THE UNIVERSITY OF CHICAGO

XBART: A SCALABLE STOCHASTIC ALGORITHM FOR SUPERVISED MACHINE LEARNING WITH ADDITIVE TREE ENSEMBLES

A DISSERTATION SUBMITTED TO
THE FACULTY OF THE UNIVERSITY OF CHICAGO
BOOTH SCHOOL OF BUSINESS
IN CANDIDACY FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

BY
JINGYU HE

CHICAGO, ILLINOIS
JUNE 2020

To my parents.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

First and foremost, I would like to express my gratitude to my co-advisors, Prof. P. Richard Hahn and Prof. Nicholas Polson, for the years of dedicated mentorship, inspiring discussions, continuous support, amusing tea time, and invaluable friendship. It has truly been an honor and privilege to be their student, and I thank them for being the best role model as a researcher and a person.

I would like to thank my committee members Prof. Ruey Tsay, Prof. Tengyuan Liang for all the insightful discussions and help. I also thank Prof. Sanjog Misra for providing encouragement, counsel and being an excellent advisor.

I am also grateful to my coauthors Carlos Carvalho, Christopher Glynn, Nikolay Krantsevich, Hedibert Lopes, Jared Murray, David Puelz, Peter Rossi, Maggie Wang, and Saar Yalov for their helpful discussions and collaborations through my research life.

My perfect Ph.D experience could not happen without the generous support from Booth School of Business. I want to thank my fellow doctoral classmates and friends, Jianfei Cao, Hongfan Chen, Chaoxing Dai, Guanhao Feng, Zhonglin Li, Shirley Lu, Yao Lu, Yuexi Wang, Jianeng Xu, Xian Xu. Thanks to Ming Yu and Siyao Liu for making my life so much fun, as well as their four chinchillas for the amusement and allergies they brought to me.

I would like to thank my cousin Huan He for her support and love, hope she enjoys her upcoming Ph.D. journey at Booth. Thanks Liran Chen for studying together at the Regenstein library and the CIE, and for being a steady source of consolation, understanding and happiness.

I want to express my appreciation to J.S. Bach, Ludwig van Beethoven, Johannes Brahms, Frédéric Chopin, Gustav Mahler, Sergei Rachmaninov, and Pyotr Tchaikovsky, for the company of their masterpieces in thousands of hard-working days and nights.

Finally, this dissertation is dedicated to my father Xingqiang He, and my mother Rongzhen Jing. I can never achieve what I have done today without their unconditional love.

ABSTRACT

This dissertation develops a novel stochastic tree ensemble method for nonlinear regression, which I refer to as XBART, short for Accelerated Bayesian Additive Regression Trees. By combining regularization and stochastic search strategies from Bayesian modeling with computationally efficient techniques from recursive partitioning approaches, the new method attains state-of-the-art performance: in many settings it is both faster and more accurate than the widely-used XGBoost algorithm. Via careful simulation studies, I demonstrate that our new approach provides accurate point-wise estimates of the mean function and does so faster than popular alternatives, such as BART, XGBoost and neural networks (using Keras). This dissertation also prove a number of basic theoretical results about the new algorithm, including consistency of the single tree version of the model and stationarity of the Markov chain produced by the ensemble version. Furthermore, I demonstrate that initializing standard Bayesian additive regression trees Markov chain Monte Carlo (MCMC) at XBART-fitted trees considerably improves credible interval coverage and reduces total run-time.

CHAPTER 1

INTRODUCTION

1.1 Overview

Tree-based algorithms for supervised learning, such as Classification and Regression Trees (CART) (Breiman et al., 1984), random forests (Breiman, 1996, 2001), AdaBoost (Freund and Schapire, 1997), gradient boosting (Breiman, 1997; Friedman, 2001, 2002), and XGboost (Chen and Guestrin, 2016) are widely used for applied supervised learning. As a whole, these methods are popular in applied settings due to their speed and accuracy in mean estimation and out-of-sample prediction tasks. One limitation of such methods is their well-known sensitivity to tuning parameters, which require costly cross-validation to optimize. Bayesian additive regression trees (BART) (Chipman et al., 2007, 2010) is a popular Bayesian model-based alternative that is often more accurate than other tree-based methods; specifically, BART boasts valuable robustness to the choice of tuning-parameters. However, relative to random forests and boosting, BART's wider adoption has been slowed by its more severe computational demands, owing to its reliance on a random walk Metropolis-Hastings Markov chain Monte Carlo (MCMC) algorithm.

Despite this limitation, BART has inspired a considerable body of research in recent years. Applications to causal inference (Hill, 2011; Hahn et al., 2020; Logan et al., 2019; Starling et al., 2019), extensions to novel model settings (Murray, 2017; Linero and Yang, 2018; Linero et al., 2019; Kindo et al., 2016; Pratola et al., 2017; Starling et al., 2018; van der Pas and Ročková, 2017), computational innovations (Pratola et al., 2014; Pratola, 2016), and posterior consistency theory (Ročková and Saha, 2019; Rocková, 2019) are some of the notable active research areas. For a more comprehensive review of this literature, see Linero (2017) and Hill et al. (2020). Important precursors of the BART model include Chipman et al. (1998), Denison et al. (1998), and Gramacy and Lee (2008). The primary reason of

BART's success is its ability to learn non-linear main and interaction effects without having to specify the mechanism. A strong regularizing prior is placed on the tree structure to prevent overfitting.

This dissertation contributes to this growing literature by developing a novel stochastic tree ensemble method that combines the hyper-parameter robustness of BART with the efficient recursive computational techniques of traditional tree-based methods. Specifically, I propose a novel tree splitting criterion derived from an integrated-likelihood calculation and suggest a parameter-sampling approach (as opposed to a bootstrapping approach, as in random forests) for avoiding over-fitting. These modifications lead to a tree sampling algorithm that is substantially faster than BART while retaining its state-of-the-art predictive accuracy. This new approach to Bayesian tree models both leads to a substantial speed-up of model fitting and also opens the door for new theoretical results adapted from the literature on random forests (Scornet et al., 2015).

This dissertation proceeds as follows: The remainder of chapter 1 reviews tree based algorithms including CART, random forests, boosting and BART. Chapter 2 introduces XBART for Gaussian mean regression. Chapter 3 shows theory of consistency of XBART. Chapter 4 demonstrates XBART on simulation studies and empirical examples. Chapter 5 extends XBART to binary classification and multinomial classification. Chapter 6 summarizes some conjectures and future research directions.

1.2 Classification and regression tree

Trees are a family of non-parametric models that are built from data directly. The essential idea behind the tree model is to learn a high-dimensional function locally and provide a simple representation by a sequence of decision rules. In specific, a tree partitions the covariate space into a group of rectangles (leaf nodes), then fits a simple model in each leaf, such as a constant or linear regression. A single decision tree is simple to visualize and

understand as it follows how do humans make decisions by a chain of rules.

Consider a nonparametric regression model

$$y = f(\mathbf{x}) + \epsilon \tag{1.1}$$

where y is the response, $\mathbf{x} \in [0,1]^p$ a p-dimensional vector predictors, f an unknown regression function of interest and $\epsilon \sim N(0, \sigma^2)$. Regression tree model assumes that the true function is a tree,

$$f(\mathbf{x}) = g(\mathbf{x}, T, \mu), \tag{1.2}$$

where T denotes a regression tree and μ is a vector of scalars associated to each leaf node of T. Tree T consists of a set of decision rules which define a partition of the covariate space, denote it as $\mathcal{A}_1, \cdots, \mathcal{A}_B$ where B is total number of leaf nodes in tree T. Each node of the partition is associated with a leaf parameter μ_b. Both the partition T and the leaf parameters μ together define a piecewise step function,

$$g(\mathbf{x}) = \sum_{b=1}^{B} \mu_b \mathbb{1}\{x \in \mathcal{A}_b\}, \tag{1.3}$$

where $\mathbb{1}\{\mathbf{x} \in \mathcal{A}_b\}$ is the indicator that x is in leaf node $\mathcal{A}_b$ in tree T.

Despite the existence of multi-way split trees (Kass, 1980), most tree methods have a binary partition of the covariate space. One concern of multi-way split is the combinatorial explosion in practice, furthermore Laurent and Rivest (1976) show that constructing an optimal binary decision tree is NP-complete. Therefore I focus on binary trees in this dissertation.

Figure (1.1) illustrates a simple binary decision tree, where all split lines are parallel to the coordinate axes. To fit a tree model, we first evaluate the split criterion on all cutpoint candidates, pick the best one, and split the entire space into two nodes. Then the

process is repeated for two nodes recursively until some pre-set stopping rules are achieved, see Algorithm 1. Efficiency and interpretability are two primary advantages of a recursive binary tree. It is straightforward to implement the recursion algorithm and interpret split rules even for non-experts.

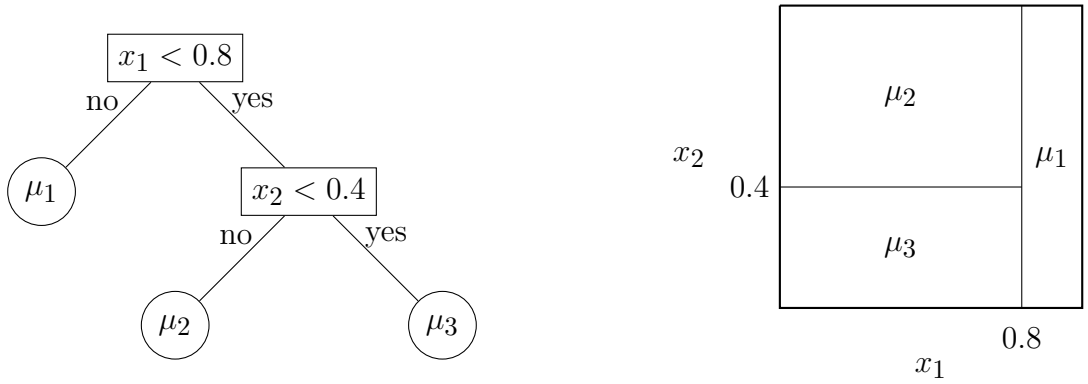


Figure 1.1: (Top) An example binary tree, with internal nodes labelled by their splitting rules and terminal nodes labelled with the corresponding parameters μ_{lb}. (Bottom) The corresponding partition of the sample space and the step function.

Algorithm 1 Pseudocode of growing a tree recursively.

1: Start at a root node.
2: Select a cutpoint by some pre-specified split criterion, then partition the root node into two child nodes according to the selected decision rule.
3: If pre-specified stop conditions are satisfied, stop the algorithm and estimate leaf parameters based on data in each leaf node. Otherwise, apply step 2 to each child node.

Similar to other machine learning approaches, tree models face the problem of bias-variance trade-off (Pedro, 2000). If a tree grows deep, it partitions the space into many smaller nodes and tends to overfit the data. In an extreme case, a tree can achieve zero in-sample error if it partitions the space until each node has one and only one training data. However, it cannot generalize well to out-of-the-sample data. On the other hand, a shallow tree with only a few leaf nodes tends to have a higher bias of estimating the true function. As a result, proper regularization of the tree structure is necessary for better generalization.

This section reviews the Classification and regression tree (CART), one of the most

adopted decision tree models. Consider spitting at variable j and cutpoint s, CART searches for the combination (split point) (j, s) that minimizes the sum of squares,

$$\min_{(j,s)} \left[\min_{c_{\text{left}}} \sum_{x_i \in \mathcal{A}_{\text{Left}}} (y_i - c_{\text{left}})^2 + \min_{c_{\text{right}}} \sum_{x_i \in \mathcal{A}_{\text{right}}} (y_i - c_{\text{right}})^2 \right]. \tag{1.4}$$

For any (j, s), the inner minimization is simply average of data in the nodes,

$$\widehat{c}_{\text{left}} = \text{mean}(y_i \mid x_i \in \mathcal{A}_{\text{left}}), \quad \widehat{c}_{\text{right}} = \text{mean}(y_i \mid x_i \in \mathcal{A}_{\text{right}}). \tag{1.5}$$

At each recursion, scanning all possible split variables and cutpoint (j, s) is feasible and can be done efficiently. After finding the best split, the data is partitioned into two child nodes, and the process is repeated for both child nodes until some terminating conditions are met.

Two remarks are noteworthy regarding the split criterion (1.4) of CART. First, it is a function of *both* cutpoint (j, s) and leaf parameters $(c_{\text{left}}, c_{\text{right}})$. Intuitively, estimation of the leaf parameters is not accurate with noisy data, which might make a bad cutpoint have better criterion evaluation. Second, CART optimizes the split criterion to select cutpoint rather than randomization. In practice, it is hard to conclude that the second-best cutpoint candidate is a bad one, especially when the criterion evaluations of the best two candidates are close. The difference in evaluations might be a result of noise in the data. XBART is inspired by these two remarks above; it estimates cutpoints and leaf parameters separately and selects cutpoints stochastically. Details of XBART will be introduced in the following chapters.

Another key question is, when should the algorithm terminate? Traditional stop rules include setting a max depth for the tree or minimal leaf size. The algorithm terminates if the current node is too deep or has only a few data within it. Another approach is stopping if the decrease of sum-of-squares loss cannot exceed a threshold. These conditions cannot guarantee proper regularization of the tree. Besides that, a frequently used strategy is to

grow a large tree, then *prune* it and remove some nodes. Next we discuss about the pruning algorithm of CART.

Tree size is a key tuning parameter that controls complexity. To prune a grown tree T, we first define a loss function that measures error and complexity by

$$L_\alpha(T) = L(T) + \alpha|T|, \tag{1.6}$$

where $L(T)$ is the error of tree T on the testing set, $|T|$ is number of nodes or tree size, and α is tuning parameter that controls the strength of penalty. If $\alpha = 0$, we do not care about the complexity of the tree but only the prediction error.

Denote $T_B < T_A$ if T_B is a subtree of T_A, or T_B can be achieved by removing or collapsing some leaf nodes of T_A. Breiman et al. (1984) prove that for a given α, there exists a unique best subtree $T_\alpha < T$. Furthermore, $0 = \alpha_0 < \alpha_1, \cdots, \alpha_n < +\infty$ generating intervals $[\alpha_i, \alpha_{i+1}), i = 0, 1, \cdots, n$. The set of all best subtrees $\{T_0, \cdots, T_n\}$ on each interval of α is a sequence such that $T_n < T_{n-1} < \cdots < T_0$. As a result the pruning algorithm can be done recursively, we collapse the internal nodes successively and continue until getting a root (single node). Algorithm 2 presents the pruning procedue. Estimation of α is usually done by cross validation.

Algorithm 2 Prune classification and regression tree.

1: Calculate loss function (1.6) of the current tree $L_\alpha(T)$.
2: **for** each leaf nodes **do**
3: Calculate loss function (1.6) for subtree T_{sub} if the leaf node is pruned, denote it as $L_\alpha(T_{\text{sub}})$.
4: If $L_\alpha(T_{\text{sub}}) \leq L_\alpha(T)$, prune the leaf node. Its ancestor becomes a new leaf node.
5: **end for**
6: Repeat the process for all leaf nodes until none of them can be pruned.

Rather than pruning an overgrown tree, Bayesian CART (Chipman et al., 1998) and Bayesian additive regression trees (Chipman et al., 2010) take a fundamentally different approach. They alleviate overfitting problems by putting a strong shrinkage prior on the

tree space and terminate splitting early. This shrinkage prior is introduced later in this chapter.

Lastly, a single decision tree is prone to overfitting, sensitive to noisy data, and does not generalize well. Ensemble learning methods overcome this problem by combining predictions of many trees. The next two sections review two major ways for combining trees: random forests and boosting.

1.3 Random forests

Random forests (Breiman, 2001) reduce variance of prediction function by bootstrap aggregation approach, or bagging (Breiman, 1996). Bagging averages over trees that are trained on randomly sampled (with replacement) subset of training data. Suppose the training data is $\mathbf{D} = \{(x_1, y_1), \cdots, (x_N, y_N)\}$, for each bootstrap sample $\mathbf{D}^b$, $b = 1, \cdots, B$, a CART tree $T_b(x)$ is fitted using m out of total p randomly drawn covariates. The random forests estimator is simply average of all trees

$$\widehat{f}_B(x) = \frac{1}{B} \sum_{b=1}^{B} T_b(x). \tag{1.7}$$

Each tree in the random forests is independent of others since it fits an independent bootstrap subset of data. Therefore random forests can be parallelized naturally.

Bagging reduces variance of prediction dramatically, to see this, assume the training data $\{(x_i, y_i)\}_{i=1,\cdots,N}$ are drawn independently from probability distribution $\mathcal{P}$. A tree $T_b(x)$ fits the bootstrap data $\{(x_i^*, y_i^*)\}_{i=1,\cdots,N}$. Consider the *theoretical* bagging estimate $E_{\mathcal{P}} T_b(x)$

that draws samples from the real distribution $\mathcal{P}$ rather than the data. We write

$$\begin{aligned} E_{\mathcal{P}}[Y - T_b(x)]^2 &= E_{\mathcal{P}}[Y - E_{\mathcal{P}}T_b(x) + E_{\mathcal{P}}T_b(x) - T_b(x)]^2 \\ &= E_{\mathcal{P}}[Y - E_{\mathcal{P}}T_b(x)]^2 + E_{\mathcal{P}}[T_b(x) - E_{\mathcal{P}}T_b(x)]^2 \\ &\geq E_{\mathcal{P}}[Y - E_{\mathcal{P}}T_b(x)]^2. \end{aligned} \tag{1.8}$$

This shows that population aggregation does not increase mean squared loss. However, it is not true for classification under 0-1 loss since the bias and variance are not addable.

Note that random forests take one more extra step than bagging; it samples both data observations and covariate variables. If variables have positive pairwise correlation ρ and variance σ^2, the variance of the average is

$$\rho\sigma^2 + \frac{1-\rho}{B}\sigma^2, \tag{1.9}$$

where the first term never reduces as B increases. Random forests fit trees on a subset of variables to reduce the correlation between the trees. Algorithm 3 summarizes the fitting procedure of random forests.

Algorithm 3 Random Forests.

1: Draw a size N bootstrap sample (with replacement) from the training data.
2: Grow a CART tree T_b on the bootstrap data, for each recursion, pick the best cutpoint based on m randomly drawn variables from all p variables.
3: Repeat step 1-2 for $b = 1, \cdots, B$.

For more discussion about bagging or random forests, see Hastie et al. (2005). Other notable reference about random forests include Scornet et al. (2015) for proving consistency of random forests in L^2 norm. Wright and Ziegler (2015) is a fast R package of random forests for high dimensional data.

1.4 Boosting

Boosting tree has shown remarkable results for a variety of prediction problems. It is a solution to a classification problem: is it possible to boost weak classifiers (the one whose prediction error rate is slightly better than random guess) to a powerful classifier? The idea behind boosting is to apply the weak learners repeatedly to generate a sequence of classifiers; each one is based on the prediction of all previous classifiers. The final prediction is then combined through a weighted vote of this committee. Initially, boosting is proposed to solve classification problems, and the idea applies to regression problems as well. The first boosting procedure are Schapire (1990); Freund (1995). Gradient boosting regression tree (GBDT) (Friedman, 2001, 2002) and Extreme Gradient Boosting (XGboost) (Chen and Guestrin, 2016) are the current state-of-the-art tree-based algorithms. Although both algorithms optimize loss function and approximate gradients by trees, there are a number of major differences between them. I review both methods briefly in this section and emphasis the differences. Other notable boosting algorithms includes AdaBoost (Freund et al., 1996; Freund and Schapire, 1997), LightGBM (Ke et al., 2017). I focus on boosting trees in this dissertation while boosting applies to general machine learning algorithms. For more discussion from the perspective of learning theory and PAC learnability, see Valiant (1984); Schapire (1990); Shalev-Shwartz and Ben-David (2014). For a comprehensive review of ensemble methods, see Zhou (2012) and Hastie et al. (2005).

1.4.1 Gradient boosting

Gradient boosting (Friedman, 2001) works as its name shown, it takes gradient descend method in a functional space. It has been observed to be successful particularly for tree models (GBDT). In specific, gradient boosting trees fit a new tree to the negative gradient of the loss function and produce a sequence of trees.

