *The Annals of Statistics*, 17:453–510, 1989.

[16] Rich Caruana, Yin Lou, Johannes Gehrke, Paul Koch, Marc Sturm, and Noemie Elhadad. Intelligible models for healthcare: Predicting pneumonia risk and hospital 30-day readmission. In *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1721–1730, 2015.

[17] Giuseppe Casalicchio, Christoph Molnar, and Bernd Bischl. Visualizing the feature importance for black box models. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 655–670. Springer, 2018.

[18] Gaëlle Chastaing, Fabrice Gamboa, Clémentine Prieur, et al. Generalized hoeffding-sobol decomposition for dependent variables-application to sensitivity analysis. *Electronic Journal of Statistics*, 6(2012):2420–2448, 2012.

[19] Hugh Chen, Joseph D Janizek, Scott Lundberg, and Su-In Lee. True to the model or true to the data? *arXiv preprint arXiv:2006.16234*, 2020.

[20] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794, 2016.

[21] Tianqi Chen, Tong He, Michael Benesty, Vadim Khotilovich, Yuan Tang, Hyunsu Cho, Kailong Chen, Rory Mitchell, Ignacio Cano, Tianyi Zhou, Mu Li, Junyuan Xie, Min Lin, Yifeng Geng, and Yutian Li. xgboost: Extreme gradient boosting, 2020. R package version 1.2.0.1.

[22] Xi Chen and Hemant Ishwaran. Random forests for genomic data analysis. *Genomics*, 99(6):323–329, 2012.

[23] Victor Chernozhukov, Denis Chetverikov, Mert Demirer, Esther Duflo, Christian Hansen, Whitney Newey, and James Robins. Double/debiased machine learning for treatment and structural parameters. *The Econometrics Journal*, 21(1):C1–C68, 2018.

[24] Chien-Ming Chi, Patrick Vossler, Yingying Fan, and Jinchi Lv. Asymptotic properties of high-dimensional random forests. *arXiv preprint arXiv:2004.13953*, 2020.

[25] Hugh A Chipman, Edward I George, and Robert E McCulloch. Bart: Bayesian additive regression trees. *The Annals of Applied Statistics*, 4(1):266–298, 2010.

[26] Ronan Collobert and Jason Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th International Conference on Machine Learning*, pages 160–167, 2008.

[27] Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. Natural language processing (almost) from scratch. *Journal of machine learning research*, 12(ARTICLE):2493–2537, 2011.

[28] Ian Covert, Scott M Lundberg, and Su-In Lee. Understanding global feature contributions with additive importance measures. *Advances in Neural Information Processing Systems*, 33:17212–17223, 2020.

[29] George Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals and Systems*, 2(4):303–314, 1989.

[30] Yann N Dauphin, Razvan Pascanu, Caglar Gulcehre, Kyunghyun Cho, Surya Ganguli, and Yoshua Bengio. Identifying and attacking the saddle point problem in high-dimensional non-convex optimization. *Advances in neural information processing systems*, 27, 2014.

[31] Dheeru Dua and Casey Graff. UCI machine learning repository, 2017. `http://archive.ics.uci.edu/ml`.

[32] Richard M. Dudley. Metric entropy of some classes of sets with differentiable boundaries. *Journal of Approximation Theory*, 10(3):227–236, 1974.

[33] Hadi Fanaee-T and Joao Gama. Event labeling combining ensemble detectors and background knowledge. *Progress in Artificial Intelligence*, 2(2):113–127, 2014.

[34] Yoav Freund and Robert E Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of computer and system sciences*, 55(1):119–139, 1997.

[35] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. Additive logistic regression: a statistical view of boosting (with discussion and a rejoinder by the authors). *The annals of statistics*, 28(2):337–407, 2000.

[36] Jerome H Friedman. Multivariate adaptive regression splines. *The annals of statistics*, 19(1):1–67, 1991.

[37] Jerome H Friedman. Greedy function approximation: a gradient boosting machine. *Annals of statistics*, 29(5):1189–1232, 2001.

[38] Jerome H Friedman and Werner Stuetzle. Projection pursuit regression. *Journal of the American Statistical Association*, 76:817–823, 1981.

[39] Christopher Frye, Colin Rowat, and Ilya Feige. Asymmetric Shapley values: incorporating causal knowledge into model-agnostic explainability. *Advances in Neural Information Processing Systems*, 33:1229–1239, 2020.

[40] Jianlong Fu, Heliang Zheng, and Tao Mei. Look closer to see better: Recurrent attention convolutional neural network for fine-grained image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4438–4446, 2017.

[41] Pierre Geurts, Damien Ernst, and Louis Wehenkel. Extremely randomized trees. *Machine learning*, 63(1):3–42, 2006.

[42] Michel Grabisch and Marc Roubens. An axiomatic approach to the concept of interaction among players in cooperative games. *International Journal of Game Theory*, 28(4):547–565, 1999.

[43] Karl Gregory, Enno Mammen, and Martin Wahl. Optimal estimation of sparse high-dimensional additive models. *The Annals of Statistics*, forthcoming, 2020.

[44] Leo Grinsztajn, Edouard Oyallon, and Gael Varoquaux. Why do tree-based models still outperform deep learning on typical tabular data? In *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2022.

[45] Kyunghee Han, Hans-Georg Müller, and Byeong U Park. Additive functional regression for densities as responses. *Journal of the American Statistical Association*, 115:997–1010, 2020.

[46] Kyunghee Han, Byeong U Park, et al. Smooth backfitting for errors-in-variables additive models. *The Annals of Statistics*, 46:2216–2250, 2018.

[47] Alexander Hapfelmeier, Torsten Hothorn, Kurt Ulm, and Carolin Strobl. A new variable importance measure for random forests with missing data. *Statistics and Computing*, 24(1):21–34, 2014.

[48] Wolfgang Härdle, Stefan Sperlich, and Volodia Spokoiny. Structural tests in additive regression. *Journal of the American Statistical Association*, 96:1333–1347, 2001.

[49] Chris Harris, Richard Pymar, and Colin Rowat. Joint Shapley values: a measure of joint feature importance. In *International Conference on Learning Representations*, 2022.

[50] Maintainer Trevor Hastie. Package 'mda', 2020.

[51] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[52] Andrew Herren and P Richard Hahn. Statistical aspects of shap: Functional anova for model interpretation. *arXiv preprint arXiv:2208.09970*, 2022.

[53] Munir Hiabu, Enno Mammen, M Dolores Martínez-Miranda, and Jens P Nielsen. Smooth backfitting of proportional hazards with multiplicative components. *Journal of the American Statistical Association*, forthcoming, 2020.

[54] Munir Hiabu, Enno Mammen, and Joseph T Meyer. Random planted forest: a directly interpretable tree ensemble. *arXiv preprint arXiv:2012.14563*, 2020.

[55] Munir Hiabu, Enno Mammen, and Joseph T Meyer. Local linear smoothing in additive models as data projection. *arXiv preprint arXiv:2201.10930*, 2022.

[56] Munir Hiabu, Joseph T Meyer, and Marvin N Wright. Unifying local and global model explanations by functional decomposition of low dimensional structures. *arXiv preprint arXiv:2208.06151*, 2022.

[57] Geoffrey Hinton, Li Deng, Dong Yu, George E Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara N Sainath, et al. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal processing magazine*, 29(6):82–97, 2012.

[58] Arthur E Hoerl and Robert W Kennard. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(1):55–67, 1970.

[59] Benjamin Hofner, Andreas Mayr, Nikolay Robinzonov, and Matthias Schmid. Model-based boosting in R: A hands-on tutorial using the R package mboost. *Computational statistics*, 29(1):3–35, 2014.

[60] Giles Hooker. Generalized functional anova diagnostics for high-dimensional functions of dependent variables. *Journal of Computational and Graphical Statistics*, 16(3):709–732, 2007.

[61] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5):359–366, 1989.

[62] Nicholas R Howe, Toni M Rath, and R Manmatha. Boosted decision trees for word recognition in handwritten document retrieval. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 377–383, 2005.

[63] Tianyang Hu, Jun Wang, Wenjia Wang, and Zhenguo Li. Understanding square loss in training overparametrized neural network classifiers. *arXiv preprint arXiv:2112.03657*, 2021.

[64] Masaaki Imaizumi and Kenji Fukumizu. Deep neural networks learn non-smooth functions effectively. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 869–878. PMLR, 2019.