Suppose the aim is to learn a sum of trees model $f^M(x) = f_1(x) + \cdots + f_M(x)$ to minimize

the loss function on data $\{(y_i, x_i)\}_{i=1,\cdots,N}$

$$\begin{aligned} L(f^M) &= \sum_{i=1}^{N} L(y_i, f^M(x_i)) \\ &= \sum_{i=1}^{N} L(y_i, f^{M-1}(x_i) + f_M(x_i)) \end{aligned} \tag{1.10}$$

Steepest descent method optimizes the objective function by multiple iterations. The m-th iteration takes step in the direction of $-\alpha \mathbf{g}_m$ where α is learning rate and the i-th element of the gradient vector is

$$g_{m,i} = -\left[\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)}\right]_{f(x)=f^{m-1}(x)}. \tag{1.11}$$

The fitted function is then updated to

$$f^m = f^{m-1} - \alpha \mathbf{g}_m. \tag{1.12}$$

Optimizing (1.14) can be easily achieved by steepest descent. However, notice that the gradient is defined on the training set, to gain better generalization we fit a tree to approximate the gradient such that

$$T(x \mid \widehat{\Theta}) = \arg\min_{\Theta} \sum_{i=1}^{N} (-g_{m,i} - T(x_i \mid \Theta))^2, \tag{1.13}$$

where we fit tree T with parameters $\widehat{\Theta}$. Loop over M iterations, the corresponding sum of trees is learnt by algorithm 4.

Algorithm 4 Gradient Boosting Trees.

Initialization: $f^0(x) = \arg\min_c \sum_{i=1}^{N} L(y, c)$.
for $m = 1, 2, \cdots, M$ **do**
    for $i = 1, 2, \cdots, N$ **do**
        Calculate

$$r_{m,i} = -\left[\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)}\right]_{f(x)=f^{m-1}(x)}.$$

    end for
    Fit a regression tree to $r_{m,i}$, get leaf nodes $R_{mj}, j = 1, \cdots, J$ where J is total number of leaf nodes.
    for $j = 1, 2, \cdots, J$ **do**
        Calculate

$$c_{m,j} = \arg\min_c \sum_{x_i \in R_{m,j}} L(y_i, f^{m-1}(x_i) + c).$$

    end for
    The m-th tree is $\sum_{j=1}^{J} c_{m,j} I(x \in R_{m,j})$, update sum of trees: $f^m(x) = f^{m-1}(x) + \sum_{j=1}^{J} c_{m,j} I(x \in R_{m,j})$.
end for
The output predictive function is $f^M(x)$.

1.4.2 XGboost

XGboost (Chen and Guestrin, 2016) stands for Extreme Gradient Boosting; it is an implementation of the gradient boosting methods while more accurate approximation and engineering optimization are applied. XGboost optimizes the loss function by Newton's method while the regular gradient boosting trees uses gradient descend. Furthermore, it employs a number of tricks that make it is exceptionally efficient and is widely used in many applications.

XGboost computes second-order gradients of the loss function to gain more information of the direction of gradients. Not like GBDT, XGboost has penalty of tree complexity in the loss function to improve model generalization. The loss function of XGboost has the form of

$$
\begin{aligned}
L(f^M) &= \sum_{i=1}^{N} L(y_i, f^M(x_i)) + \sum_{m=1}^{M} \Omega(f_m) \\
\text{where} \quad \Omega(f_m) &= \gamma G_m + \frac{1}{2}\lambda ||\mu(f_m)||^2.
\end{aligned} \tag{1.14}
$$

Here $\Omega(f_m)$ term penalizes the complexity of the regression tree model, where G_m is number of leaf nodes of tree f_m and $\mu(f_m)$ is a vector of all leaf parameters of the tree. Comparing to GBDT, the extra penalization term in the objective function helps to smooth the final tree to avoid overfitting.

Another notable improvement of XGboost is that it expands the loss function to the second order derivative. From respectives of optimization, GBDT optimizes the objective function by gradient descent and fit CART to negative gradients, but XGboost approximates the change of loss function by second order derivatives and finds corresponding analytical solutions. Consider Taylor expansion of the loss function at a new tree $f_M(x)$,

$$
\begin{aligned}
L(f^M) &= \sum_{i=1}^{N} L(y_i, f^M(x_i)) + \Omega(f_M) \\
&= \sum_{i=1}^{N} L(y_i, f^{M-1}(x_i)) + g_i f_M(x_i) + \frac{1}{2} h_i f_M^2(x_i) + \Omega(f_M) \\
\text{where} \quad g_i &= \left[\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)}\right]_{f(x)=f^{m-1}(x)} \text{and } h_i = \left[\frac{\partial^2 L(y_i, f(x_i))}{\partial f(x_i)^2}\right]_{f(x)=f^{m-1}(x)}.
\end{aligned} \tag{1.15}
$$

Remove the constants, the simplified objective function is

$$
\begin{aligned}
\tilde{L}(f^M) &= \sum_{i=1}^{N} g_i f_M(x_i) + \frac{1}{2} h_i f_M^2(x_i) + \gamma G_m + \frac{1}{2}\lambda \sum_{i=1}^{G_m} \mu_i^2 \\
&= \sum_{j=1}^{G_m} \left[\left(\sum_{i \in I_j} g_i \right) \mu_j + \frac{1}{2} \left(\sum_{i \in I_j} h_i + \lambda \right) \mu_j^2 + \gamma G_m \right]
\end{aligned} \tag{1.16}
$$

where I_j denotes set of data in the j-th leaf node. The optimal leaf parameter of leaf j is

given by

$$\mu_j^* = -\frac{\sum_{i \in I_j} g_i}{\sum_{i \in I_j} h_i + \lambda} \tag{1.17}$$

and the corresponding optimal value of the objective function is

$$\tilde{L}^* = -\frac{1}{2} \sum_{j=1}^{G_m} \frac{\left(\sum_{i \in I_j} g_i \right)^2}{\sum_{i \in I_j} h_i + \lambda} + \gamma G_m. \tag{1.18}$$

Equation (1.18) can be used as split criterion to evaluate goodness of a cutpoint candidate. Beyond penalty of tree, XGboost deploys similar covariate variables subsampling strategy as random forest to prevent overfitting.

XGboost deploys tricks of computer system to improve the performance further, such as missing value handling, store data in compressed column format for parallel computing, cache-aware access, and blocks for out-of-core computation. For more details, see Chen and Guestrin (2016). Nielsen (2016) attempt to explain the reason of XGboost's remarkable performance.

1.5 Bayesian CART and Bayesian additive regression trees

The earliest attempt to apply Bayesian approach to tree models is arguably Buntine (1990). Chipman et al. (1998) and Denison et al. (1998) propose Bayesian classification and regression tree models independently. Denison et al. (1998) estimate the model via maximum likelihood but not a full Bayesian inference approach. Chipman et al. (1998) propose a shrinkage prior on tree space and a random-walk Metropolis-Hastings algorithm to sample from the full posterior. In this dissertation, I will more closely follow that outlined by Chipman et al. (1998). For a book treatment of Bayesian nonlinear classification and regression models, see Denison et al. (2002).

Bayesian additive regression trees (BART) (Chipman et al., 2010) uses a sum of trees

model, imposes a regularizing prior that keeps individual tree effect small. The BART model uses the same prior forms of Bayesian CART, although the choice of hyper-parameters is markedly different. BART is not merely a simple Bayesian version of random forest or boosting, in which prior distributions are placed over parameters. Instead, the Bayesian perspective inspires a fundamentally new tree growing split criterion. It achieves state-of-art prediction accuracy with robustness to the choice of hyper-parameters.

In the following text, I view the BART model and sampling algorithm. Consider the fundamental problem of learning an unknown function that predicts y by p dimensional covariate vector $\mathbf{x}$,

$$y = f(\mathbf{x}) + \epsilon \tag{1.19}$$

where $\mathbf{x} = (x^{(1)}, \cdots x^{(p)})$ is a vector of p covariates, ϵ is assumed to be independent $N(0, \sigma^2)$ distributed and $f(\mathbf{x})$ is an unknown mean function $f(\mathbf{x}) = \mathbb{E}[Y \mid \mathbf{x}]$. BART prior assume f to be a sum of regression trees

$$f(\mathbf{x}) = \sum_{i=1}^{L} g_l(\mathbf{x}, T_l, \mu_l), \tag{1.20}$$

where T_l denotes a regression tree (a partition of the space of X) and μ_l is a vector of scalars associated to each leaf node of T_l. Each tree T_l consists of a set of decision rules which define a partition of the covariate space, denote it as $\mathcal{A}_1, \cdots, \mathcal{A}_{B(l)}$ where $B(l)$ is total number of leaf nodes in tree T_l. Each node of the partition is associated with a leaf parameter μ_{lb}. Both the partition T_l and the leaf parameters μ_l together define a piecewise step function,

$$g_l(\mathbf{x}) = \sum_{b=1}^{B(l)} \mu_{lb} \mathbb{1}\{x \in \mathcal{A}_b\}, \tag{1.21}$$

where $\mathbb{1}\{\mathbf{x} \in \mathcal{A}_b\}$ is the indicator that x is in leaf node $\mathcal{A}_b$ in tree T_l. See Figure (1.1) for an illustration.

Following Chipman et al. (2010), the tree prior $p(T_l)$ is a branching process prior which can be interpreted as a tree generating process with three components.

$$\begin{aligned} p\left((T_1,\mu_1),\cdots,(T_L,\mu_L),\sigma^2\right) &= \left[\prod_{l=1}^{L} p(T_l,\mu_l)\right] p(\sigma^2) \\ &= \left[\prod_{l=1}^{L} p(\mu_l \mid T_l)p(T_l)\right] p(\sigma^2) \\ &= \left[\prod_{l=1}^{L}\prod_{b=1}^{B(l)} (\mu_{lb} \mid T_l)p(T_l)\right] p(\sigma^2) \end{aligned} \tag{1.22}$$

- $p(T_l)$: Each node at depth d has children with probability $\alpha(1+d)^{-\beta}$. Hyperparameters $\alpha \in (0,1)$ and $\beta \in [0,\infty)$ controls prior belief of size of the tree. Chipman et al. (2010) recommend $\alpha = 0.95$ and $\beta = 2$ to enforce small trees. If a node has children, it is assigned a split rule $x_i \leq M_b$. We assume uniform prior on the splitting variable x_i and also uniform prior on a discrete set of possible splitting values M_b.

- $p(\mu_{lb} \mid T_l)$: Leaf parameter μ_{lb} is assumed independent and identical Gaussian prior $N(0,\tau)$. Note that the implied prior on response Y is $N(0,\tau L)$.

- $p(\sigma^2)$: The residual standard deviation is assumed a standard inverse-Gamma prior $iG(a,b)$.

Note that the regularizing prior $p(T_l)$ has a strong penalty on complex trees if the default parameters are chosen. It plays the same role as the penalty term in the loss function of XGboost.

Chipman et al. (2010) explore the posterior of BART via a Bayesian backfitting Markov chain Monte Carlo (MCMC) scheme of Hastie et al. (2000). Leaf parameter μ_l and residual variance σ^2 are drawn from standard conjugate posterior and tree structure T_l is updated by a random walk Metropolis-Hastings (MH) step. Let $\mathcal{T} = \{T_1,\cdots,T_L\}$ and $\mathcal{M} = \{\mu_1,\cdots,\mu_l\}$

denote the set of all trees and leaf parameter vectors respectively. Set $\mathcal{T}_{-l} = \mathcal{T}_l/T_l$ denotes all trees except the l-th tree. Notation $\mathcal{M}_{-l}$ is analogous. The Bayesian backfitting algorithm is presented as follows,

1. $T_l, \mu_l \mid \mathcal{T}_{-l}, \mathcal{M}_{-l}, \sigma^2, \mathrm{y}$, for $l = 1, \ldots, L$, which is done compositionally (for each l) as
   (a) $T_l \mid \mathcal{T}_{-l}, \mathcal{M}_{-l}, \sigma^2, \mathrm{y}$,
   (b) $\mu_l \mid \mathcal{T}, \mathcal{M}_{-l}, \sigma^2, \mathrm{y}$,
2. $\sigma^2 \mid \mathcal{T}, \mathcal{M}, \mathrm{y}$.

Taking advantage of the additive structure of the model, these updates can be written as

1. $T_l, \mu_l \mid \mathrm{r}_l, \sigma^2$, for $l = 1, \ldots, L$, which is done compositionally (for each l) as
   (a) $T_l \mid \mathrm{r}_l, \sigma^2$,
   (b) $\mu_l \mid T_l, \mathrm{r}_l, \sigma^2$,
2. $\sigma^2 \mid \mathrm{r}$.

for "residuals" defined as

$$\mathrm{r}_l^{(k+1)} \equiv \mathrm{y} - \sum_{l'<l} g(\mathbf{X}; T_{l'}, \mu_{l'})^{(k+1)} - \sum_{l'>l} g(\mathbf{X}; T_{l'}, \mu_{l'})^{(k)},$$

and

$$\mathrm{r}^{(k)} \equiv \mathrm{y} - \sum_{l=1}^{L} g(\mathbf{X}; T_l, \mu_l)^{(k)}, \tag{1.23}$$

The draw of $T_l \mid \mathrm{r}_l, \sigma^2$ can be achieved using the random walk Metropolis-Hastings algorithm of Chipman et al. (1998) which proposes a tree based on the current tree via four possible moves. The four moves are as follows:

1. Grow: Uniformly draw a leaf node and split it to two children.

2. Prune: Uniformly draw a leaf node and collapse it.

3. Swap: Swap split rules of two interior nodes.

4. Change: Uniformly draw an interior node and update its split rule.

The probability of four moves is 0.25, 0.25, 0.1, and 0.4, respectively. Note that Grow and Prune only operate at a leaf node, Swap and Change make it possible to update nodes on the top when the current tree is deep. Figure (1.2) illustrates the random walk proposals.

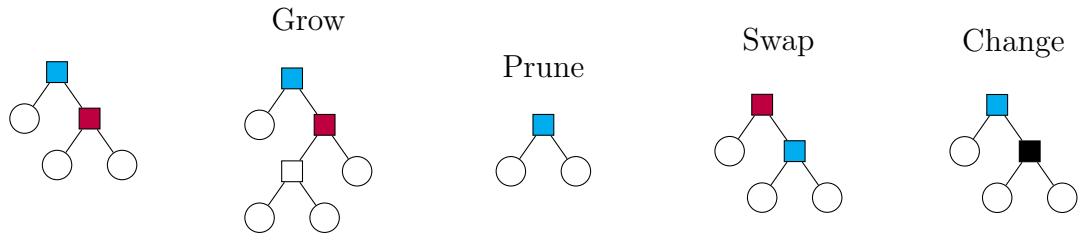


Figure 1.2: Four possible moves for the random walk proposal of BART MCMC. Square nodes are intermediate nodes and circles are leaf nodes. We denote swapping decision rules of two nodes by swapping their color in the graph.

Denote the proposal as T_l', the probability to accept the proposal is

$$p(T_l \to T_l') = \min\left\{1, \frac{q(T_l', T_l)}{q(T_l, T_l')} \frac{L(\mathrm{r}_l \mid X, T_l', \mu_l)}{L(\mathrm{r}_l \mid X, T_l, \mu_l)} \frac{P(T_l')}{P(T_l)}\right\}. \tag{1.24}$$

The transition ratio $\frac{q(T_l', T_l)}{q(T_l, T_l')}$ is calculated by the probability of four moves.

Next, the draw of μ_l is only independent draws of the leaf node parameter from a conjugate normal posterior distribution. Then the residual updates for the subsequent $j+1$-th tree to fit. Lastly, the draw of σ^2 is a draw from an inverse gamma distribution, and it is the standard sampling scheme of residual variance.

The backfitting MCMC algorithm mixes surprisingly much better than the single tree model of Chipman et al. (1998). If we only consider a single tree model, the MCMC algorithm tends to stuck in a local neighborhood of a giant tree. In contrast, the backfitting MCMC

of BART regrows each tree in every iteration, which helps to explore the posterior better. Furthermore, experiments show that the MCMC algorithm gives similar robust results if we restart the fitting process with different random seeds.

1.6 Comparison of tree-based methods

This section compares popular tree algorithms with XBART. Table 1.1 lists comparisons of CART, random forest, boosting, XBART, and BART on various aspects. CART, random forests, and boosting take likelihood equation 1.4 as the criterion, which is both a function of cutpoint and leaf parameters. Thus they have to optimize cutpoint and leaf parameters jointly. BART and XBART integrate out leaf parameters for split criterion and estimate them after growing the tree. Boosting, XBART and BART fit trees sequentially and take the sum of them, while random forest averages (aggregation) of multiple independent trees. Here iteration indicates whether the algorithm learns multiple forests or not, both XBART and BART sample multiple forests sequentially. Finally, BART fits trees by an inefficient random-walk Metropolis-Hastings algorithm rather than a recursive partition algorithm and has much higher computational demands.

Table 1.1: Comparison of tree-based methods.

	CART	RF	XGBoost	XBART	BART
Leaf Parameters	optimized with splits	optimized with splits	optimized with splits	integrate out at split then sample	integrate out at split then sample
Criteria	likelihood	likelihood	likelihood	marginal likelihood	marginal likelihood
Aggregation	No	aggregation of trees	No	aggregation of forests	aggregation of forests
Sequential fitting	No	No	Yes	Yes	Yes
Iterations	No	No	No	Yes	Yes
Recursion	Yes	Yes	Yes	Yes	No

1.7 Contributions of this dissertation

BART has shown a promising prediction model in many applications recently, but its wider adoption has been slowed by severe computational demands relative to alternatives. The random walk Metropolis-Hastings Markov-chain Monte Carlo sampling approach makes posterior exploration inefficient. This dissertation develops a variant of BART that gives rapid posterior estimation, making it as fast as XGboost and still retaining BART's accuracy and robustness to hyper-parameters. XBART fits tree recursively similar to CART while the regularizing prior of BART is attained. The algorithm of XBART for Gaussian regression and computational tricks are summarized in He et al. (2019) and He and Hahn (2020). In addition to computation, this dissertation also contributes to the theory of XBART. I prove consistency in $\mathcal{L}^2$ norm sense for a simplified model of XBART. Lastly, this dissertation concludes with discussion about directions for future research.

CHAPTER 2

XBART REGRESSION

Let y denote a continuous outcome in $\mathbb{R}^1$. Our goal is to predict y by a length p covariate vector $\mathbf{x} = (\mathbf{x}^{(1)}, \cdots, \mathbf{x}^{(p)})$. I demonstrate the general framework of the algorithm in this section; details of the regression setting are introduced in the following section. This section begin with the algorithm for fitting a single tree, then proceed to tree ensembles, or forests. Next I show computational strategies in implementing an efficient algorithm. Lastly, I discuss about the model implied by the recursively growing algorithm.

2.1 Fitting a single tree recursively and stochastically

A tree T_l $(1 \leq l \leq L)$ is a set of decision rules defining a rectangular partition of the covariate space $\{\mathcal{A}_{l1}, \cdots, \mathcal{A}_{lB_l}\}$. Each terminal node $\mathcal{A}_{lb}$ is associated with a vector of leaf parameter μ_{lb}. I denote a tree $g(\mathbf{x}; T_l, \mu_l)$ where $\mu_l = (\mu_{l1}, \cdots, \mu_{lB_l})$ is a vector of all leaf parameters. Each pair of (T_l, μ_l) parameterizes a step function on covariate space,

$$g(\mathbf{x}; T_l, \mu_l) = \mu_{lb}, \quad \text{if } \mathbf{x} \in \mathcal{A}_{lb}.$$

Figure 1.1 depicts a regression tree. The left panel shows a decision rule structure, and the right panel plots the corresponding partition of the space as well as the associated leaf parameters. Ideally, a tree partitions the space into fine irregular mesh where outcome observations within each leaf node (defining a hyperrectangle in covariate space) are nearly homogeneous. Predicting the outcome of a new observation then follows according to the leaf parameter associated with the node it falls within.

Usually, a tree algorithm learns the partition by recursively partitioning the data set one $\mathbf{x}$ variable at a time, and partitions the child nodes similarly as parent node until reaching terminating conditions. Algorithm 1 gives the pseudocode for the essential step of fitting a

tree recursively.

Most popular tree-based methods deploy Algorithm 1 varying in terms of their split criteria and stopping conditions. Essentially, split criteria are functions of a cutpoint, measuring homogeneity within the two child nodes produced by the implied split. CART (Breiman et al., 1984), for instance, uses mean squared error to define its split criterion. Note also that CART and other recursive tree algorithms *optimize* their split criteria over the set of all cutpoint candidates in order to select a cutpoint. Frequently used stop conditions include maximum depth of a tree, minimal number of data observations within leaf node or a threshold for percent change of split criterion from parent to child nodes. Despite its simplicity and elegance, the exhaustive search approach tends to grow a tree unnecessarily deep, thereby over-fitting the data, thus pruning (the merge of some leaf nodes via a bottom up process) after model fitting is usual necessary to further throttle model complexity. In contast, BART provides a Bayesian perspective on tree models by using a regularization prior on tree space which preferences smaller trees. The excellent empirical performance of BART suggests that this prior regularization is beneficial. However, BART does not fit trees recursively but rather explores the posterior distribution over trees via a Bayesian backfitting MCMC scheme, which is computationally intensive, especially for large scale data. This computational burden hampers BART's usefulness on large scale data.

Inspired by both CART and BART, our proposed XBART framework combines strength from each. XBART is a sum of trees in which each individual tree is grown according to a model-based split criterion derived from an integrated-likelihood, but draws a cutpoint with probability proportional to split criterion. Furthermore, XBART stochastically terminates the growing process, the so-called *no-split* option; this allows trees to stop growing before reaching stop conditions and helps to prevent over-fitting.

Let's briefly clarify notation before turning to details of the algorithm. The predictor matrix $\mathbf{X}$, with dimension $n \times p$, defines a set of splitting rule candidates, denoted $\mathcal{C}$, which

are indexed as (j, k) where $j = 1, \ldots p$ indexes a variable (column) of $\mathbf{X}$ and k indexes a set of candidate cutpoints. Let $|\mathcal{C}|$ denote the total number of candidate splitting rules (cutpoints). Let Φ denote prior hyper-parameters and Ψ denote model parameters, which are both considered given and fixed when growing a single tree (a distinction that will be clarified in specific examples).

Inspired by the Bayesian approach, we define a likelihood $L(\mathrm{y}_b; \mu_b, \Psi_b)$ on *one* leaf node b with leaf-specific parameter μ_b and other model parameters Φ_b (given and fixed during the tree growing process). In the following text, we omit subscript b for simplicity. For instance, the likelihood can be Gaussian for regression, details of which are given in section 3. The leaf parameter μ is given a prior $\pi(\mu \mid \Phi)$. We derive our split criterion by integrating out the leaf parameter μ:

$$m(s \mid \Phi, \Psi) := \int L(\mathrm{y}; \mu, \Psi)\pi(\mu \mid \Phi)d\mu, \tag{2.1}$$

where s represents sufficient statistics of data y falling in the current node.

A cutpoint (j, k) partitions the current node to left and right child nodes, with sufficient statistics s^l_{jk} and s^r_{jk} calculated based on y respectively. Assuming that observations in separate leaf nodes are independent, the joint integrated-likelihood is simply the product of the two sides

$$m(s^l_{jk} \mid \Phi, \Psi)m(s^r_{jk} \mid \Phi, \Psi), \tag{2.2}$$

which defines the split criterion for cutpoint (j, k). Furthermore, the split criterion for *no-split* is defined as

$$|\mathcal{C}| \left(\frac{(1+d)^\beta}{\alpha} - 1 \right) m(s^\emptyset \mid \Phi, \Psi) \tag{2.3}$$

where d is depth of the current node, $s^\emptyset$ represents sufficient statistics on current node and α, β are hyper-parameters. The weight of no-split increases significantly as a tree grows deeper, strongly penalizing deep trees and thereby favoring "weak learners" in the parlance

of the boosting literature. Besides the no-split option, traditional stopping conditions are also imposed, such as setting a maximum depth or a minimal number of observations per node.

Once the split criterion has been evaluated at all cutpoint candidates, as well as the no-split option (retained from the previous split), a cutpoint is randomly sampled with probability proportional to its split criterion value. Unlike non-model-based tree algorithms, XBART's split criterion has a natural probabilistic interpretation as it is derived as an integrated-likelihood and sampling follows according to Bayes rule. Indeed, the prior probability of the no-split option (after normalizing with respect to all other cutpoint candidates) matches that of the tree prior used in standard BART (Chipman et al., 2010).

Theorem 1. *The a priori probability of splitting at a node of depth d implied by algorithm 5 (grow-from-root) is $\alpha(1+d)^{-\beta}$.*

Proof. The proof is by direct calculation. Ignore the data contribution from the marginal likelihood function $m(\cdot)$ by setting these terms to 1 in the expressions in line 8. Accordingly, the probability of any cutpoint $(j,k) \in \mathcal{C}$ has prior probability proportional to 1 and the prior probability of no-split is proportional to $|\mathcal{C}|\left(\frac{(1+d)^{\beta}}{\alpha} - 1\right)$. Therefore, the total weight given to splitting is $\sum_{\mathcal{C}} 1 = |\mathcal{C}|$ and normalizing gives the prior probability of splitting as

$$\frac{\text{split weight}}{\text{split weight} + \text{no split weight}} = \frac{|\mathcal{C}|}{|\mathcal{C}|\left(\frac{(1+d)^{\beta}}{\alpha} - 1\right) + |\mathcal{C}|} = \alpha(1+d)^{-\beta}.$$

□

Remark Observe that sampling a cutpoint stochastically rather than optimizing the split criterion substantially improves performance, based on simulation experiments. Intuitively, sampling cutpoints rather than optimizing them helps alleviate over-fitting and encourages wider exploration of the tree space.

Algorithm 5 GrowFromRoot

procedure GFR(y, $\mathbf{X}, \Psi, \Phi$, d, T, `node`)
outcome Modifies T by adding nodes and sampling associated leaf parameters.
    $s^{\emptyset} \leftarrow s(y, \mathbf{X}, \Psi, \mathcal{C}, \text{all})$. ▷ Compute sufficient statistic of not splitting.
    for $(j,k) \in \mathcal{C}$ **do** ▷ Calculated recursively in a single sweep of the data per variable.
        $s^{l}_{jk} \leftarrow s(y, \mathbf{X}, \Psi, \mathcal{C}, j, k, \text{left})$. ▷ Compute sufficient statistic of left candidate node.
        $s^{r}_{jk} \leftarrow s(y, \mathbf{X}, \Psi, \mathcal{C}, j, k, \text{right})$. ▷ Compute sufficient statistic of right candidate node.
    end for
    Sample cutpoint (j,k) proportional to integrated likelihoods

$$m(s^{l}_{jk})m(s^{r}_{jk})$$

or

$$|\mathcal{C}| \left(\frac{(1+d)^{\beta}}{\alpha} - 1 \right) m(s^{\emptyset})$$

for the no-split option.
    if no-split is selected or stop conditions are reached **then**
        $\theta_{\texttt{node}} \leftarrow$ SampleParameters($s^{\emptyset}$)
        return.
    else
        Create two new nodes, denoted `left_node` and `right_node`, and growing T by designating them as the current node's (`node`) children.
        Sift the data into left and right parts, according to the selected cutpoint $x_{ij'} \leq x^{*}_{kj}$ and $x_{ij'} > x^{*}_{kj}$, respectively, where x^{*}_{kj} is the value corresponding to the sampled cutpoint (j,k).
        GFR($\text{y}_{\text{left}}, \mathbf{X}_{\text{left}}, \Psi, \Phi$, $d+1$, T, `left_node`)
        GFR($\text{y}_{\text{right}}, \mathbf{X}_{\text{right}}, \Psi, \Phi$, $d+1$,T, `right_node`)
    end if
end procedure

Once the no-split option is selected or stopping conditions are met, the current node becomes a terminating node or a *leaf*. The leaf parameter μ_{lb} is then updated based on data in the current leaf by standard Bayesian posterior sampling using conjugate likelihood and prior. Algorithm 5 presents the GrowFromRoot function that grows a single tree for XBART.

In summary, XBART implies the same prior probability as BART by the design of the split criterion, while fitting a tree recursively. This framework enjoys great flexibility of potential applications. In this dissertation I focus on the case where the marginal likelihood arises from a Gaussian mean regression model (section 2.6), but it is straightforward to

substitute an integrated-likelihood function $m(s \mid \Phi, \Psi)$ from other models.

2.2 Forest

A forest, as the name sugggests, is an ensemble of trees. Ensemble learning is a widely used technique to combine multiple learning algorithms to improve the overall prediction accuracy. Random forests (Breiman, 1996, 2001) takes an average of trees fitting bootstrap resampled data; adaBoost (Freund and Schapire, 1997), gradient boosting (Breiman, 1997; Friedman, 2001, 2002) and BART (Chipman et al., 2010) explicitly fit a sum of trees. XBART Gaussian nonlinear regression takes the same sum of trees form as BART does:

$$f(\mathbf{x}) = \sum_{i=1}^{L} g_l(\mathbf{x}; T_l, \mu_l), \tag{2.4}$$

where step function $g_l(\mathbf{x}; T_l, \mu_l)$ denotes a tree defined by partition T_l and corresponding leaf parameters μ_l.

The stochastic tree ensemble method proceeds similarly to an MCMC algorithm. Suppose we draw I samples (sweeps) of forests, and each forest contains L trees. When updating the h-th tree in the $iter$-th iteration, a new tree is grown to fit the *partial* residuals $\mathrm{r}_h^{(iter)}$, which are defined as the partial residual of the target (y) after subtracting off the contribution of all the other trees. Specifically, the partial residuals are defined as

$$\mathrm{r}_h^{(iter+1)} \equiv \mathrm{y} - \sum_{h'<h} g(\mathbf{X}; T_{h'}, \mu_{h'})^{(iter+1)} - \sum_{h'>h} g(\mathbf{X}; T_{h'}, \mu_{h'})^{(iter)},$$

while the total residual is taken with respect to all trees

$$\tilde{\mathrm{r}}_h^{(iter+1)} \equiv \mathrm{r}_h^{(iter+1)} - g(\mathbf{X}; T_h, \mu_h)^{(iter+1)}.$$

Algorithm 6 draws I samples of the forest; we refer to one pass of the algorithm, sampling

each tree, as a sweep. Every tree is updated by algorithm 5 GrowFromRoot in each iteration, where the "data" are the partial residuals as calculated at the current iteration. Extra model-dependent non-tree parameters Ψ are updated in between sampling each tree; specifically, the residual standard deviation σ is sampled after each tree. Details are summarized in the following sections.

Algorithm 6 Accelerated Bayesian Additive Regression Trees (XBART)

procedure XBART($\mathrm{y}, \mathbf{X}, \Phi, L, I$)
output I posterior draws of a forest (and associated leaf parameters) comprising L trees.
    Initialize Ψ, partial fit R_h^{iter}.
    for *iter* in 1 to I **do**
        for h in 1 to L **do**
            Create `new_node`.
            Initialize tree T_h^{iter} consisting only of `new_node`.
            GFR(r_h^{iter}, $\mathbf{X}, \Psi, \Phi$, T_h^{iter}, $d = 0$, `new_node`)
            Update r_{h+1}^{iter} (or r_1^{iter+1} if $h = L$), the target to fit for the next tree and full residual $\tilde{\mathrm{r}}_h^{(iter+1)}$.
            Sample non-tree parameters of Ψ, probably based on the full residual $\tilde{\mathrm{r}}_h^{(iter+1)}$.
        end for
    end for
end procedure

Next, I study theoretical properties of Algorithm 6.

Theorem 2. *The algorithm sampling $F = \{T_h\}_{1 \leq h \leq L}$ is a finite-state Markov chain with stationary distribution.*

Proof. We consider the process $F = \{T_h\}_{1 \leq h \leq L}$. Leaf parameters $\mu = \{\mu_h\}_{1 \leq h \leq L}$ are updated conditional on forest F based on standard conjugate Bayesian posterior draws and are not to be regarded as part of the Markov chain of the forest.

First, observe that each tree has a maximum depth and all cutpoint candidates are defined on a finite covariate matrix $\mathbf{X}$. Therefore a single tree has finite states. The forest is an ensemble of a finite number of trees, thus has a finite number of states as well. The probability of the GrowFromRoot algorithm drawing a single tree is a product of the probabilities of drawing specific cutpoints at each node, thus $p(T_j \mid T_{-j}, \mu_{-j}) > 0$. In addition, the

GrowFromRoot algorithm updates T_h^{iter} fitting r_h^{iter}, which is defined by trees and leaf parameters with subscript $1 < j < h$ in $iter$-th sweeps and $h+1 < j < L$ in $(iter-1)$-th sweeps. Therefore, the forest process is a finite-state Markov chain.

Second, I claim that because the split criterion is defined by an integrated likelihood, it has non-zero evaluations for all cutpoint candidates (including the no-split option) given fitting data r_h^{iter}. Let $T_{-j} = \{T_h\}_{1\leq h\leq L}/T_j$ and $\mu_{-j} = \{\mu_h\}_{1\leq h\leq L}/\mu_j$ be trees and leaf parameters excepting the j-th one, respectively. We have

$$p(T_j \mid T_{-j}) = \int\int p(T_j \mid \mathrm{y}, T_{-j}, \mu_{-j}, \Psi) f(\mu_{-j} \mid \mathrm{y}, T_{-j}) f(\Psi) d\Psi d\mu_{-j} > 0, \tag{2.5}$$

since $f(\mu_{-j} \mid \mathrm{y}, T_{-j})$, the usual Bayesian posterior of drawing leaf parameters, is non-zero. Note that this integral arises via the algorithmic implementation that draws T_j by first drawing μ_{-j} and Ψ, and then drawing T_j via GrowFromRoot.

Lastly, consider the transition probability between any two forests, $F^1 = \{T_h^1\}_{1\leq h\leq L}$ and $F^2 = \{T_h^2\}_{1\leq h\leq L}$. Observe that there is at least one way to transition from one forest to another, which is to regrow each tree and replace them one by one. Therefore, we have

$$P(F_2 \mid F_1) \geq \prod_{j=1}^{L} p\left(T_j^2 \mid \{T_h^2\}_{1\leq h<j}, \{T_h^1\}_{j+1\leq h<L}\right) > 0,$$

where the last inequality is from equation (2.5).

In conclusion, the forest process has a finite number of possible states, and the transition probability between any two states is positive. Therefore, by standard results, it is a finite-state Markov chain with a stationary distribution. □

To obtain a prediction from XBART, we take posterior averages as if the sampled trees were draws from a standard Bayesian Monte Carlo algorithm. That is, given I iterations of the algorithm, the final $I - I_0$ samples are used to compute a point-wise average function

evaluation, where $I_0 < I$ denotes the length of the burn-in period. We recommend $I = 40$ and $I_0 = 15$ for routine use. The final estimator is therefore expressible as

$$\bar{f}(\mathbf{X}) = \frac{1}{I - I_0} \sum_{k>I_0}^{I} f^{(k)}(\mathbf{X}).$$

where $f^{(k)}$ denotes a sample of the forest, as in equation 2.4, drawn by algorithm 6. This would correspond to the Bayes optimal estimator under mean squared error estimation loss, if we regard our samples as coming from a proper posterior distribution. As the GrowFromRoot strategy is not a proper full conditional, this estimator must be considered an approximation of some sort. Nonetheless, simulation results strongly suggest that the approximation is adequate. In subsequent sections we also provide some theory suggesting that XBART is a consistent estimator in its own right.

As for quantification of estimation uncertainty, note that with only $I = 40$ sweeps, the XBART posterior would certainly understate the estimation uncertainty even if we had independent Monte Carlo draws from a valid posterior distribution. However, the standard BART MCMC is probably not mixing well in most contexts, either, and yet still provides useful, if approximate, uncertainty quantification. It is noteworthy that experiments with a version of an XBART estimate based on only the final sweep (that is, letting $I - I_0 = 1$) perform worse than XBART with $I - I_0 > 1$, suggesting that the posterior exploration, while partial, is still beneficial. In any event, the next section describes how to combine XBART with standard BART MCMC to get full Bayesian inference that appears to be both faster and more accurate than BART MCMC alone.

2.3 Warm-start BART MCMC

Standard BART MCMC (Chipman et al., 2010) initializes each tree at the root (i.e., a tree only one node) and explores the posterior over trees via a random-walk Metropolis-Hastings

algorithm. This approach works surprisingly well in practice, but it is natural to wonder if it takes unnecessarily long to find favorable regions in tree space. Because XBART provides a fast approximation to the BART posterior, initializing BART MCMC at XBART trees rather than roots is a promising strategy to help speed convergence and also to accelerate posterior exploration by running multiple chains. In fact, we find that this approach yields improved point estimation and posterior credible intervals with substantially higher pointwise frequentist coverage of the mean function, and in a fraction of the total run time. These simulation results are reported in section 4.2.

2.4 Adaptive variable importance weights

Our XBART implementation strikes an intermediate balance between the local BART updates, which randomly consider one variable at a time, and the all-variables Bayes rule described above. Specifically, we consider only $m \leq V$ variables at a time when sampling each splitting rule. Rather than drawing these variables uniformly at random as is done in random forests, we introduce a parameter vector w which denotes the prior probability that a given variable is chosen to be split on, as suggested in Linero (2018). Before sampling each splitting rule, we randomly select m variables (without replacement) with probability proportional to w.

2.5 Computational strategies

In the remainder of this section, I catalogue implementation details that improve the computational efficiency of the algorithm. These implementational details serve to make the algorithm competitive with state-of-the-art supervised learning algorithms, such as XGBoost. These particular strategies, such as variable presorting and careful handling of categorical covariates, are inapplicable in the standard BART MCMC and XBART's ability to incorporate

them is the basis of its improved performance.

2.5.1 Pre-sorting predictor variables

Observe that the XBART split criterion depends on sufficient statistics only, namely the sum of the observations in a node (that is, at a given level of the recursion). An important implication of this, for computation, is that with sorted predictor variables, the various cutpoint integrated likelihoods can be computed rapidly via a single sweep through the data (per variable), taking cumulative sums. Let $\mathbf{O}$ denote the V-by-n array such that o_{vh} denotes the index, in the data, of the observation with the h-th smallest value of the v-th predictor variable x_v. Then, taking the cumulative sums gives

$$s(\leq, v, c) = \sum_{h \leq c} \mathrm{r}_{o_{vh}}$$

and

$$s(>, v, c) = \sum_{h=1}^{n} r_{lh} - s(\leq, v, c).$$

The subscript l on the residual indicates that these evaluations pertain to the update of the lth tree.

The above formulation is useful if the data can be presorted and, furthermore, the sorting can be maintained at all levels of the recursive tree-growing process. To achieve this, we must "sift" each of the variables before passing to the next level of the recursion. Specifically, we form two new index matrices $\mathbf{O}^{\leq}$ and $\mathbf{O}^{>}$ that partition the data according to the selected cutpoint. For the selected split variable v and selected split c, this is automatic: $O_v^{\leq} = O_{v,1:c}$ and $O_v^{>} = O_{v,(c+1):n}$. For the other $V-1$ variables, we sift them by looping through all n available observations, populating $O_q^{\leq}$ and $O_q^{>}$, for $q \neq v$, sequentially, with values o_{qj} according to whether $x_{vo_{qj}} \leq c$ or $x_{vo_{qj}} > c$, for $j = 1, \ldots, n$.

Because the data is processed in sorted order, the ordering will be preserved in each of

the new matrices $\mathbf{O}^{\leq}$ and $\mathbf{O}^{>}$. This strategy was first presented in Mehta et al. (1996) in the context of classification algorithms and has be rediscovered a number of times since then. The pre-sorting and sifting **O** strategy is easy to implement for continuous covariates, but not for categorical covariates due to the possibility of ties in the data. Appendix A describes a special data structure for dealing with ties efficiently.

2.5.2 Adaptive cutpoint grid

Evaluating the integrated likelihood criterion is straightforward, but the summation and normalization required to sample the cutpoints contributes a substantial computational burden itself. Therefore, it is helpful to consider a restricted number of cutpoints C. This can be achieved simply by taking every jth value (starting from the smallest) as an eligible cutpoint with $j = \lfloor \frac{n_b - 2}{C} \rfloor$. As the tree grows deeper, the amount of data that is skipped over diminishes. Eventually, we get $n_b < C$, and each data point defines a unique cutpoint. In this way, the data could, without regularization, be fit perfectly, even though the number of cutpoints at any given level is given an upper limit. As a default, we set the number of cutpoints to $\min(n, 100)$, where n is the sample size of the entire data set.

Our cutpoint subsampling strategy is more straightforward than the elaborate cutpoint subselection search heuristics used by `XGBoost` (Chen and Guestrin, 2016) and `LightGBM` (Ke et al., 2017), which both consider the gradient evaluated at each cutpoint when determining the next split. Our approach does not consider the response information at all, but rather defines a predictor-dependent prior on the response surface. That is, given a design matrix $\mathbf{X}$, sample functions can be drawn from the prior distribution by sampling trees, splitting uniformly at random among the cutpoints defined by the node-specific quantiles, in a sequential fashion.