[65] Jan Ittner, Lukasz Bolikowski, Konstantin Hemker, and Ricardo Kennedy. Feature synergy, redundancy, and independence in global model explanations using SHAP vector decomposition. *arXiv preprint arXiv:2107.12436*, 2021.

[66] Dominik Janzing, Lenon Minorics, and Patrick Blöbaum. Feature relevance quantification in explainable AI: A causal problem. In *International Conference on Artificial Intelligence and Statistics*, pages 2907–2916. PMLR, 2020.

[67] Jeong Min Jeon, Byeong U Park, et al. Additive regression with Hilbertian responses. *The Annals of Statistics*, 48:2671–2697, 2020.

[68] Hui Jiang, Zheng He, Gang Ye, and Huyin Zhang. Network intrusion detection based on pso-xgboost model. *IEEE Access*, 8:58392–58401, 2020.

[69] Tosio Kato. *Perturbation theory for linear operators.* Springer Science & Business Media, 2013.

[70] Javed Khan, Jun S Wei, Markus Ringner, Lao H Saal, Marc Ladanyi, Frank Westermann, Frank Berthold, Manfred Schwab, Cristina R Antonescu, Carsten Peterson, and Paul S Metzler. Classification and diagnostic prediction of cancers using gene expression profiling and artificial neural networks. *Nature Medicine*, 7(6):673–679, 2001.

[71] Yongdai Kim, Ilsang Ohn, and Dongha Kim. Fast convergence rates of deep neural networks for classification. *Neural Networks*, 138:179–197, 2021.

[72] H Kober. A theorem on Banach spaces. *Compositio Mathematica*, 7:135–140, 1940.

[73] Michael Kohler, Adam Krzyżak, and Sophie Langer. Estimation of a function of low local dimensionality by deep neural networks. *IEEE Transactions on Information Theory*, 2022.

[74] Michael Kohler and Sophie Langer. Statistical theory for image classification using deep convolutional neural networks with cross-entropy loss. *arXiv preprint arXiv:2011.13602*, 2020.

[75] Michael Kohler and Sophie Langer. On the rate of convergence of fully connected deep neural network regression estimates. *The Annals of Statistics*, 49(4):2231–2249, 2021.

[76] I. Elizabeth Kumar, Suresh Venkatasubramanian, Carlos Scheidegger, and Sorelle Friedler. Problems with Shapley-value-based explanations as feature importance measures. In *Proceedings of the 37th International Conference on Machine Learning*, pages 5491–5500. PMLR, 2020.

[77] Indra Kumar, Carlos Scheidegger, Suresh Venkatasubramanian, and Sorelle Friedler. Shapley residuals: Quantifying the limits of the Shapley value for explanations. *Advances in Neural Information Processing Systems*, 34:26598–26608, 2021.

[78] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

[79] Christian Leibig, Vaneeda Allken, Murat Seçkin Ayhan, Philipp Berens, and Siegfried Wahl. Leveraging uncertainty information from deep neural networks for disease detection. *Scientific Reports*, 7(1):1–14, 2017.

[80] Benjamin Lengerich, Sarah Tan, Chun-Hao Chang, Giles Hooker, and Rich Caruana. Purifying interaction effects with the functional anova: An efficient algorithm for recovering identifiable additive models. In *International Conference on Artificial Intelligence and Statistics*, pages 2402–2412. PMLR, 2020.

[81] Yin Lou, Rich Caruana, and Johannes Gehrke. Intelligible models for classification and regression. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 150–158, 2012.

[82] Yin Lou, Rich Caruana, Johannes Gehrke, and Giles Hooker. Accurate intelligible models with pairwise interactions. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 623–631, 2013.

[83] Scott M Lundberg, Gabriel Erion, Hugh Chen, Alex DeGrave, Jordan M Prutkin, Bala Nair, Ronit Katz, Jonathan Himmelfarb, Nisha Bansal, and Su-In Lee. From local explanations to global understanding with explainable AI for trees. *Nature Machine Intelligence*, 2(1):56–67, 2020.

[84] Scott M Lundberg and Su-In Lee. A unified approach to interpreting model predictions. *Advances in Neural Information Processing Systems*, 30, 2017.

[85] Jan MacDonald, Stephan Wäldchen, Sascha Hauch, and Gitta Kutyniok. A rate-distortion framework for explaining neural network decisions. *arXiv preprint arXiv:1905.11092*, 2019.

[86] E Mammen, JS Marron, BA Turlach, and MP Wand. A general projection framework for constrained smoothing. *Statistical Science*, 16:232–248, 2001.

[87] Enno Mammen, Oliver Linton, and Jans Perch Nielsen. The existence and asymptotic properties of a backfitting projection algorithm under weak conditions. *The Annals of Statistics*, 27(5):1443–1490, 1999.

[88] Enno Mammen and Jens Perch Nielsen. Generalised structured models. *Biometrika*, 90:551–566, 2003.

[89] Enno Mammen, Byeong U Park, and Melanie Schienle. Additive models: Extensions and related models. In Jeffrey S. Racine, Liangjun Su, and Aman Ullah, editors, *The Oxford Handbook of Applied Nonparametric and Semiparametric Econometrics and Statistics.* Oxford Univ. Press, 2014.

[90] Enno Mammen and Stefan Sperlich. Additivity tests based on smooth backfitting. *Biometrika*, forthcoming, 2021.

[91] Enno Mammen and Alexander B. Tsybakov. Smooth discrimination analysis. *The Annals of Statistics*, 27(6):1808–1829, 1999.

[92] Enno Mammen and Kyusang Yu. Nonparametric estimation of noisy integral equations of the second kind. *Journal of the Korean Statistical Society*, 38:99–110, 2009.

[93] Warren S McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4):115–133, 1943.

[94] Lucas Mentch and Giles Hooker. Quantifying uncertainty in random forests via confidence intervals and hypothesis tests. *The Journal of Machine Learning Research*, 17(1):841–881, 2016.

[95] Lucas Mentch and Giles Hooker. Formal hypothesis tests for additive structure in random forests. *Journal of Computational and Graphical Statistics*, 26(3):589–597, 2017.

[96] Joseph T Meyer. Optimal convergence rates of deep neural networks in a classification setting. *arXiv preprint arXiv:2207.12180*, 2022.

[97] John Ashworth Nelder and Robert WM Wedderburn. Generalized linear models. *Journal of the Royal Statistical Society: Series A (General)*, 135(3):370–384, 1972.

[98] Stefano Nembrini, Inke R König, and Marvin N Wright. The revival of the Gini importance? *Bioinformatics*, 34(21):3711–3718, 05 2018.

[99] Jens Perch Nielsen and Stefan Sperlich. Smooth backfitting in practice. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(1):43–61, 2005.

[100] Jean D Opsomer. Asymptotic properties of backfitting estimators. *Journal of Multivariate Analysis*, 73(2):166–179, 2000.

[101] Guillermo Owen. Multilinear extensions of games. *Management Science*, 18(5-part-2):64–79, 1972.

[102] Judea Pearl. *Causality.* Cambridge University Press, 2009.

[103] Wei Peng, Tim Coleman, and Lucas Mentch. Asymptotic distributions and rates of convergence for random forests via generalized U-statistics. *arXiv preprint arXiv:1905.10651*, Preprint:1–62, 2019.

[104] Philipp Petersen and Felix Voigtlaender. Optimal approximation of piecewise smooth functions using deep relu neural networks. *Neural Networks*, 108:296–330, 2018.

[105] Philipp Petersen and Felix Voigtlaender. Optimal learning of high-dimensional classification problems using deep neural networks. *arXiv preprint arXiv:2112.12555*, 2021.

[106] Waseem Rawat and Zenghui Wang. Deep convolutional neural networks for image classification: A comprehensive review. *Neural computation*, 29(9):2352–2449, 2017.

[107] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "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, 2016.

[108] Jürgen Schmidhuber. Deep learning in neural networks: An overview. *Neural Networks*, 61:85–117, 2015.

[109] Johannes Schmidt-Hieber. Nonparametric regression using deep neural networks with relu activation function. *The Annals of Statistics*, 48(4):1875–1897, 2020.

[110] Erwan Scornet, Gérard Biau, Jean-Philippe Vert, et al. Consistency of random forests. *The Annals of Statistics*, 43(4):1716–1741, 2015.

[111] Lloyd S Shapley. A value for n-person games, contributions to the theory of games, 2, 307–317, 1953.

[112] Bernard W Silverman. *Density estimation for statistics and data analysis.* Routledge, 2018.