2.5.3 Variable importance weights

The variable weight parameter w is given a Dirichlet prior with hyper-parameters $\bar{\text{w}}$ that is initialized to all ones. At each iteration of the first sweep through the forest, $\bar{\text{w}}$ is incremented to count the total number of splits across all trees. The split counts are then updated in between each tree sampling/growth step:

$$\bar{\text{w}} \leftarrow \bar{\text{w}} - \bar{\text{w}}_l^{(k-1)} + \bar{\text{w}}_l^{(k)}$$

where $\bar{\text{w}}_l^{(k)}$ denotes the length-V vector recording the number of splits on each variable in tree l at iteration k. The weight parameter is then re-sampled as $\text{w} \sim \text{Dirichlet}(\bar{\text{w}})$, which is a standard practice sampling weights (Blei et al., 2003; Kolar et al., 2017; Yu et al., 2018a, 2017a). Splits that improve the likelihood function will be chosen more often than those that don't. The parameter w is then updated to reflect that, making chosen variables more likely to be considered in subsequent sweeps. In practice, we find it is helpful to use all V variables during an initialization phase, to more rapidly obtain an accurate initial estimate of w.

2.6 Tree-based models for nonlinear regression

This section provides details of XBART in the Gaussian nonlinear regression setting. We derive specific split criteria and sampling strategies for leaf parameters μ and non-tree parameters Ψ. We begin by considering a nonlinear mean regression additive error model

$$y = f(\mathbf{x}) + \epsilon, \tag{2.6}$$

where f is the unknown mean regression function $f(\mathbf{x}) = \mathbb{E}[y \mid \mathbf{x}]$ and $\epsilon \sim N(0, \sigma^2)$. The extra non-tree parameter is residual variance σ^2, which is given a standard inverse-

Gamma(a, b) prior and updated in between each tree update. Reviewing notation, $\mathbf{x} = (\mathbf{x}^{(1)}, \cdots, \mathbf{x}^{(p)})$ is a p dimensional covariate vector and $y \in \mathbb{R}$ is the real response variable. Capital letters represent a vector or matrix of data, $Y = (y_1, \cdots, y_n)$ is a vector of n observations and $\mathbf{X} = (\mathbf{x}_1', \cdots, \mathbf{x}_n)'$ is a $n \times p$ matrix of covariate data. Leaf parameters are given independent and identical Gaussian priors, $\mu \sim N(0, \tau)$. In the notation from above, these modeling choices correspond to hyper-parameter and model parameters $\Phi = (a, b, \tau)$ and $\Psi = (\sigma)$, respectively.

Assuming that observations in the same leaf node share a common mean parameter, the prior predictive distribution — obtained by integrating out the unknown group-specific mean — is simply a mean-zero multivariate Gaussian distribution with covariance matrix $\mathbf{V}$,

$$p(Y \mid \tau, \sigma^2) = \int N(Y \mid \mu, \sigma^2 \mathbf{I}_n) \mathrm{N}(\mu \mid 0, \tau) d\mu = N(0, \mathbf{V}), \tag{2.7}$$

where $\mathrm{N}(Y \mid \mu, \sigma^2 \mathbf{I}_n)$ denotes the density of multivariate Gaussian distribution with mean μ and covariance matrix $\sigma^2 \mathbf{I}_n$, n is number of data observations in the current node. We have

$$\mathbf{V} = \tau \mathrm{J}\mathrm{J}^t + \sigma^2 \mathbf{I}_n, \quad \mathbf{V}^{-1} = \sigma^{-2} \mathbf{I} - \frac{\tau}{\sigma^2(\sigma^2 + \tau n)} \mathrm{J}\mathrm{J}^t,$$

where J is a length n column vector of all ones. Observe that the prior predictive density of $Y \sim \mathrm{N}(0, \mathbf{V})$ is

$$p(Y \mid \tau, \sigma^2) = (2\pi)^{-n/2} \det(\mathbf{V})^{-1/2} \exp\left(-\frac{1}{2} Y^t \mathbf{V}^{-1} Y\right),$$

which can be simplified by a direct application of the matrix inversion lemma to $\mathbf{V}^{-1}$. Applying Sylvester's determinant theorem to $\det \mathbf{V}^{-1}$ yields

$$\det \mathbf{V}^{-1} = \sigma^{-2n} \left(1 - \frac{\tau n}{\sigma^2 + \tau n}\right) = \sigma^{-2n} \left(\frac{\sigma^2}{\sigma^2 + \tau n}\right).$$

Taking logarithms yields a marginal log-likelihood of

$$-\frac{n}{2}\log(2\pi) - n\log(\sigma) + \frac{1}{2}\log\left(\frac{\sigma^2}{\sigma^2+\tau n}\right) - \frac{1}{2}\frac{Y^tY}{\sigma^2} + \frac{1}{2}\frac{\tau}{\sigma^2(\sigma^2+\tau n)}s^2,$$

where we write the sufficient statistics $s \equiv Y^t\mathrm{J} = \sum_i y_i$ so that $Y'\mathrm{J}\mathrm{J}^tY = (\sum_i y_i)^2 = s^2$. This likelihood is applied separately to two child nodes of a single cutpoint (j,k). Because observations in different leaf nodes are independent (conditional on σ^2), the full marginal log-likelihood is given by

$$\begin{aligned}
&\sum_{b=1}^{2}\left\{-\frac{n_b}{2}\log(2\pi) - n_b\log(\sigma) + \frac{1}{2}\log\left(\frac{\sigma^2}{\sigma^2+\tau n_b}\right) - \frac{1}{2}\frac{Y_b^tY_b}{\sigma^2} + \frac{1}{2}\frac{\tau}{\sigma^2(\sigma^2+\tau n_b)}s_b^2\right\} \\
=&-\frac{n}{2}\log(2\pi) - n\log(\sigma) - \frac{1}{2}\frac{Y^tY}{\sigma^2} + \frac{1}{2}\sum_{b=1}^{2}\left\{\log\left(\frac{\sigma^2}{\sigma^2+\tau n_b}\right) + \frac{\tau}{\sigma^2(\sigma^2+\tau n_b)}s_b^2\right\},
\end{aligned}$$

where index b runs over two child nodes and $\sum_{b=1}^{2} n_b = n$. Notice that the first three terms are not functions of the partition (the tree parameter), and so may be ignored, leaving

$$\frac{1}{2}\sum_{b=1}^{2}\left\{\log\left(\frac{\sigma^2}{\sigma^2+\tau n_b}\right) + \frac{\tau}{\sigma^2(\sigma^2+\tau n_b)}s_b^2\right\}$$

as the model-based split criterion, where (n_b, s_b) are functions of the data. Therefore, we define the log-integrated-likelihood

$$\log(m(s)) = \log\left(\frac{\sigma^2}{\sigma^2+\tau n}\right) + \frac{\tau}{\sigma^2(\sigma^2+\tau n)}s^2. \tag{2.8}$$

The logarithm of split criterion (j,k) is

$$\begin{aligned}\log\left(m(s_{jk}^l)m(s_{jk}^r)\right) = &\log\left(\frac{\sigma^2}{\sigma^2+\tau n_{jk}^l}\right) + \frac{\tau}{\sigma^2(\sigma^2+\tau n_{jk}^l)}\left(s_{jk}^l\right)^2 \\ &+ \log\left(\frac{\sigma^2}{\sigma^2+\tau n_{jk}^r}\right) + \frac{\tau}{\sigma^2(\sigma^2+\tau n_{jk}^r)}\left(s_{jk}^r\right)^2,\end{aligned} \tag{2.9}$$

where $n_{jk}^l = |\mathcal{A}_L(j,k)|$ and $n_{jk}^r = |\mathcal{A}_R(j,k)|$ are number of data observations on left or right child node if split at cutpoint (j,k). s_{jk}^l and s_{jk}^r are sufficient statistics

$$s_{jk}^l = \sum_{i:x_i\in\mathcal{A}_L(j,k)} y_i, \quad s_{jk}^r = \sum_{i:x_i\in\mathcal{A}_R(j,k)} y_i.$$

Similarly, the log-probability of no-split is

$$\log\left(|\mathcal{C}|\left(\frac{(1+d)^\beta}{\alpha}-1\right)\right) + \log\left(\frac{\sigma^2}{\sigma^2+\tau n}\right) + \frac{\tau}{\sigma^2(\sigma^2+\tau n)}s^2. \tag{2.10}$$

For notational simplicity, we overload n as the number of data observations in the *current* node and s is sum of all y in the *current* node. It is apparent that $n = n_{jk}^l + n_{jk}^r$ and $s = s_{jk}^l + s_{jk}^r$ for all cutpoints (j,k). Note that the split criterion involves residual standard error σ, meaning that it is adaptively regularizing within the model fitting process.

If the no-split option is selected or stopping conditions are satsified, we label this node as leaf $\mathcal{A}_{lb}$, the b-th leaf of l-th tree. Leaf parameter μ_{lb} associated with leaf $\mathcal{A}_{lb}$ is updated in step 10 of Algorithm 5. We assume a conjugate Gaussian prior $\mu_{lb} \sim N(0,\tau)$, therefore the posterior to sample from is

$$\mu_{lb} \sim N\left(\frac{s_{lb}}{\sigma^2\left(\frac{1}{\tau}+\frac{n_{lb}}{\sigma^2}\right)}, \frac{1}{\frac{1}{\tau}+\frac{n_{lb}}{\sigma^2}}\right), \tag{2.11}$$

where n_{lb} is number of data observations and $s_{lb} = \sum_{y\in\mathcal{A}_{lb}} y$ is the sufficient statistic in the

leaf node corresponding to leaf parameter μ_{lb}.

Next, we describe the model parameter sampling steps in Algorithm 6. The only non-tree model parameter for Gaussian nonlinear regression is the residual variance σ^2, which updates after one draw of a tree in step 10 of Algorithm 6. For σ^2 we assume a standard inverse-Gamma prior, $\sigma^2 \sim$ inverse-Gamma(a, b), and the posterior is

$$\sigma^2 \sim \text{inverse-Gamma}\left(N + a, \tilde{\mathrm{r}}_h^{(iter)t}\tilde{\mathrm{r}}_h^{(iter)} + b\right), \tag{2.12}$$

where $\tilde{\mathrm{r}}_h^{(iter)}$ is the *total* residual after updating the h-th tree in the *iter*-th Monte Carlo iteration, defined as

$$\tilde{\mathrm{r}}_h^{(iter)} \equiv \mathrm{y} - \sum_{h' \leq h} g(\mathbf{X}; T_{h'}, \mu_{h'})^{(iter+1)} - \sum_{h' > h} g(\mathbf{X}; T_{h'}, \mu_{h'})^{(iter)}.$$

Remark The derivations above pertain to growing a single tree by Algorithm 5. Note that in the context of the forest, the data y in the above would instead be the residual r_h.

The default parameters, used in all simulations reported here, are $L = 30$ trees and $\tau = \text{Var}(\mathrm{y})/L$.

2.7 Model implied by the GrowFromRoot algorithm

The XBART cutpoint sampling, while based on Bayes rule, is *myopic* in the sense that it does not consider the entire tree structure when evaluating its (marginal) likelihood. In particular, the recursive structure of a binary tree implies that the data is "reused" at different levels of the tree.

However, interestingly, in the case of a single tree it is possible to show that GrowFromRoot can be interpreted as a proper Bayesian model, as follows. From equation (2.7), the integrated likelihood of a single leaf is Gaussian with mean a vector of zeros and precision

matrix

$$\mathbf{\Omega} := \mathbf{V}^{-1} = \sigma^{-2}\mathbf{I} - \frac{\tau}{\sigma^2(\sigma^2 + \tau n)}\mathrm{J}\mathrm{J}^t,$$

where $\mathbf{I}$ is a $n \times n$ identity matrix and J is a vector of ones with length n. Now, regard the tree growing algorithm as an exhaustive one, always growing maximum depth trees (relative to $\mathbf{X}$): while the tree keeps splitting, the integrated likelihood is a Gaussian likelihood; once a node stops splitting, the likelihood of all the nodes beneath it degenerate to 1.

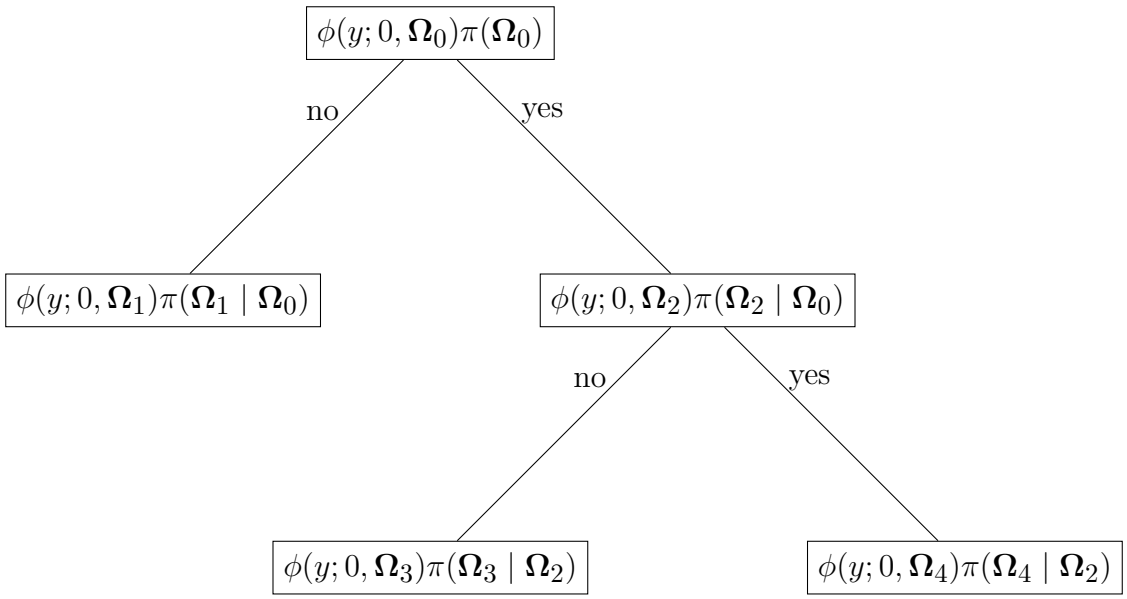
The posterior of a single tree model is

$$\pi_{gfr}(T \mid \mathrm{y}) \propto \prod_{i=0}^{B} \phi(y; 0, \mathbf{\Omega}_i) P(i), \tag{2.13}$$

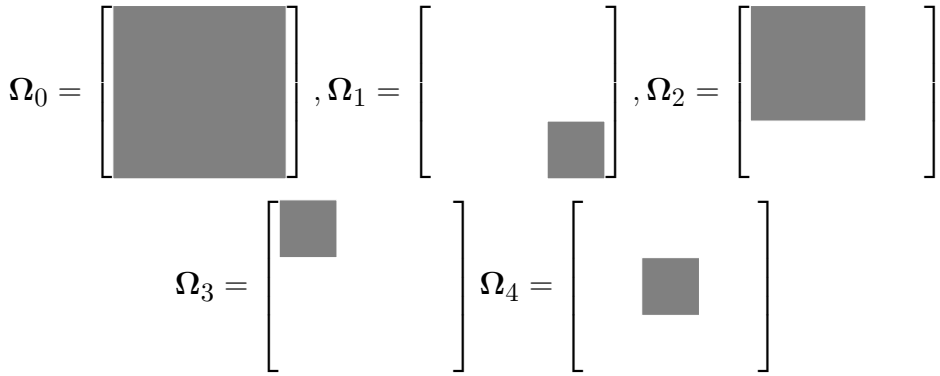
where B is number of all nodes in the tree, and $\phi(y; 0, \mathbf{\Omega})$ is a multivariate Gaussian PDF with precision matrix $\mathbf{\Omega}$ and $P(i) = \alpha(1 + d_i)^{-\beta}$ is the BART prior probability of the i-th node reaching depth d_i. Figure 2.1 illustrates the assignment and structure of precision matrices. All precision matrices have the same dimension n, the total number of observations. Since each non-root node only has a subset of the data, the precision matrix has a block-diagonal structure with a non-zero sub-matrix on the diagonal indicating correlation of data observations in that node and 0 elsewhere (it is always possible to rearrange order of the data to make the precision matrix block-diagonal). The cumulative product of Gaussian kernels in equation (2.13) represents another Gaussian kernel up to a normalizing constant

$$\prod_{i=1}^{B} \phi(y; 0, \mathbf{\Omega}_i) P(i) = \exp\left(\xi_{i=1,\cdots,B} - \xi_B\right) \exp\left(\xi_B - \frac{1}{2} y^t \mathbf{\Omega}^B y\right) \prod_{i=1}^{B} P(i), \tag{2.14}$$

where $\mathbf{\Omega}^B = \sum_{i=1}^{B} \mathbf{\Omega}_i$, the normalizing constants $\xi_B = -\frac{1}{2}\left(N \log(2\pi) - \log|\mathbf{\Omega}^B|\right)$ and $\xi_{i=1,\cdots,B} = \sum_{i=1}^{B} -\frac{1}{2}\left(N \log(2\pi) - \log|\mathbf{\Omega}_i|\right)$. Therefore, we may consider the GrowFrom-



(a) Nodes and corresponding precision matrices.



(b) Precision matrices

Figure 2.1: An illustration of the precision matrices at each node, from root to leaves. Left panel: assignment of precision matrix at each node. Right panel: illustration of precision matrices, where grey block represents non-zero elements and white blocks are 0.

Root likelihood to be the single multivariate Gaussian

$$\phi(y; 0, \mathbf{\Omega}^B) = \exp\left(\xi_B - \frac{1}{2} y^t \mathbf{\Omega}^B y\right), \tag{2.15}$$

in which the data only appear once, and the implied prior of GrowFromRoot (which does not include y), is

$$\exp\left(\xi_{i=1,\cdots,B}\right) \prod_{i=1}^{B} P(i). \tag{2.16}$$

On the other hand, the BART posterior is

$$\pi_{bart}(T \mid \mathrm{y}) \propto \prod_{i \in \mathrm{Leaf}} \phi(y; 0, \mathbf{\Omega}_i) \prod_{i=1}^{B} P(i), \tag{2.17}$$

Here the likelihood is $\prod_{i \in \mathrm{Leaf}} \phi(y; 0, \mathbf{\Omega}_i)$ and the prior is $\prod_{i=1}^{B} P(i)$.

Remark The discussion above is only to show that the GrowFromRoot sampling process can be considered a well defined model, but it is *not* the one that is used to sample leaf parameters. Indeed, the multivariate Gaussian model above corresponds to building up the mean function from a weighted average of node-specific mean vectors. We attempted estimating the mean parameters in this fashion and it was dramatically outperformed by using only the leaf parameters, as in BART. Nonetheless, the GrowFromRoot algorithm appears to produce samples of trees that perform well in conjunction with the leaf-only estimation method of the conditional means.

CHAPTER 3

CONSISTENCY OF REGRESSION TREE

3.1 Overview of theoretical results

Tree-based methods have a substantial, if incomplete, body of theory going back several decades. Gordon and Olshen (1980) analyze the consistency of recursive partitioning irrespective of the specific split criterion; to achieve this they assume that the diameter of leaf node hyperrectangles shrink to zero at a certain rate. Breiman (2001) gives an upper bound of the generalization error of random forest, and Lin and Jeon (2006) show a lower bound of the generalization error of a nonadaptive forest. Biau et al. (2008) and Ishwaran et al. (2008) establish consistency of a simplified random forest model. Scornet et al. (2015) is the first consistency result of the original random forest algorithm, and their theory applies to the consistency of CART directly. Wager and Athey (2018) study the asymptotic sampling distribution of random forest. Zhang et al. (2005) prove consistency and derive the convergence rate for boosting with early stopping, although the result is non-constructive in that their results are not known to apply to any specific stopping-rule. Bartlett and Traskin (2007) establish consistency theory for the adaBoost algorithm.

There has also been a surge of recent theoretical results for BART. Coram et al. (2006) prove consistency for Bayesian histograms of binary regression. Rocková and van der Pas (2017) prove posterior consistency for a variant of the BART prior and Ročková and Saha (2019) study posterior concentration of the exact BART prior. Linero and Yang (2018) establish posterior consistency for a fractional posterior of soft BART (SBART), whose trees have soft decision rules.