[113] Dylan Slack, Sophie Hilgard, Emily Jia, Sameer Singh, and Himabindu Lakkaraju. Fooling LIME and SHAP: Adversarial attacks on post hoc explanation methods. In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, pages 180–186, 2020.

[114] Daniel Soudry and Yair Carmon. No bad local minima: Data independent training error guarantees for multilayer neural networks. *arXiv preprint arXiv:1605.08361*, 2016.

[115] Rodney Sparapani, Charles Spanbauer, and Robert McCulloch. Nonparametric machine learning and efficient computation with bayesian additive regression trees: the BART R package. *Journal of Statistical Software*, 97(1):1–66, 2021.

[116] Charles J Stone. Optimal global rates of convergence for nonparametric regression. *The Annals of Statistics*, 10:1040–1053, 1982.

[117] Charles J Stone. The use of polynomial splines and their tensor products in multivariate function estimation. *The Annals of Statistics*, 22(1):118–171, 1994.

[118] Carolin Strobl, Anne-Laure Boulesteix, Achim Zeileis, and Torsten Hothorn. Bias in random forest variable importance measures: Illustrations, sources and a solution. *BMC bioinformatics*, 8(1):1–21, 2007.

[119] Mukund Sundararajan, Kedar Dhamdhere, and Ashish Agarwal. The Shapley Taylor interaction index. In *International Conference on Machine Learning*, pages 9259–9268. PMLR, 2020.

[120] Mukundfau Sundararajan and Amir Najmi. The many Shapley values for model explanation. In *International Conference on Machine Learning*, pages 9269–9278. PMLR, 2020.

[121] Grzegorz Swirszcz, Wojciech Marian Czarnecki, and Razvan Pascanu. Local minima in training of neural networks. *arXiv preprint arXiv:1611.06310*, 2016.

[122] Yan Shuo Tan, Abhineet Agarwal, and Bin Yu. A cautionary tale on fitting decision trees to data from additive models: generalization lower bounds. In *International Conference on Artificial Intelligence and Statistics*, pages 9663–9685. PMLR, 2022.

[123] Yan Shuo Tan, Chandan Singh, Keyan Nasseri, Abhineet Agarwal, and Bin Yu. Fast interpretable greedy-tree sums (figs). *arXiv preprint arXiv:2201.11931*, 2022.

[124] Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1):267–288, 1996.

[125] Che-Ping Tsai, Chih-Kuan Yeh, and Pradeep Ravikumar. Faith-shap: The faithful shapley interaction index. *arXiv preprint arXiv:2203.00870*, 2022.

[126] Alexander B. Tsybakov. Optimal aggregation of classifiers in statistical learning. *The Annals of Statistics*, 32(1):135–166, 2004.

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

[128] Brian Williamson and Jean Feng. Efficient nonparametric statistical inference on population feature importance using Shapley values. In *International Conference on Machine Learning*, pages 10282–10291. PMLR, 2020.

[129] Simon N Wood. Fast stable restricted maximum likelihood and marginal likelihood estimation of semiparametric generalized linear models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 73(1):3–36, 2011.

[130] Marvin N Wright and Andreas Ziegler. Ranger: A fast implementation of random forests for high dimensional data in c++ and r. *arXiv preprint arXiv:1508.04409*, 2015.

[131] Qiang Wu and Ding-Xuan Zhou. SVM soft margin classifiers: Linear programming versus quadratic programming. *Neural Computation*, 17(5):1160–1187, 2005.

[132] Dmitry Yarotsky. Error bounds for approximations with deep ReLU networks. *Neural Networks*, 94:103–114, 2017.

[133] Chih-Kuan Yeh, Kuan-Yun Lee, Frederick Liu, and Pradeep Ravikumar. Threading the needle of on and off-manifold value functions for Shapley explanations. In *International Conference on Artificial Intelligence and Statistics*, pages 1485–1502. PMLR, 2022.

[134] Kyusang Yu, Byeong U Park, and Enno Mammen. Smooth backfitting in generalized additive models. *The Annals of Statistics*, 36(1):228–260, 2008.

[135] Hao Zhang, Yichen Xie, Longjie Zheng, Die Zhang, and Quanshi Zhang. Interpreting multivariate Shapley interactions in DNNs. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(12):10877–10886, May 2021.