In this chapter, I prove the consistency of a single tree of the XBART algorithm for the Gaussian nonlinear regression case. First, we establish the connection of our XBART sampling strategy to the optimization approach in CART by applying the perturb-max

theorem. Having reconciled the sampling versus optimizing distinction, we are then able to adapt the consistency proof for CART to the XBART split criterion. The key step of the proof is to show that variation of the true function is small in each hyper-rectangular cells associated to a leaf node, as the number of data observations grows large enough, either because the diameter of the cell shrinks to zero or because the true function is flat over that region. I follow the proof of Scornet et al. (2015) closely.

3.2 Theory of consistency

Before diving into the theorem and proofs, I again review notations. Suppose $\mathbf{x} \in [0,1]^p$ is a vector of input variables and $y \in \mathbb{R}^1$ is the corresponding outcome variable. Our goal is to estimate the regression function $f(\mathbf{x}) = \mathbb{E}[y \mid \mathbf{x}]$ as $f_n : (0,1)^p \to \mathbb{R}$ based on data $\{(y_1, \mathbf{x}_1), \cdots, (y_n, \mathbf{x}_n)\}$. Let d_n denote maximum depth of a tree.

A key assumption of Scornet et al. (2015) is that the regression function is additive,

Assumption 1 (A1)**.**

$$y = \sum_{j=1}^{p} f_j(x^{(j)}) + \epsilon$$

where $\mathbf{x} = (\mathbf{x}^{(1)}, \cdots, \mathbf{x}^{(p)})$ *is uniformly distributed on* $[0,1]^p$*.* $\epsilon \sim N(0, \sigma^2)$*.*

See the remark following Lemma 2 concerning the possibility of relaxing this strong assumption.

I show consistency for the case of regression with Gaussian noise and focus on a variant of XBART algorithm which only contains a single tree. Our main theorem states that a single XBART regression tree approximates the true underlying mean function in $\mathcal{L}^2$ norm if maximum depth goes to infinity slower than a function of the number of data points.

Theorem 3. *Assume (A1) holds. Let* $n \to \infty$*,* $d_n \to \infty$ *and* $(2^{d_n} - 1)(\log n)^9/n \to 0$*,*

XBART is consistent in the sense that

$$\lim_{n\to\infty} \mathbb{E}[f_n(\mathbf{x}) - f(\mathbf{x})]^2 = 0. \tag{3.1}$$

Both a single-tree XBART and CART learn decision rules by a recursive algorithm, but with a different way of selecting cutpoints. CART optimizes its split criterion while XBART draws cutpoints randomly with probability proportional to the split criterion. However, sampling from a so-called perturb-max model is equivalent to optimizing an objective function with an additional random draw from Gumbel$(0, 1)$ distribution, see Corollary 6.2 from Hazan et al. (2016), restated here for the sake of completeness.

Lemma 1 (Perturb-max theorem)**.** *Suppose there are $|\mathcal{C}|$ finite cutpoint candidates $\{c_{jk}\}$ at a specific node. We are interested in drawing one of them according to probability $P(c_{jk}) = \frac{\exp(l(c_{jk}))}{\sum_{c_{jk}\in\mathcal{C}} \exp(l(c_{jk}))}$. We have*

$$\frac{\exp(l(c_{jk}))}{\sum_{c_{jk}\in\mathcal{C}} \exp(l(c_{jk}))} = P\left(c_{jk} = \arg\max_{c_{jk}\in\mathcal{C}} \{l(c_{jk}) + \gamma_{jk}\}\right) \tag{3.2}$$

where $\{\gamma_{jk}\}$ are independent random draws from a Gumbel$(0, 1)$ distribution with density $p(x) = \exp(-x + exp(-x))$.

The independent random draws $\{\gamma_{jk}\}$ can be treated as known constants if conditioning on a random seed Θ, as in Scornet et al. (2015). That is, Θ is used to sample Gumbel random draws, and we always assume taking the condition of Θ in the following proof. Lemma 1 states that XBART's sampling cutpoint strategy is equivalent to optimizing an objective function. Thus CART and XBART fitting algorithms only differ in the specific form of the split criterion to optimize. Our proof of consistency is based on the work of Scornet et al. (2015), where only Lemma 1 and Lemma 2 involve the specific function form of split criterion. Therefore I only have to check that Lemma 1 and Lemma 2 of Scornet et al. (2015)

are still valid for the XBART split criterion. Recalling equation (2.9), the logarithm of the split criterion c_{jk} is

$$
\begin{aligned}
l(c_{jk}) = \log\left(\frac{\sigma^2}{\sigma^2+\tau n^l_{jk}}\right) &+ \frac{\tau}{\sigma^2\left(\sigma^2+\tau n^l_{jk}\right)}\left(\sum_{i:\mathbf{x}_i\in\mathcal{A}_L(j,k)} y_i\right)^2 \\
&+ \log\left(\frac{\sigma^2}{\sigma^2+\tau n^r_{jk}}\right) + \frac{\tau}{\sigma^2\left(\sigma^2+\tau n^r_{jk}\right)}\left(\sum_{i:\mathbf{x}_i\in\mathcal{A}_R(j,k)} y_i\right)^2 \\
= \frac{\tau}{\sigma^2\left(\sigma^2+\tau n^l_{jk}\right)}&\left(n^l_{jk}\sum_{i:\mathbf{x}_i\in\mathcal{A}_L(j,k)} y_i^2 - (n^l_{jk}-1)\sum_{i:\mathbf{x}_i\in\mathcal{A}_L(j,k)}(y_i-\bar{y}_l)^2\right) \\
&+ \frac{\tau}{\sigma^2\left(\sigma^2+\tau n^r_{jk}\right)}\left(n^r_{jk}\sum_{i:\mathbf{x}_i\in\mathcal{A}_R(j,k)} y_i^2 - (n^r_{jk}-1)\sum_{i:\mathbf{x}_i\in\mathcal{A}_R(j,k)}(y_i-\bar{y}_r)^2\right) \\
&+ \log\left(\frac{\sigma^2}{\sigma^2+\tau n^l_{jk}}\right) + \log\left(\frac{\sigma^2}{\sigma^2+\tau n^r_{jk}}\right),
\end{aligned}
$$

where $\bar{y}_l = \frac{1}{n^l_{jk}}\sum_{i:\mathbf{x}_i\in\mathcal{A}_L(j,k)} y_i$ and $\bar{y}_r = \frac{1}{n^r_{jk}}\sum_{i:\mathbf{x}_i\in\mathcal{A}_R(j,k)} y_i$ are averagea of y in the left and right children respectively. Following Lemma 1, we optimize

$$
c^*_{jk} = \arg\max_{c_{jk}\in\mathcal{C}}\{l(c_{jk}) + \gamma_{jk}\},
$$

where γ_i are random draws from Gumbel$(0,1)$ and can be treated as fixed constant if we condition on random seed Θ. Note that the optimization problem is invariant if the objective function is scaled by a constant n, used here to denote the number of observations in the current node, so that

$$
\arg\max_{c_{jk}\in\mathcal{C}} \frac{l(c_{jk})}{n} + \frac{\gamma_{jk}}{n}.
$$

and our "empirical" split criterion (in the terminology of Scornet et al. (2015)) is defined as

$$
L_n(c_{jk}) = \frac{l(c_{jk})}{n} + \frac{\gamma_x}{n}. \tag{3.3}
$$

Letting $n \to \infty$, our empirical split criterion function $L_n(x)$ converges to the "theoretical" version

$$L^*(j, c_{jk}) = \frac{1}{\sigma^2} P(\mathbf{x}^{(j)} \leq c_{jk}) \left[\mathbb{E}(y \mid \mathbf{x}^{(j)} \leq c_{jk})\right]^2 + \frac{1}{\sigma^2} P(\mathbf{x}^{(j)} > c_{jk}) \left[\mathbb{E}(y \mid \mathbf{x}^{(j)} > c_{jk})\right]^2. \tag{3.4}$$

Importantly, $L^*(j, c_{jk})$ does not rely on the training data because, by the strong law of large numbers, $L_n(c_{jk}) \to L^*(c_{jk})$ almost surely as $n \to \infty$. Again following Scornet et al. (2015), we refer to a tree grown according to the empirical split criterion $L_n(c_{jk})$ or the theoretical criterion $L^*(c_{jk})$ as an empirical tree or theoretical tree, respectively. It worth emphasizing that the theoretical split criterion of XBART and CART are the same up to a multiplicative constant $1/\sigma^2$.

In the rest of the section, we recap the proof of consistency for CART and random forest by Scornet et al. (2015) and verify that all lemmas involving the CART split criterion are also valid for that of XBART, equation (3.3).

More notation is needed for the proof. Write $c = (c^{(1)}, c^{(2)})$ to represent a cutpoint, where $c^{(1)} \in \{1, \cdots, p\}$ indicates cut variables and $c^{(2)} \in [0, 1]$ indicates cut values. Let $\mathcal{A}_n(\mathbf{x}, \Theta)$ denote the leaf node of an *empirical tree* built with random parameter Θ that contains $\mathbf{x}$. Let $\mathcal{A}_k^*(\mathbf{x}, \Theta)$ be a cell of the *theoretical tree* at depth k containing $\mathbf{x}$. Additionally, $\mathcal{A}(\mathbf{x}, \mathbf{c}_k)$ is the node containing $\mathbf{x}$ built with sequence of cuts $\mathbf{c}_k$. This node is reached via a sequence of cuts $\mathbf{c}_k = (c_1, \cdots, c_k)$ and we call $\mathbb{A}_k(\mathbf{x})$ the set of all possible $k \geq 1$ cuts used to create the node containing $\mathbf{x}$. The distance between two cut sequences $\mathbf{c}_k, \mathbf{c}_k' \in \mathbb{A}_k(x)$ is defined as

$$||\mathbf{c}_k - \mathbf{c}_k'||_\infty = \sup_{1 \leq j \leq k} \max\left(\left|c_j^{(1)} - c_j'^{(1)}\right|, \left|c_j^{(2)} - c_j'^{(2)}\right|\right).$$

The distance between a cut $\mathbf{c}_k$ and a set $\mathbb{A} \subset \mathbb{A}_k(\mathbf{x})$ is

$$c_\infty(\mathbf{c}_k, \mathbb{A}) = \inf_{\mathbf{c} \in \mathcal{A}} ||\mathbf{c}_k - \mathbf{c}||_\infty.$$

We define the total variation of the true function f within any leaf node $\mathcal{A}$ as

$$\Delta(f, \mathcal{A}) = \sup_{\mathbf{x}, \mathbf{x}' \in \mathcal{A}} |f(\mathbf{x}) - f(\mathbf{x}')|.$$

The proof of Theorem 3 relies critically on the following proposition:

Proposition 1. *Assume (A1) holds. For all $\rho > 0$ and $\xi > 0$, there exists an $N \in \mathbb{N}^*$ such that, for all $n > N$,*

$$\mathbb{P}\left[\Delta(f, \mathcal{A}_n(\mathbf{x}, \Theta)) \leq \xi\right] \geq 1 - \rho. \tag{3.5}$$

Proposition 1 states that the total variation of the true function f within any leaf node of the empirical tree is small if the number of observations, n, used to fit the tree is large enough. In general, the consistency result controling the behavior of the true function on each of the partitions defined by the leaf nodes, in that either the cell diameter shrinks to zero or else the true function is constant over any non-vanishing cell. The proof of Proposition 1 is based on three lemmas below. Lemma 2 and Lemma 3 are the only two pieces involving the specific functional form of the split criterion in the complete proof of Theorem 3.

Lemma 2. *Assume that (A1) holds. Then for all $x \in (0,1)^p$,*

$$\Delta(f, \mathcal{A}_k^*(\mathbf{x}, \Theta)) \to 0 \qquad \textit{almost surely as } k \to \infty.$$

Lemma 2 shows that as $n \to \infty$ and tree grows deeper, variation of the true function f tends to zero in the leaf node of a *theoretical* tree.

Remark Assumption (A1) is used in the proof of Lemma 2 only. If the true function f is additive, Lemma 2 is valid. However, a weaker replacement of assumption (A1) is to assume Lemma 2 is valid directly. Although this is perhaps less interpretable than an assumption of an additive model, it is also presumably a weaker assumption in that it may be satisfied by non-additive models.

Next, we show that the cuts of an empirical tree will be close to its associated theoretical tree in a certain sense. Suppose the empirical tree has grown following a sequence of cuts $\mathbf{c}_{k-1}$, and consider splitting node $\mathcal{A}(\mathbf{x}, \mathbf{c}_{k-1})$. Let $\mathcal{A}_L(\mathbf{x}, \mathbf{c}_{k-1}) = \mathcal{A}(\mathbf{x}, \mathbf{c}_{k-1}) \cap \{\mathbf{x} : \mathbf{x}^{(c_k^{(1)})} \leq c_k^{(2)}\}$ and $\mathcal{A}_R(\mathbf{x}, \mathbf{c}_{k-1}) = \mathcal{A}(\mathbf{x}, \mathbf{c}_{k-1}) \cap \{\mathbf{x} : \mathbf{x}^{(c_k^{(1)})} > c_k^{(2)}\}$ be left and right child nodes of node $\mathcal{A}(\mathbf{x}, \mathbf{c}_{k-1})$ given cut c_k. We write the split criterion equation (3.3) explicitly for $\mathcal{A}(\mathbf{x}, \mathbf{c}_{k-1})$,

$$
\begin{aligned}
L_{n,k}(\mathbf{x}, \mathbf{c}_k) = \frac{1}{n} \frac{\tau}{\sigma^2 \left(\sigma^2 + \tau \mathrm{N}_n(\mathcal{A}_L(\mathbf{x}, \mathbf{c}_{k-1}))\right)} \Bigg(& \mathrm{N}_n(\mathcal{A}_L(\mathbf{x}, \mathbf{c}_{k-1})) \sum_{i:\mathbf{x}_i \in \mathrm{N}_n(\mathcal{A}_L(\mathbf{x}, \mathbf{c}_{k-1}))} y_i^2 \\
& - (\mathrm{N}_n(\mathcal{A}_L(\mathbf{x}, \mathbf{c}_{k-1})) - 1) \sum_{i:\mathbf{x}_i \in \mathrm{N}_n(\mathcal{A}_L(\mathbf{x}, \mathbf{c}_{k-1}))} (y_i - \bar{y}_{\mathcal{A}_L(\mathbf{x}, \mathbf{c}_{k-1})})^2 \Bigg) \\
+ \frac{1}{n} \frac{\tau}{\sigma^2 \left(\sigma^2 + \tau \mathrm{N}_n(\mathcal{A}_L(\mathbf{x}, \mathbf{c}_{k-1}))\right)} \Bigg(& \mathrm{N}_n(\mathcal{A}_L(\mathbf{x}, \mathbf{c}_{k-1})) \sum_{i:\mathbf{x}_i \in \mathcal{A}_R(\mathbf{x}, \mathbf{c}_{k-1})} y_i^2 \\
& - (\mathrm{N}_n(\mathcal{A}_L(\mathbf{x}, \mathbf{c}_{k-1})) - 1) \sum_{i:\mathbf{x}_i \in \mathcal{A}_R(\mathbf{x}, \mathbf{c}_{k-1})} (y_i - \bar{y}_r)^2 \Bigg) \\
+ \frac{1}{n} \log \left(\frac{\sigma^2}{\sigma^2 + \tau \mathrm{N}_n(\mathcal{A}_L(\mathbf{x}, \mathbf{c}_{k-1}))} \right) & + \frac{1}{n} \log \left(\frac{\sigma^2}{\sigma^2 + \tau \mathrm{N}_n(\mathcal{A}_R(\mathbf{x}, \mathbf{c}_{k-1}))} \right).
\end{aligned}
$$

Lemma 3 below states that $L_{n,k}(\mathbf{x}, \mathbf{c}_k)$ is "stochastically equicontinuous" on $\mathbf{c}_k$ for all $\mathbf{x} \in [0,1]^p$. For all $\xi > 0$ and $\mathbf{x} \in [0,1]^p$, $\mathbb{A}_{k-1}^{\xi}(\mathbf{x}) \subset \mathbb{A}_{k-1}(\mathbf{x})$ denotes the set of all sequences of cuts $\mathbf{c}_{k-1}$ such that the node $\mathcal{A}(\mathbf{x}, \mathbf{c}_{k-1})$ contains a hypercube with edge length ξ. The set $\bar{\mathbb{A}}_k^{\xi}(x) = \{\mathbf{c}_k : \mathbf{c}_{k-1} \in \mathbb{A}_{k-1}^{\xi}(x)\}$ is equipped with norm $||\cdot||_\infty$.

Lemma 3. *Assume that (A1) holds. Fix $x \in [0,1]^p$, $k \in \mathbb{N}^*$ and let $\xi > 0$. Then $L_{n,k}(x, \dot{})$ is*

stochastically equicontinuous on $\bar{\mathbb{A}}_k^{\xi}(x)$*, that is, for all* α*,* $\rho > 0$*, there exist* $\delta > 0$ *such that*

$$\lim_{n\to\infty} \mathbb{P}\left[\sup_{\substack{||\mathbf{c}_k - \mathbf{c}_k'||_\infty \leq \delta \\ \mathbf{c}_k, \mathbf{c}_k' \in \bar{\mathbb{A}}_k^{\xi}(\mathbf{x})}} \left|L_{n,k}(\mathbf{x}, \mathbf{c}_k) - L_{n,k}(\mathbf{x}, \mathbf{c}_k')\right| > \alpha\right] \leq \rho.$$

Lemma 3 is used in the proof of Lemma 4 below.

Lemma 4. *Assume that (A1) holds. Fix* $\xi > 0$*,* $\rho > 0$*, and* $k \in \mathbb{N}^*$*. Then there exists* $N \in \mathbb{N}^*$ *such that for all* $n \geq N$*,*

$$\mathbb{P}\left[c_\infty(\widehat{\mathbf{c}}_{k,n}(\mathbf{x}, \Theta), \mathcal{A}_k^*(\mathbf{x}, \Theta)) \leq \xi\right] \geq 1 - \rho. \tag{3.6}$$

Lemma 4 states that the empirical tree converges to the theoretical tree in probability. The proof of Proposition 1 is the same as in Scornet et al. (2015) and so is omitted here. Only proofs of Lemma 2 and 3 rely on the specific form of split criterion; complete proofs are presented in the Appendix.

Finally, we are equipped to prove Theorem 3. Proposition 1 offers good control of approximation error if the tree is grown by the XBART split criterion. There are two steps for the proof. First, the result is proved for the case of a truncated estimator, which is based on Theorem 10.2 of Györfi et al. (2006), presented as Theorem 4 below. Then, the truncation is released to prove the untruncated case.

The truncation T_{β_n} is defined as

$$\begin{cases} T_{\beta_n}(u) = u & \text{if} \quad |u| \leq \beta_n \\ T_{\beta_n}(u) = \text{sign}(u)\beta_n & \text{if} \quad |u| > \beta_n \end{cases}$$

where $\{\beta_n\}$ is a sequence of positive real numbers. The partition obtained with random

variable Θ and data set $\mathcal{D}_n$ is denoted by $\mathcal{P}_n$. Let $\mathcal{M}_n(\Theta)$ is the set of all functions $m : [0,1]^p \to \mathbb{R}$ which is piecewise constant on each node of the partition $\mathcal{P}_n(\Theta)$.

Theorem 4 (Györfi et al. (2006))**.** *Assume that*

1. $\lim_{n\to\infty} \beta_n = \infty$;
2. $\lim_{n\to\infty} \mathbb{E}\left[\inf_{\substack{m\in\mathcal{M}_n(\Theta)\\ ||m||_\infty \le \beta_n}} \mathbb{E}_X\left[m(\mathbf{x}) - f(\mathbf{x})\right]^2\right] = 0$;
3. *for all truncations at* $L > 0$,

$$\lim_{n\to\infty} \mathbb{E}\left[\sup_{\substack{m\in\mathcal{M}_n(\Theta)\\ ||f||_\infty \le \beta_n}} \left|\frac{1}{a_n}\sum_{i=1}^{n}\left[m(\mathbf{x}_i) - T_L\left(y_i\right)\right]^2 - \mathbb{E}\left[m(\mathbf{x}) - T_L(y)\right]^2\right|\right] = 0.$$

Then

$$\lim_{n\to\infty} \mathbb{E}[T_{\beta_n}\left(f_n(X,\Theta)\right) - f(X)]^2 = 0.$$

It is sufficient to verify the three assumptions of Theorem 4 to show that the truncated estimator is consistent. Intuitively, the first condition says that the truncation is relaxed as $n \to \infty$, and the next two conditions control the approximation error and the estimation error, respectively. For the sake of brevity, we skip the proof. Interested readers may refer to Scornet et al. (2015) for details.

CHAPTER 4

SIMULATION STUDIES OF XBART REGRESSION

4.1 Time-accuracy comparisons to other popular machine learning methods

4.1.1 Synthetic regression data

To demonstrate the performance of XBART, we estimate function evaluations with a hold-out set that is a quarter of the training sample size and judge accuracy according to root mean squared (estimation) error (RMSE). We consider four different challenging functions, f, as defined in Table 4.1. In all cases, $x_j \overset{\text{iid}}{\sim} N(0,1)$ for $j = 1, \ldots, d = 30$. The data is generated according to the additive error mode, with $\epsilon_i \overset{\text{iid}}{\sim} N(0,1)$. We consider $\sigma = \kappa \text{Var}(f)$ for $\kappa \in \{1, 10\}$.

Table 4.1: Four true f functions

Name	Function
Linear	$\mathbf{x}^t\gamma;\ \gamma_j = -2 + \frac{4(j-1)}{d-1}$
Single index	$10\sqrt{a} + \sin(5a);\ a = \sum_{j=1}^{10}(x_j - \gamma_j)^2;\ \gamma_j = -1.5 + \frac{j-1}{3}.$
Trig + poly	$5\sin(3x_1) + 2x_2^2 + 3x_3x_4$
Max	$\max(x_1, x_2, x_3)$

We compare to leading machine learning algorithms: random forests, gradient boosting machines, neural networks, and BART. All implementations had an `R` interface and were the current fastest implementations to our knowledge: `ranger` (Wright and Ziegler, 2015), `xgboost` (Chen and Guestrin, 2016), and `Keras` (Chollet et al., 2015), `dbarts` respectively. For `Keras` we used a single architecture but varied the number of training epochs depending on the noise level of the problem. For `xgboost` we consider two specifications, one using the software defaults and another determined by a 5-fold cross-validated grid optimization (see Table 4.2); a reduced grid of parameter values was used at sample sizes $n > 10,000$.

Comparison with `ranger` and `dbarts` are shown in supplementary material.

Table 4.2: Hyperparameter Grid for XGBoost

Parameter name	$N = 10$K	$N > 10$K
`eta`	$\{0.1, 0.3\}$	$\{0.1, 0.3\}$
`max_depth`	$\{4, 8, 12\}$	$\{4, 12\}$
`colsample_bytree`	$\{0.7, 1\}$	$\{0.7, 1\}$
`min_child_weight`	$\{1, 10, 15\}$	10
`subsample`	0.8	0.8
`gamma`	0.1	0.1

The software used is `R` version 3.4.4 with XGBoost 0.71.2, `dbarts` version 0.9.1, `ranger` 0.10.1 and `keras` 2.2.0. The default hyperparameter settings for XGBoost are `eta` = 0.3, `colsample_bytree` = 1, `min_child_weight` = 1 and `max_depth` = 6. Ranger was fit with `num.trees` = 500 and `mtry` = $5 \approx \sqrt{d}$. BART, with the package `dbarts`, was fit with the defaults of `ntrees` = 200, `alpha` = 0.95, `beta` = 2, with a burn-in of 5,000 samples (`nskip` = 5000) and 2,000 retrained posterior samples (`ndpost` = 2000).

The default `dbarts` algorithm uses an evenly spaced grid of 100 cutpoint candidates along the observed range of each variable (`numcuts` = 100, `usequants = FALSE`). For `Keras` we build a network with two fully connected hidden layers (15 nodes each) using ReLU activation function, ℓ_1 regularization at 0.01, and with 50/20 epochs depending on the signal to noise ratio.

4.1.2 Results

The performance of the new XBART algorithm was excellent, showing superior speed and performance relative to all the considered alternatives on virtually every data generating process. The full results, averaged across five Monte Carlo replications, are reported in Table 4.3. Neural networks perform as well as XBART in the low noise settings under the Max and Linear functions. Unsurprisingly, neural networks outperform XBART under the linear function with low noise. Across all data generating processes and sample sizes,

XBART was 31% more accurate than the cross-validated XGBoost method and typically faster. Specifically, the supplement examines the empirical examples given in Chipman et al. (2010).

The XBART method was slower than the untuned default XGBoost method but was 350% more accurate. This pattern points to one of the main benefits of the proposed method, which is that it has excellent performance using the same hyperparameter settings across all data generating processes. Importantly, these default hyperparameter settings were decided on the basis of prior elicitation experiments using different true functions than were used in the reported simulations. While XGBoost is quite fast, the tuning processes are left to the user and can increase the total computational burden by orders of magnitude.

Random forests and BART were prohibitively slow at larger sample sizes. However, at $n = 10,000$ several notable patterns did emerge; see the supplementary material for full details. First was that BART and XBART typically gave very similar results, as would be expected. BART performed slightly better in the low noise setting and quite a bit worse in the high noise setting (likely due to inadequate burn-in period). Similarly, random forests do well in higher noise settings, while XGBoost and neural networks perform better in lower noise settings.

4.2 Warm-start BART MCMC

In this section, we demonstrate the advantage of initializing BART MCMC at XBART draws. The data generating process is the same as section 4.1.1, and the data size is fixed at 10,000 while noise level κ varies. We fit 40 XBART forests, the first 15 are thrown out as burn-in draws, and 25 forest draws are retained. BART was fit with a burn-in of 1,000 samples, and 2,500 retrained posterior samples. For the warm-start BART, 25 *independent* BART MCMC chains were initialized at the 25 forest draws obtained from XBART and each was run for 100 iterations with no burn-in. Note that the total number of posterior draws

Table 4.3: Root mean squared error (RMSE) of each method. Column XGBoost +CV is result of XGBoost with tuning parameter by cross validation and column NN is result of neural networks. The number in parenthesis is running time in seconds. First column is number of data observations (in thousands).

$\kappa = 1$

n	XBART	XGBoost +CV	XGBoost	NN
		Linear		
10k	1.74 (20)	2.63 (64)	3.23 (0)	1.39 (26)
50k	1.04 (180)	1.99 (142)	2.56 (4)	0.66 (28)
250k	0.67 (1774)	1.50 (1399)	2.00 (55)	0.28 (40)
		Max		
10k	0.39 (16)	0.42 (62)	0.79 (0)	0.40 (30)
50k	0.25 (134)	0.29 (140)	0.58 (4)	0.20 (32)
250k	0.14 (1188)	0.21 (1554)	0.41 (60)	0.16 (44)
		Single Index		
10k	2.27 (17)	2.65 (61)	3.65 (0)	2.76 (28)
50k	1.54 (153)	1.61 (141)	2.81 (4)	1.93 (31)
250k	1.14 (1484)	1.18 (1424)	2.16 (55)	1.67 (41)
		Trig + Poly		
10k	1.31 (17)	2.08 (61)	2.70 (0)	3.96 (26)
50k	0.74 (147)	1.29 (141)	1.67 (4)	3.33 (29)
250k	0.45 (1324)	0.82 (1474)	1.11 (59)	2.56 (41)

$\kappa = 10$

n	XBART	XGBoost +CV	XGBoost	NN
		Linear		
10k	5.07 (16)	8.04 (61)	21.25 (0)	7.39 (12)
50k	3.16 (135)	5.47 (140)	16.17 (4)	3.62 (14)
250k	2.03 (1228)	3.15 (1473)	11.49 (54)	1.89 (19)
		Max		
10k	1.94 (16)	2.76 (60)	7.18 (0)	2.98 (15)
50k	1.22 (133)	1.85 (139)	5.49 (4)	1.63 (16)
250k	0.75 (1196)	1.05 (1485)	3.85 (54)	0.85 (22)
		Single Index		
10k	7.13 (16)	10.61 (61)	28.68 (0)	9.43 (14)
50k	4.51 (133)	6.91 (139)	21.18 (4)	6.42 (16)
250k	3.06 (1214)	4.10 (1547)	14.82 (54)	4.72 (21)
		Trig + Poly		
10k	4.94 (16)	7.16 (61)	17.97 (0)	8.20 (13)
50k	3.01 (132)	4.92 (139)	13.30 (4)	5.53 (14)
250k	1.87 (1216)	3.17 (1462)	9.37 (49)	4.13 (20)

is 2,500, the same as the number of posterior draws by BART. We repeat drawing synthetic data and computing intervals 100 times, all measurement below were taken average with respect to those 100 replications.

Table 4.4 shows the credible interval coverage, length, RMSE of the point estimate, and running time of the three approaches. The running time for warm-start BART is reported as time in seconds for a single *indepedent* BART MCMC, while the number in parenthesis is the running time of the entire warm-start BART fitting process, including XBART fit and assuming all 25 indepedent warm-start BART MCMC were fitted *sequentially* rather than in parallel. In other words, the number in parentheses is the most conservative estimation of the total running time, because the 25 independent BART chains can be trivially parallelized to achieve much lower total running time. Indeed, with 25 processors, the run time would be the XBART run time plus the warm-start run time (not in parentheses).

The warm-start BART boasts a substantial advantage in terms of credible interval coverage and root mean squared error. In all cases, warm-start BART has the best coverage and RMSE among all three approaches and is still faster than BART under the most conservative estimation of running time. Especially when the true function is linear, warm-start initialization helps BART get a considerable improvement in estimation, which may indicated inadequate chain length of BART (that is, poor mixing).

Table 4.4: Coverage and length of credible interval of f at 95% level for warm-start BART MCMC. The table also shows running time (in seconds) and root mean squared error (RMSE) of all approaches.

$\kappa = 1$

		XBART	BART	Warm-start BART
Max	coverage	0.86	0.78	0.95
	interval length	0.36	0.35	0.46
	running time	1.57	44.41	1.21 (31.82)
	RMSE	0.11	0.14	0.11
Trig + Poly	coverage	0.90	0.74	0.96
	interval length	3.61	2.89	4.23
	running time	4.68	92.75	3.02 (80.18)
	RMSE	1.03	1.27	1.01
Single Index	coverage	0.77	0.73	0.87
	interval length	4.84	4.62	5.88
	running time	5.10	102.87	2.97 (79.35)
	RMSE	1.94	2.08	1.92
Linear	coverage	0.78	0.77	0.99
	interval length	7.82	6.14	9.92
	running time	5.61	131.17	3.85 (101.86)
	RMSE	3.11	2.51	1.81

$\kappa = 2$

		XBART	BART	Warm-start BART
Max	coverage	0.88	0.84	0.97
	interval length	0.58	0.64	0.76
	running time	1.35	40.23	1.22 (31.85)
	RMSE	0.17	0.22	0.17
Trig + Poly	coverage	0.90	0.82	0.96
	interval length	5.62	5.06	6.86
	running time	3.68	86.81	2.90 (74.17)
	RMSE	1.65	1.87	1.60
Single Index	coverage	0.81	0.83	0.91
	interval length	6.81	7.67	8.49
	running time	3.92	90.70	2.81 (74.0490)
	RMSE	2.51	2.73	2.47
Linear	coverage	0.50	0.83	0.98
	interval length	6.53	8.82	11.84
	running time	3.61	109.43	3.33 (86.86)
	RMSE	4.74	4.13	2.53

CHAPTER 5

XBART CLASSIFICATION

5.1 Log-Logit multinomial classification

The original Bayesian additive regression trees (Chipman et al. (2013)) is limited to Gaussian models. Chipman et al. (2013) introduce binary classification BART using a probit link function and Albert and Chib (1993) data augmentation strategies. Murray (2017) develops new data augmentation strategies, prior distributions and corresponding efficient MCMC samplers for log-linear BART. In this chapter, we show the similar data augmentation strategies for multinomial classification apply to XBART as well. Note that the integrated likelihood (2.1) does not assume a specific form of the likelihood, thus the extension is not limited to multinomial classification. Interested readers may extend XBART to other models such as probit, or Poisson count model as in Murray (2017) similarly.

For easier comparison with previous literature, I follow notations of Murray (2017) closely. Consider a classification problem of C categories. Suppose $\mathbf{x}_i = (x_{i1}, \cdots, x_{ip})$ represents a vector of covariate variables, which are observed n_i times in the data set. The response y_{ij} is the number of observations with covariate $\mathbf{x}_i$ in category j, where $1 \leq i \leq n$ and $1 \leq j \leq C$, then $\sum_{j=1}^{C} y_{ij} = n_i$. The probability of response of x_i belongs to category j is

$$\pi_j(\mathbf{x}_i) = \frac{f^{(j)}(\mathbf{x}_i)}{\sum_{h=1}^{C} f^{(h)}(\mathbf{x}_i)}. \tag{5.1}$$

We assume a sum of trees form

$$\log\left(f^{(j)}(\mathbf{x})\right) = \sum_{l=1}^{L} g\left(\mathbf{x}, T_l^{(j)}, \mu_l^{(j)}\right),$$

which leads to a multinomial logistic trees model:

$$\pi_j(\mathbf{x}_i) = \frac{\exp\left[\sum_{l=1}^{L} g\left(\mathbf{x}, T_l^{(j)}, \mu_l^{(j)}\right)\right]}{\sum_{h=1}^{C} \exp\left[\sum_{l=1}^{L} g\left(\mathbf{x}, T_l^{(h)}, \mu_l^{(h)}\right)\right]}. \tag{5.2}$$

Taking exponents of a tree $g(\mathbf{x}, T_l, \mu_l)$ only affects leaf parameter μ_l but not tree structure T_l. Let $\lambda_{lt} = \exp(\mu_{lt})$ and $\Lambda_l = (\lambda_{l1}, \cdots, \lambda_{lb_l})'$, a sum of trees becomes a product of trees as

$$f(\mathbf{x}) = \exp\left[\sum_{l=1}^{L} g(\mathbf{x}, T_l, \mu_l)\right] = \prod_{l=1}^{L} g(\mathbf{x}, T_l, \Lambda_l), \tag{5.3}$$

where $g(\mathbf{x}, T_l, \Lambda_l) = \lambda_{ht}$ if $\mathbf{x} \in \mathcal{A}_{ht}$ for $1 \le t \le b_h$. Let θ collects all additional parameters or latent variables, the corresponding integrated likelihood is

$$L(T_l; T_{(l)}, \Lambda_{(l)}, \theta, y) = \int L(T_l, \Lambda_l; T_{(l)}, \Lambda_{(l)}, \theta, y) p(\Lambda_l) d\Lambda_l. \tag{5.4}$$

It is straightforward to calculate the integrated likelihood in Gaussian case since the conjugate prior. However there is no simple conjugate prior for general log-linear model. We follow Murray (2017), rearrange terms of the full likelihood $L(T_l, \Lambda_l; T_{(l)}, \Lambda_{(l)}, \theta, y)$ as

$$\begin{aligned}
L(T_l, \Lambda_l; T_{(l)}, \Lambda_{(l)}, y) &= \prod_{i=1}^{n} w_i f(\mathbf{x}_i)^{u_i} \exp[v_i f(\mathbf{x}_i)] \\
&= \prod_{i=1}^{n} w_i [f_{(l)}(\mathbf{x}_i) g(\mathbf{x}_i, T_l, \Lambda_l)]^{u_i} \exp[v_i f_{(l)}(\mathbf{x}_i) g(\mathbf{x}_i, T_l, \Lambda_l)] \\
&= \prod_{t=1}^{b_l} \prod_{1:\mathbf{x}_i \in \mathcal{A}_{lt}} w_i [f_{(l)}(\mathbf{x}_i) \lambda_{lt}]^{u_i} \exp[v_i f_{(l)}(\mathbf{x}_i) \lambda_{lt}] \\
&= c_l \prod_{t=1}^{b_l} \lambda_{lt}^{r_{lt}} \exp[-s_{lt} \lambda_{lt}]
\end{aligned} \tag{5.5}$$

where the outer product runs over all leaf nodes of the tree T_l and inner product runs over

all observations inside one leaf node, and

$$c_l = \prod_{i=1}^{n} w_i f_{(l)}(\mathbf{x}_i)^{u_i}, \quad r_{lt} = \sum_{i:\mathbf{x}_i \in \mathcal{A}_{lt}} u_i, \quad s_{lt} = \sum_{i:\mathbf{x}_i \in \mathcal{A}_{lt}} f_{(l)}(\mathbf{x}_i) v_i. \tag{5.6}$$

with r_{lt} and s_{lt} working as sufficient statistics. In order to derive split criterion for XBART, it is still necessary to calculate the integrated likelihood of (5.4). Murray (2017) propose a symmetric, conditionally conjugate prior for the likelihood has the form of (5.5). I review it briefly in the following text.

We first assume the symmetric prior on sum of trees is approximately normal $\log[f(\mathbf{x})] \overset{\text{approx}}{\sim} N(0, a_0^2)$. The hyper parameter a_0 plays the same role as σ_μ^2 in the original BART prior. Therefore for the independent prior on each leaf parameter λ_{lt} should have $E(\log[\lambda_{lt}]) = 0$ and $\text{Var}(\log[\lambda_{lt}]) = a_0^2/m$. Murray (2017) introduce a mixture of generalized inverse Gaussian (GIG) distribution

$$p_\lambda(\lambda_{lt} \mid c, d) = \frac{1}{2} p_{\text{GIG}}(\lambda_{lt} \mid -c, 2d, 0) + \frac{1}{2} p_{\text{GIG}}(\lambda_{lt} \mid c, 0, 2d), \tag{5.7}$$

where c and d are hyper-parameters determined by a_0. If a_0^2/m is small, let $c \approx m/a_0^2$ and $d \approx m/a_0^2$ can guarantee $\text{Var}(lambda) \approx a_0^2/m$ as desired. The GIG distribution has density

$$p_{\text{GIG}}(\lambda \mid \eta, \chi, \psi) = \frac{\lambda^{\eta-1} \exp\left[-\frac{1}{2}(\chi/\lambda + \psi\lambda)\right]}{Z(\eta, \chi, \psi)} \tag{5.8}$$

with normalizing constants

$$Z(\eta, \chi, \psi) = \begin{cases} \Gamma(\eta)\left(\frac{2}{\psi}\right)^{\eta} & \text{if } \eta > 0, \chi = 0, \psi > 0 \\ Gamma(-\eta)\left(\frac{2}{\chi}\right)^{-\eta} & \text{if } \eta < 0, \chi > 0, \psi = 0 \\ \frac{2K_\eta(\sqrt{\psi\chi})}{(\psi/\chi)^{\eta/2}} & \text{if } \chi > 0, \psi > 0 \end{cases} \tag{5.9}$$

where $K_\eta(x)$ is the modified Bessel function of the second kind.

Next we show the data augmentation strategies for multinomial likelihood of the form

$$p_{MN}(y_i) = \binom{n_i}{y_{i1}y_{i2}\cdots y_{ic}} \frac{\prod_{j=1}^{c} f^{(j)}(x_i)^{y_{ij}}}{[\sum_{l=1}^{c} f^{(l)}(\mathbf{x}_i)]^{n_i}}. \tag{5.10}$$

We take the data augmentation scheme of Murray (2017), assume a new latent variable ϕ_i and define the joint likelihood of (y_i, ϕ_i) as

$$\begin{aligned} p_{MN}(y_i, \phi_i) &= \binom{n_i}{y_{i1}y_{i2}\cdots y_{ic}} \left(\prod_{j=1}^{c} f^{(j)}(\mathbf{x}_i)^{y_{ij}} \right) \frac{\phi_i^{n_i-1}}{\Gamma(n_i)} \exp\left[-\phi_i \sum_{j=1}^{c} f^{(j)}(\mathbf{x}_i) \right] \\ &= \binom{n_i}{y_{i1}y_{i2}\cdots y_{ic}} \frac{\phi^{n_i-1}}{\Gamma(n_i)} \prod_{j=1}^{c} f^{(j)}(\mathbf{x}_i)^{y_{ij}} \exp\left[-\phi_i f^{(j)}(\mathbf{x}_i) \right], \end{aligned} \tag{5.11}$$

where the likelihood of y_i has form of (5.5) if conditional on ϕ_i.

For any pair outcome categories $j \neq j'$, the log odds in favor of j' are $\log[f^{(j')}(\mathbf{x}_i)] - \log[f^{j}(\mathbf{x}_i)]$ and each $f^{(l)}(\mathbf{x})$ has an independent log-linear BART prior. Further we assume each $f^{(l)}(\mathbf{x})$ uses the same number of trees m and prior parameter a_0. The induced prior on the log odds is approximately $N(0, 2a_0^2)$, therefore a_0 is chosen to reflect prior beliefs of the log odds. Murray (2017) recommends default choice $a_0 = 3.5/\sqrt{2}$.

We fit C separate forests for each category, where the fitting algorithm of each forest follows the general framework of Algorithm 5 and 6, despite details of sampling leaf parameters and latent variables. Due to the transformation of equation (5.3), we present the final prediction as product rather than sum of trees, as a result the h-th tree for category j is defined accordingly as

$$\mathrm{r}_h^{(iter+1),(j)} \equiv \frac{\prod_{h'=1}^{L} f^{(j)}(\mathbf{X}; T_{h'}, \mu_{h'})^{(iter)}}{\prod_{h'<h} f^{(j)}(\mathbf{X}; T_{h'}, \mu_{h'})^{(iter+1)} \times \prod_{h'>h} f^{(j)}(\mathbf{X}; T_{h'}, \mu_{h'})^{(iter)}},$$

while the total residual is taken with respect to all trees

$$\tilde{\mathrm{r}}_h^{(iter+1),(j)} \equiv \frac{\mathrm{r}_h^{(iter+1),(j)}}{f^{(j)}(\mathbf{X}; T_h, \mu_h)^{(iter+1)}}.$$

Thus step 10 of Algorithm 5 that drawing leaf parameters based on sufficient statistics c_l and s_{lt} is

- For $i \leq j \leq c$, draw parameters of $f^{(j)}$ independently with

$$r_{lt} = \sum_{i:\mathbf{x}_i \in A_{lt}^{(j)}} y_{ij}, \quad s_{lt} = \sum_{i:\mathbf{x}_i \in A_{lt}^{(j)}} \phi_i f_{(l)}^{(j)}(\mathbf{x}_i), \tag{5.12}$$

  where $f_{(l)}^{(j)}(\mathbf{x}_i) = \prod_{l \neq h} g\left(\mathbf{x}, T_l^{(j)}, \Lambda_l^{(j)}\right)$ is the partial residual for the l-th tree.

Furthermore, the latent variables $\phi_i, 1 \leq i \leq n$ are updated after fitting a tree at step 9 of Algorithm 6 by

- For $1 \leq i \leq n$, sample $\phi_i \sim \text{Gamma}\left(1, \sum_{j=1}^{c} f^{(j)}(\mathbf{x}_i)\right)$.

There is only one augmented latent variable per observation, regardless of number of categories, and they are sampled from standard Gamma distribution.

XBART classification is still working in progress. There are a few questions that remained unsolved. First, the data augmentation strategy described above does not include a parameter that plays the role of "residual variance", similar as σ^2 in the regression case. The residual variance of regression case appears in the split criterion thus help stop splitting when the information is vague (i.e. residual variance is high). A potential solution is a new hierarchical prior over τ. Second, although the extra computational burden induced by latent variables is often light, it could be a problem if the number of observations is extremely large. In that case, stop sampling some latent variables with tiny value and fix them at zero might speed up sampling without loose to much accuracy. Lastly, the current algorithm

assumes same weight on all observations, while putting more weights on data that are hard to predict (similar to AdaBoost) might improve the overall performance.

5.2 Simulation studies

We illustrate the performance of XBART by simulation studies. We consider a synthetic data with 5 categories, the outcome is drawn from multinomial distribution with probability of each category proportional to the following functions,

$$\begin{aligned} p_1 &\propto \exp\left(2 \times |2 \times x_1 - x_2|\right) \\ p_2 &\propto \exp\left(1\right) \\ p_3 &\propto \exp\left(3 \times x_3^2\right) \\ p_4 &\propto \exp\left(x_3 \times |x_4|\right) \\ p_5 &\propto \exp\left(x_5^2 + x_3\right) \end{aligned} \tag{5.13}$$

where the covariate variables $\mathbf{x} = (x_1, x_2, \cdots, x_p)$, the number of training observations n and number of variables $p \geq 5$ vary. When $p > 5$, the extra variables are non-informative for the outcome category.

We compare to random forests (`ranger`,Wright and Ziegler (2015)) and XGboost (`xgboost`, Chen and Guestrin (2016)). Similar to chapter 4, we use `R` version 3.4.4 with `xgboost` 0.71.2, `ranger` version 0.10.1. The default hyperparameters for `xgboost` are `eta`=`0.3`, `colsample_bytree = 1`, `min_child_weight = 1`, `max_depth = 6` and . For `ranger`, we set `num.trees = 500` and `mtry` $= \sqrt{p}$.

Table 5.1 reports the prediction accuracy on testing sets and running time for various p and n. Overall, this 5 categories data generating process is hard and all three methods obtain accuracy less than 0.6. Across all sample sizes, XBART is about similar accurate as XGboost and usually faster. Unsurprisingly, random forest underperform as p increases

Table 5.1: Prediction accuracy on synthetic testing set and running time of each method for multinomial classification.

p	n	XBART		Random Forests		XGboost	
		Accuracy	Time	Accuracy	Time	Accuracy	Time
5	100	0.457	0.189	0.476	0.098	0.485	0.609
5	500	0.503	0.343	0.504	0.289	0.514	1.735
5	5000	0.530	1.056	0.525	4.059	0.532	6.609
20	100	0.466	0.512	0.466	0.108	0.464	0.761
20	500	0.503	0.849	0.493	0.363	0.501	1.789
20	5000	0.526	2.458	0.513	5.575	0.529	9.276
100	100	0.468	2.548	0.472	0.155	0.459	2.840
100	500	0.498	3.888	0.474	0.592	0.494	5.089
100	5000	0.525	9.738	0.485	12.852	0.523	28.016

since less chance drawing informative variables.

5.3 Empirical studies

This section compares multinomial classification XBART with several popular classification models on 20 datasets from the UCI repository. We process similarly as Fernández-Delgado et al. (2014) and Murray (2017). The purpose of this study is not to show that multinomial XBART outperforms all methods on all datasets, but it can give reasonable results on various real datasets.

The alternative methods include random forests, gradient boosting trees, penalized multinomial probit regression, support vector machine with radial basis function, and neural nets. All methods use default parameter grid for 10-fold cross validation in the R package `caret` (Kuhn et al., 2008).

We choose datasets that have 3-6 categories, 100-3000 observations. Each dataset is split into 80% training and 20% testing. We perform 10-fold cross validation for the methods using CV on the training set and evaluate performance on testing set. We repeat the random partition 50 times and generate 50 out-of-sample prediction accuracy.

Table 5.2: Results of classification studies on UCI machine learning repository. Asterisks denote best performing methods. Entries in italic are statistically significantly different from XBART while the gray ones are significantly worse than XBART.

	rf	gbm	mno	svm	nnet	xbart
balance-scale	*0.848 (0.023)*	*0.925 (0.010)*	*0.897 (0.021)*	*0.909 (0.025)*	*0.961 (0.019)**	0.911 (0.012)
car	*0.983 (0.006)**	*0.979 (0.008)*	*0.834 (0.019)*	*0.774 (0.033)*	0.947 (0.015)	0.938 (0.015)
cardiotocography-3clases	0.937 (0.009)	*0.949 (0.009)**	*0.894 (0.011)*	*0.911 (0.011)*	*0.909 (0.013)*	0.931 (0.011)
contrac	*0.546 (0.024)*	*0.557 (0.023)*	*0.516 (0.028)*	*0.551 (0.024)*	*0.556 (0.028)**	0.409 (0.020)
dermatology	0.970 (0.016)	0.972 (0.020)*	0.968 (0.020)	*0.759 (0.024)*	0.970 (0.022)	0.976 (0.018)*
glass	*0.798* (0.062)*	*0.771 (0.055)*	*0.622 (0.066)*	*0.679 (0.054)*	*0.673 (0.064)*	0.702 (0.076)
heart-cleveland	*0.578 (0.033)*	0.586 (0.039)	0.587 (0.039)	0.620 (0.038)*	0.603 (0.052)	0.583 (0.034)
heart-va	0.357 (0.071)*	0.320 (0.067)	0.349 (0.069)	0.315 (0.069)	0.302 (0.08)	0.308 (0.064)
iris	*0.948 (0.034)*	0.945 (0.034)	0.965 (0.029)*	0.947 (0.034)	0.954 (0.045)	0.954 (0.033)
lymphography	0.866 (0.063)*	0.853 (0.057)	0.821 (0.069)	0.850 (0.062)	0.818 (0.077)	0.835 (0.058)
pittsburg-bridges-MATERIAL	*0.840 (0.058)*	0.844 (0.049)	0.834 (0.061)	0.860 (0.048)*	*0.824 (0.066)*	0.849 (0.046)
pittsburg-bridges-REL-L	0.725 (0.084)*	0.681 (0.093)	0.650 (0.087)	0.692 (0.082)	0.659 (0.091)	0.680 (0.083)
pittsburg-bridges-SPAN	0.637 (0.098)	0.648 (0.101)	0.675 (0.100)	0.681 (0.099)*	0.647 (0.102)	0.631 (0.105)
pittsburg-bridges-TYPE	0.609 (0.088)*	0.581 (0.089)	0.549 (0.089)	0.540 (0.075)	0.565 (0.092)	0.585 (0.093)
seeds	0.940 (0.030)	0.940 (0.031)	0.948 (0.035)*	*0.929 (0.029)*	0.944 (0.034)	0.945 (0.042)
synthetic-control	0.984 (0.012)	*0.971 (0.015)*	0.984 (0.012)	*0.716 (0.024)*	0.987 (0.011)*	0.983 (0.017)
teaching	*0.622 (0.085)**	0.557 (0.086)	0.526 (0.072)	0.547 (0.073)	0.525 (0.087)	0.491 (0.086)
vertebral-column-3clases	0.847 (0.038)	0.829 (0.038)	0.861 (0.033)*	0.839 (0.042)	0.859 (0.033)	0.842 (0.039)
wine-quality-red	*0.702 (0.021)**	*0.631 (0.026)*	0.597 (0.024)	*0.576 (0.026)*	0.597 (0.025)	0.613 (0.022)
wine	0.985 (0.018)*	0.979 (0.021)	0.979 (0.026)	0.980 (0.025)	0.977 (0.027)	0.969 (0.032)

Table 5.2 illustrates the average and standard deviation of accuracy on the testing datasets. We test the null hypothesis of no different between XBART and each method by a paired Wilcoxon test. Italics entries are significantly different than XBART at 0.05 level and gray entries are statistically different and worse than XBART. Random forests and XBART have statistically different out-of-sample accuracy on 9 datasets, where random forest is better in 5 and worse in 4. None of the methods dominate others on all datasets. In conclusion, XBART has competitive performance comparing with other popular alternatives in machine learning toolbox.

CHAPTER 6

DISCUSSION

In this dissertation, I have introduced a novel tree-based ensemble framework, XBART, for supervised learning, that has a wide range of applicability. I demonstrated the advantage of the algorithm in simulation studies: XBART has state-of-the-art prediction accuracy with computational demands that are competitive with alternatives. While this paper has focused on Gaussian nonlinear regression, the proposed algorithm extends to other settings straightforwardly, such as logit multi-class classification, binary probit classification, Poisson regression, or classification by a central-limit approximation; these extensions will be described in separate forthcoming manuscripts.

While I proved the consistency of a single XBART tree in this paper, an open question is whether the forest is consistent or not. One proof strategy would be to make a minor modification of the model wherein an initial tree is fit to the data, and then a forest is fit to the residual. By consistency of the initial tree, the residual will eventually converge to pure noise, and a proof would need to show that the resulting Markov Chain had expectation of zero in the large data limit.

Another interesting research topic is the connection between XBART and full Bayesian methods. In this dissertation, I show that the warm-start provided by XBART helps the MCMC algorithm converge faster and attain better credible interval coverage. Although I show the forest is a Markov chain with stationary distribution, it is still not clear if the algorithm is a Gibbs sampler corresponding to a valid posterior.

There are several new extensions of XBART I am working on, including the joint work with P. Richard Hahn and Nikolay Krantsevich to apply XBART to Bayesian causal forest (Hahn et al., 2020). As previously mentioned in chapter 5, there are a few potential directions to improve the performance of the XBART classification, which is currently under investigation with P. Richard Hahn, Maggie Wang, and Jared Murray.

The software package `XBART` is available online in `R` and `python`. The package is still undergoing active development for further extensions and speed optimization.

APPENDIX A

CATEGORICAL COVARIATES

Section 2.5.1 suggests pre-sorting covariates to compute sufficient statistics efficiently, this strategy is straightforward for continuous covariates. However, because of possible ties in ordered categorical covariates, a more efficient algorithm is needed to calculate sufficient statistics.

We restate notations in section 2.5.1. Without loss of generality, we assume that all covariates are categorical. Let $\mathbf{O}$ denote the V-by-n array such that o_{vh} denotes the index, in the data, of the observation with the h-th smallest value of the v-th predictor variable x_v. Then, taking the cumulative sums gives

$$s(\leq, v, c) = \sum_{h \leq c} \mathrm{r}_{o_{vh}}$$

and

$$s(>, v, c) = \sum_{h=1}^{n} r_{lh} - s(\leq, v, c).$$

The subscript l on the residual indicates that these evaluations pertain to the update of the lth tree. Notice that when covariates are categorical, x_{vh} is not necessarily smaller than $x_{v(h+1)}$ due to potential ties in x. As a result, the number of unique cutpoint candidates is less than n. We propose an extra data structure to bookkeeping unique cutpoint and number of ties as follows. For the v-th categorical predictor variable x_v, let `unique_val` be a vector of unique values (sorted, from small to large) in x_v and `val_count` be a vector of counts of replication for each unique value. Therefore, the cutpoint candidate is a element in the vector `unique_val`, say the i-th element. Then the cumulative sums is

$$s(\leq, v, \texttt{unique\_val}[i]) = \sum_{h \in [\sum_{m=1}^{i-1} \texttt{val\_count[m]}, \sum_{m=1}^{i} \texttt{val\_count[m]}]} r_{o_{vh}}.$$

Algorithm 7 Pseudocode of calculating sufficient statistics for categorical covariates.

1: Sort categorical covariates, create **O** matrix. Count number of unique observations `unique_val` and `val_count` vector (suppose vectors are length K).
2: **for** i from 1 to K **do**
3: Calculate sufficient statistics for cutpoint candidate `unique_val`[i] as

$$s(\leq, v, \texttt{unique\_val}[i]) = \sum_{h \in [\sum_{m=1}^{i-1} \texttt{val\_count[m]}, \sum_{m=1}^{i} \texttt{val\_count[m]}]} r_{o_{vh}}.$$

and

$$s(>, v, c) = \sum_{h=1}^{n} r_{lh} - s(\leq, v, c).$$

4: **end for**
5: Calculate split criterion, determine a cutpoint.
6: **if** no-split is selected or stop conditions are reached **then**
7: Draw leaf parameters and **return**.
8: **else**
9: Sift `unique_val` and `val_count` for left and right child nodes. Repeat step 3 when evaluate split criterion at child nodes.
10: **end if**

When sifting data to left and right child after drawing a cutpoint, we create the same `unique_val` and `val_count` vector for all categorical covariates with data in two child nodes respectively. See Algorithm 7 for details.

APPENDIX B

PROOF OF LEMMA 2

First we establish the connection between theoretical split criterion of XBART (equation (3.4)) and CART. For current node A, the theoretical split criterion of XBART for candidate split at variable i and value x is

$$\begin{aligned} L^*(x) = \frac{1}{\sigma^2}\mathbb{P}(\mathbf{x}^{(i)} \leq x \mid \mathbf{x} \in A) \left[\mathbb{E}(y \mid \mathbf{x}^{(i)} \leq x, \mathbf{x} \in A)\right]^2 \\ + \frac{1}{\sigma^2}\mathbb{P}(\mathbf{x}^{(i)} > x \mid \mathbf{x} \in A) \left[\mathbb{E}(y \mid \mathbf{x}^{(i)} > x, \mathbf{x} \in A)\right]^2. \end{aligned} \tag{B.1}$$

The CART theoretical split criterion is

$$\begin{aligned} L^*_{\text{CART}}(x) = \mathbb{V}(y \mid \mathbf{x} \in A) - \mathbb{P}(\mathbf{x}^{(j)} \leq x \mid \mathbf{x} \in A)\mathbb{V}(y \mid \mathbf{x}^{(j)} \leq x, \mathbf{x} \in A) \\ - \mathbb{P}(\mathbf{x}^{(j)} > x \mid \mathbf{x} \in A)\mathbb{V}(y \mid \mathbf{x}^{(j)} > x, \mathbf{x} \in A). \end{aligned} \tag{B.2}$$

Remember that the cuts is always parallel to axis,

$$\mathbb{V}(y \mid \mathbf{x}^{(j)} \leq x, \mathbf{x} \in A) = \mathbb{E}(y^2 \mid \mathbf{x}^{(j)} \leq x, \mathbf{x} \in A) - \left[\mathbb{E}(y \mid \mathbf{x}^{(j)} \leq x, \mathbf{x} \in A)\right]^2. \tag{B.3}$$

We have

$$\mathbb{E}(y^2 \mid \mathbf{x}^{(j)} \leq x, \mathbf{x} \in A) = \frac{1}{\Omega\left(\left\{\mathbf{x}^{(j)} \leq x, \mathbf{x} \in A\right\}\right)} \int_{\mathbf{x} \in \left\{\mathbf{x}^{(j)} \leq x, \mathbf{x} \in A\right\}} m^2(\mathbf{x}) d\mathbf{x}, \tag{B.4}$$

where $\Omega(A)$ represents volume of a cube A. Observe that

$$\mathbb{P}(\mathbf{x}^{(j)} \leq x \mid \mathbf{x} \in A) = \frac{\Omega\left(\left\{\mathbf{x}^{(j)} \leq x, \mathbf{x} \in A\right\}\right)}{\Omega(A)},$$

by easy calculation, we obtain

$$
\begin{aligned}
&\mathbb{E}(y^2 \mid \mathbf{x} \in A) - \mathbb{P}(\mathbf{x}^{(j)} \leq x \mid \mathbf{x} \in A)\mathbb{E}(y^2 \mid \mathbf{x}^{(j)} \leq x, \mathbf{x} \in A) \\
&\quad - \mathbb{P}(\mathbf{x}^{(j)} > x \mid \mathbf{x} \in A)\mathbb{E}(y^2 \mid \mathbf{x}^{(j)} > x, \mathbf{x} \in A) \\
&= \frac{1}{\Omega(A)} \int_{\mathbf{x} \in A} m^2(\mathbf{x}) d\mathbf{x} - \frac{1}{\Omega(A)} \int_{\mathbf{x} \in \{\mathbf{x}^{(j)} \leq x, \mathbf{x} \in A\}} m^2(\mathbf{x}) d\mathbf{x} - \frac{1}{\Omega(A)} \int_{\mathbf{x} \in \{\mathbf{x}^{(j)} > x, \mathbf{x} \in A\}} m^2(\mathbf{x}) d\mathbf{x} \\
&= 0.
\end{aligned}
\tag{B.5}
$$

As a result, the CART theoretical split criterion is equivalent to

$$
\begin{aligned}
L^*_{\text{CART}}(x) &= [\mathbb{E}(y \mid \mathbf{x} \in A)]^2 - \mathbb{P}(\mathbf{x}^{(i)} \leq x \mid \mathbf{x} \in A) \left[\mathbb{E}(y \mid \mathbf{x}^{(i)} \leq x, \mathbf{x} \in A)\right]^2 \\
&\quad - \mathbb{P}(\mathbf{x}^{(i)} > x \mid \mathbf{x} \in A) \left[\mathbb{E}(y \mid \mathbf{x}^{(i)} > x, \mathbf{x} \in A)\right]^2 \\
&= [\mathbb{E}(y \mid \mathbf{x} \in A)]^2 - \sigma^2 L^*_{\text{XBART}}(x).
\end{aligned}
\tag{B.6}
$$

Since $[\mathbb{E}(y \mid \mathbf{x} \in A)]^2$ and σ^2 are constant, and we maximize $L^*_{\text{XBART}}(x)$ but minimize $L^*_{\text{CART}}(x)$ in practice, we claim that the two *theoretical* split criterions are equivalent. Therefore, the proof of lemma 2 for CART case in Scornet et al. (2015) can be applied directly without modification. We refer readers to Scornet et al. (2015) for details.

APPENDIX C

PROOF OF LEMMA 3

C.1 Proof of Lemma 3 for the case $k = 1$

Preliminary results Let $Z_i = \max_{1 \leq i \leq n} |\epsilon_i|$, we have

$$\mathbb{P}(Z_i \geq t) = 1 - \exp\left[n \ln(1 - 2\mathbb{P}(\epsilon_i \geq t))\right].$$

The tail of Gaussian distribution has a standard bound:

$$\mathbb{P}(\epsilon_i \geq t) \leq \frac{\sigma}{t\sqrt{2\pi}} \left(-\frac{t^2}{2\sigma^2}\right). \tag{C.1}$$

As a result, there exist a positive constant C_ρ and $N_1 \in \mathbb{N}^*$ such that with probability $1 - \rho$, for all $n > N_1$,

$$\max_{1 \leq i \leq n} |\epsilon_i| \leq C_\rho \sqrt{\log(n)}. \tag{C.2}$$

In addition, we have

$$\mathbb{P}\left[\left|\frac{1}{n}\sum_{i=1}^{n} \epsilon_i\right| \geq \alpha\right] \leq \frac{\sigma}{\alpha\sqrt{n}} \exp\left(-\frac{\alpha^2 n}{2\sigma^2}\right). \tag{C.3}$$

Let $\mathrm{N}_n(A)$ denotes number of data observations in a set A. Next we derive from the inequality above and union bound inequality that there exists $N_2 \in \mathbb{N}^*$ such that with probability $1 - \rho$, for all $n > N_2$ and all $0 \leq a_n \leq b_n \leq 1$ satisfying $\mathrm{N}_n([a_n, b_n] \times [0,1]^{p-1}) > \sqrt{n}$,

$$\left|\frac{1}{\mathrm{N}_n([a_n, b_n] \times [0,1]^{p-1})} \sum_{i: X_i \in [a_n, b_n] \times [0,1]^{p-1}} \epsilon_i\right| \leq \alpha. \tag{C.4}$$

and

$$\frac{1}{\mathrm{N}_n([a_n, b_n] \times [0,1]^{p-1})} \sum_{i: X_i \in [a_n, b_n] \times [0,1]^{p-1}} \epsilon_i^2 \leq \tilde{\sigma}^2. \tag{C.5}$$

Furthermore, it's easy to verify

$$\left| \frac{1}{\mathrm{N}_n([a_n, b_n] \times [0,1]^{p-1})} \sum_{i: X_i \in [a_n, b_n] \times [0,1]^{p-1}} Y_i \right| \leq ||m||_\infty + \alpha, \tag{C.6}$$

and

$$\left| \frac{1}{\mathrm{N}_n([a_n, b_n] \times [0,1]^{p-1})} \sum_{i: X_i \in [a_n, b_n] \times [0,1]^{p-1}} Y_i^2 \right| \leq ||m||_\infty^2 + \tilde{\sigma}^2 + 2\alpha ||m||_\infty. \tag{C.7}$$

By the Glivenko-Cantelli theorem, there exist $N_3 \in \mathbb{N}^*$ such that with probability $1 - \rho$, for all $0 \leq a \leq b \leq 1$ and all $n > N_3$,

$$(b - a - \delta^2)n \leq \mathrm{N}_n([a_n, b_n] \times [0,1]^{p-1}) \leq (b - a + \delta^2)n. \tag{C.8}$$

In the following proof, we assume to be on the event that all claims above holds with probability $1 - 3\rho$ for all $n > N = \max\{N_1, N_2, N_3\}$. Take $x_1, x_2 \in [0,1]$ such that $|x_1 - x_2| < \delta$ and assume that $x_1 < x_2$. We partition the space $[0,1]^p$ into several pieces as follows, see Figure C.1 for an illustration of notations for $p = 2$.

$$\begin{cases} A_{L,\sqrt{\delta}} = [0, \sqrt{\delta}] \times [0,1]^{p-1} \\ A_{R,\sqrt{\delta}} = [1 - \sqrt{\delta}, 1] \times [0,1]^{p-1} \\ A_{C,\sqrt{\delta}} = [\sqrt{\delta}, 1 - \sqrt{\delta}] \times [0,1]^{p-1} \end{cases} . \tag{C.9}$$

Similarly,

$$\begin{cases} A_{L,1} = [0, x_1] \times [0,1]^{p-1} \\ A_{R,1} = [x_1, 1] \times [0,1]^{p-1} \\ A_{L,2} = [0, x_2] \times [0,1]^{p-1} \\ A_{R,2} = [x_2, 1] \times [0,1]^{p-1} \\ A_C = [x_1, x_2] \times [0,1]^{p-1} \end{cases} . \tag{C.10}$$

Figure C.1 illustrates notations

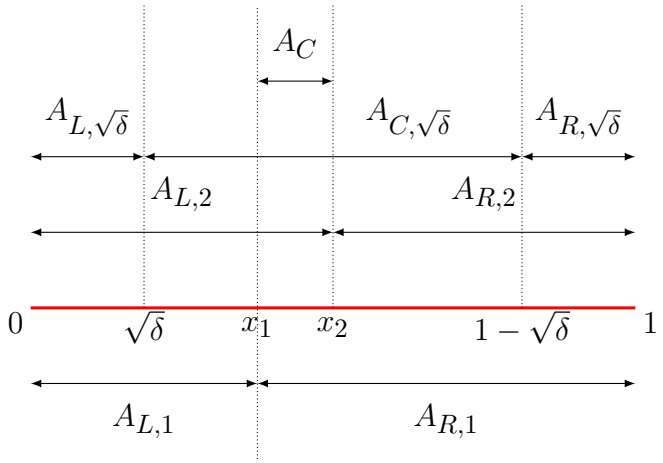


Figure C.1: Illustration of notations for $p = 2$.

For simplicity, we write the split criterion of the first cut as $L_{n,1}(1, x)$ denoting split at the first variable, at value x. Recall that our split criterion is defined as

$$\begin{aligned} L_{n,1}(1,x) = & \frac{\tau}{\sigma^2\left(\sigma^2+\tau \mathrm{N}_n(A_L)\right)} \frac{1}{n}\left(\mathrm{N}_n(A_L) \sum_{i:\mathbf{x}_i^{(1)} \leq x} y_i^2-\left(\mathrm{N}_n(A_L)-1\right) \sum_{i:\mathbf{x}_i^{(1)} \leq x}\left(y_i-\bar{y}_l\right)^2\right) \\ & +\frac{\tau}{\sigma^2\left(\sigma^2+\tau \mathrm{N}_n(A_R)\right)} \frac{1}{n}\left(\mathrm{N}_n(A_R) \sum_{i:\mathbf{x}_i^{(1)}>x} y_i^2-\left(\mathrm{N}_n(A_R)-1\right) \sum_{i:\mathbf{x}_i^{(1)}>x}\left(y_i-\bar{y}_r\right)^2\right) \\ & +\frac{\gamma_x}{n}. \end{aligned} \tag{C.11}$$

The difference of split criterion on two cutpoints x_1 and x_2 is

$$
\begin{aligned}
& L_{n,1}(1,x_1) - L_{n,1}(1,x_2) \\
=& \frac{\tau}{\sigma^2\left(\sigma^2+\tau \mathrm{N}_n(A_{L,1})\right)} \frac{1}{n}\left(\mathrm{N}_n(A_{L,1}) \sum_{i:\mathbf{x}_i^{(1)} \leq x_1} y_i^2 - (\mathrm{N}_n(A_{L,1})-1) \sum_{i:\mathbf{x}_i^{(1)} \leq x_1} (y_i - \bar{y}_{A_{L,1}})^2\right) \\
& + \frac{\tau}{\sigma^2\left(\sigma^2+\tau \mathrm{N}_n(A_{R,1})\right)} \frac{1}{n}\left(\mathrm{N}_n(A_{R,1}) \sum_{i:\mathbf{x}_i^{(1)} > x_1} y_i^2 - (\mathrm{N}_n(A_{R,1})-1) \sum_{i:\mathbf{x}_i^{(1)} > x_1} (y_i - \bar{y}_{A_{R,1}})^2\right) \\
& - \frac{\tau}{\sigma^2\left(\sigma^2+\tau \mathrm{N}_n(A_{L,2})\right)} \frac{1}{n}\left(\mathrm{N}_n(A_{L,2}) \sum_{i:\mathbf{x}_i^{(1)} \leq x_2} y_i^2 - (\mathrm{N}_n(A_{L,2})-1) \sum_{i:\mathbf{x}_i^{(1)} \leq x_2} (y_i - \bar{y}_{A_{L,2}})^2\right) \\
& - \frac{\tau}{\sigma^2\left(\sigma^2+\tau \mathrm{N}_n(A_{R,2})\right)} \frac{1}{n}\left(\mathrm{N}_n(A_{R,2}) \sum_{i:\mathbf{x}_i^{(1)} > x_2} y_i^2 - (\mathrm{N}_n(A_{R,2})-1) \sum_{i:\mathbf{x}_i^{(1)} > x_2} (y_i - \bar{y}_{A_{R,2}})^2\right) \\
& + \frac{\gamma_{x_1}}{n} - \frac{\gamma_{x_2}}{n}.
\end{aligned}
\tag{C.12}
$$

We need to prove lemma 3 for all possible cases depending on location of x_1 and x_2. For notation simplicity, note that after collecting terms, the difference of split criterion can be represented as summation of points for the range of index $\{i : \mathbf{x}_i^{(1)} < x_1\}$, $\{i : \mathbf{x}_i^{(1)} \in [x_1, x_2]\}$ and $\{i : \mathbf{x}_i^{(1)} > x_2\}$. We will use the same decomposition throughout the proof.

First case

Assume that $x_1, x_2 \in A_{C,\sqrt{\delta}}$, two cutpoint candidates are not close to the edge. Consider the

split criterion

$$
\begin{aligned}
&L_{n,1}(1,x_1) - L_{n,1}(1,x_2) \\
=&\frac{\tau}{\sigma^2\left(\sigma^2+\tau \mathrm{N}_n(A_{L,1})\right)}\frac{1}{n}\left(\mathrm{N}_n(A_{L,1})\sum_{i:\mathbf{x}_i^{(1)}\le x_1} y_i^2 - (\mathrm{N}_n(A_{L,1})-1)\sum_{i:\mathbf{x}_i^{(1)}\le x_1}(y_i-\bar{y}_{A_{L,1}})^2\right) \\
&+\frac{\tau}{\sigma^2\left(\sigma^2+\tau \mathrm{N}_n(A_{R,1})\right)}\frac{1}{n}\left(\mathrm{N}_n(A_{R,1})\sum_{i:\mathbf{x}_i^{(1)}> x_1} y_i^2 - (\mathrm{N}_n(A_{R,1})-1)\sum_{i:\mathbf{x}_i^{(1)}> x_1}(y_i-\bar{y}_{A_{R,1}})^2\right) \\
-&\frac{\tau}{\sigma^2\left(\sigma^2+\tau \mathrm{N}_n(A_{L,2})\right)}\frac{1}{n}\left(\mathrm{N}_n(A_{L,2})\sum_{i:\mathbf{x}_i^{(1)}\le x_2} y_i^2 - (\mathrm{N}_n(A_{L,2})-1)\sum_{i:\mathbf{x}_i^{(1)}\le x_2}(y_i-\bar{y}_{A_{L,2}})^2\right) \\
&-\frac{\tau}{\sigma^2\left(\sigma^2+\tau \mathrm{N}_n(A_{R,2})\right)}\frac{1}{n}\left(\mathrm{N}_n(A_{R,2})\sum_{i:\mathbf{x}_i^{(1)}> x_2} y_i^2 - (\mathrm{N}_n(A_{R,2})-1)\sum_{i:\mathbf{x}_i^{(1)}> x_2}(y_i-\bar{y}_{A_{R,2}})^2\right) \\
+&\frac{\gamma_{x_1}}{n} - \frac{\gamma_{x_2}}{n} \\
&= J_1 + J_2 + J_3 + \frac{\gamma_{x_1}}{n} - \frac{\gamma_{x_2}}{n}.
\end{aligned}
\tag{C.13}
$$

First, take n large enough, we have

$$
\left|\frac{\gamma_{x_1}}{n} - \frac{\gamma_{x_2}}{n}\right| \le \alpha. \tag{C.14}
$$

Let J_2 corresponding to $\{i \mid \mathbf{x}_i^{(1)} \in [x_1, x_2]\}$

$$
\begin{aligned}
J_2 = & \frac{\tau}{\sigma^2\left(\sigma^2+\tau \mathrm{N}_n(A_{R,1})\right)} \frac{1}{n}\left(\mathrm{N}_n(A_{R,1}) \sum_{i:\mathbf{x}_i^{(1)}\in[x_1,x_2]} y_i^2-\left(\mathrm{N}_n(A_{R,1})-1\right) \sum_{i:\mathbf{x}_i^{(1)}\in[x_1,x_2]}\left(y_i-\bar{y}_{A_{R,1}}\right)^2\right) \\
& -\frac{\tau}{\sigma^2\left(\sigma^2+\tau \mathrm{N}_n(A_{L,2})\right)} \frac{1}{n}\left(\mathrm{N}_n(A_{L,2}) \sum_{i:\mathbf{x}_i^{(1)}\in[x_1,x_2]} y_i^2\right. \\
& \left.-\left(\mathrm{N}_n(A_{L,2})-1\right) \sum_{i:\mathbf{x}_i^{(1)}\in[x_1,x_2]}\left(y_i-\bar{y}_{A_{L,2}}\right)^2\right) \\
= & \frac{\tau \mathrm{N}_n(A_{R,1})}{\sigma^2\left(\sigma^2+\tau \mathrm{N}_n(A_{R,1})\right)}\left(\frac{1}{n} \sum_{i:\mathbf{x}_i^{(1)}\in[x_1,x_2]} y_i^2\right)-\frac{\tau \mathrm{N}_n(A_{L,2})}{\sigma^2\left(\sigma^2+\tau \mathrm{N}_n(A_{L,2})\right)}\left(\frac{1}{n} \sum_{i:\mathbf{x}_i^{(1)}\in[x_1,x_2]} y_i^2\right) \\
& +\frac{\tau\left(\mathrm{N}_n(A_{L,2})-1\right)}{\sigma^2\left(\sigma^2+\tau \mathrm{N}_n(A_{L,2})\right)}\left(\frac{1}{n} \sum_{i:\mathbf{x}_i^{(1)}\in[x_1,x_2]}\left(y_i-\bar{y}_{A_{L,2}}\right)^2\right) \\
& -\frac{\tau\left(\mathrm{N}_n(A_{R,1})-1\right)}{\sigma^2\left(\sigma^2+\tau \mathrm{N}_n(A_{R,1})\right)}\left(\frac{1}{n} \sum_{i:\mathbf{x}_i^{(1)}\in[x_1,x_2]}\left(y_i-\bar{y}_{A_{R,1}}\right)^2\right) \\
= & J_{21}+J_{22} .
\end{aligned}
\tag{C.15}
$$

Note that $|ax-by| \leq |a||x-y|+|a-b||y|$, we have

$$
\begin{aligned}
|J_{22}| = & \left|\frac{\tau\left(\mathrm{N}_n(A_{L,2})-1\right)}{\sigma^2\left(\sigma^2+\tau \mathrm{N}_n(A_{L,2})\right)}\left(\frac{1}{n} \sum_{i:\mathbf{x}_i^{(1)}\in[x_1,x_2]}\left(y_i-\bar{y}_{A_{L,2}}\right)^2\right)\right. \\
& \left.-\frac{\tau\left(\mathrm{N}_n(A_{R,1})-1\right)}{\sigma^2\left(\sigma^2+\tau \mathrm{N}_n(A_{R,1})\right)}\left(\frac{1}{n} \sum_{i:\mathbf{x}_i^{(1)}\in[x_1,x_2]}\left(y_i-\bar{y}_{A_{R,1}}\right)^2\right)\right| \\
\leq & \left|\frac{\tau\left(\mathrm{N}_n(A_{L,2})-1\right)}{\sigma^2\left(\sigma^2+\tau \mathrm{N}_n(A_{L,2})\right)}\right| \times\left|\frac{1}{n} \sum_{i:\mathbf{x}_i^{(1)}\in[x_1,x_2]}\left(y_i-\bar{y}_{A_{L,2}}\right)^2-\frac{1}{n} \sum_{i:\mathbf{x}_i^{(1)}\in[x_1,x_2]}\left(y_i-\bar{y}_{A_{R,1}}\right)^2\right| \\
& +\left|\frac{\tau\left(\mathrm{N}_n(A_{L,2})-1\right)}{\sigma^2\left(\sigma^2+\tau \mathrm{N}_n(A_{L,2})\right)}-\frac{\tau\left(\mathrm{N}_n(A_{R,1})-1\right)}{\sigma^2\left(\sigma^2+\tau \mathrm{N}_n(A_{R,1})\right)}\right| \times\left|\frac{1}{n} \sum_{i:\mathbf{x}_i^{(1)}\in[x_1,x_2]}\left(y_i-\bar{y}_{A_{R,1}}\right)^2\right| .
\end{aligned}
\tag{C.16}
$$

Since we assume that $x_1, x_2 \in A_{C,\sqrt{\delta}}$, by equation (C.8)

$$\begin{aligned}\left|\frac{\tau(\mathrm{N}_n(A_{L,2})-1)}{\sigma^2\left(\sigma^2+\tau\mathrm{N}_n(A_{L,2})\right)}\right| &\leq \left|\frac{\tau(\delta^2-\sqrt{\delta})n}{\sigma^2(\sigma^2+\tau(1-\delta^2-\sqrt{\delta})n)}\right| \\ &\leq \left|\frac{\tau(\delta^2-\sqrt{\delta})}{\sigma^2(\tau(1-\delta^2-\sqrt{\delta}))}\right| \\ &= C(\delta) \to 0 \text{ as } \delta \to 0.\end{aligned} \tag{C.17}$$

Note that this bound is valid for $\mathrm{N}_n(A_{L,1}), \mathrm{N}_n(A_{L,2}), \mathrm{N}_n(A_{R,1})$ and $\mathrm{N}_n(A_{R,2})$. By inequality (C.6) and (C.7), it is obvious that

$$\left|\frac{1}{n}\sum_{i:\mathbf{x}_i^{(1)}\in[x_1,x_2]}(y_i-\bar{y}_{A_{R,1}})^2\right| \leq \left|\frac{1}{N(A_C)}\sum_{i:\mathbf{x}_i^{(1)}\in[x_1,x_2]}(y_i-\bar{y}_{A_{R,1}})^2\right| \leq M \tag{C.18}$$

by a constant M. Furthermore

$$\left|\frac{\tau(\mathrm{N}_n(A_{L,2})-1)}{\sigma^2\left(\sigma^2+\tau\mathrm{N}_n(A_{L,2})\right)} - \frac{\tau(\mathrm{N}_n(A_{R,1})-1)}{\sigma^2\left(\sigma^2+\tau\mathrm{N}_n(A_{R,1})\right)}\right| \leq 2C(\delta). \tag{C.19}$$

The bound of second term follows equation (8) in supplementary materials of Scornet et al. (2015) directly,

$$|J_{22}| \leq C(\delta)\times 4(||m||_\infty+\alpha)\left((\delta+\delta^2)(2||m||_\infty+\alpha)+\alpha\right)+2C(\delta)M. \tag{C.20}$$

The other term J_{21} is

$$\begin{aligned}|J_{21}| &= \left|\frac{\tau\mathrm{N}_n(A_{R,1})}{\sigma^2\left(\sigma^2+\tau\mathrm{N}_n(A_{R,1})\right)}\left(\frac{1}{n}\sum_{i:\mathbf{x}_i^{(1)}\in[x_1,x_2]}y_i^2\right) - \frac{\tau\mathrm{N}_n(A_{L,2})}{\sigma^2\left(\sigma^2+\tau\mathrm{N}_n(A_{L,2})\right)}\left(\frac{1}{n}\sum_{i:\mathbf{x}_i^{(1)}\in[x_1,x_2]}y_i^2\right)\right| \\ &\leq \left|\frac{\tau\mathrm{N}_n(A_{R,1})}{\sigma^2\left(\sigma^2+\tau\mathrm{N}_n(A_{R,1})\right)} - \frac{\tau\mathrm{N}_n(A_{L,2})}{\sigma^2\left(\sigma^2+\tau\mathrm{N}_n(A_{L,2})\right)}\right| \times \left|\frac{1}{n}\sum_{i:\mathbf{x}_i^{(1)}\in[x_1,x_2]}y_i^2\right|.\end{aligned} \tag{C.21}$$

The bound of coefficient here is slightly different from equation (C.17) and (C.19)

$$\begin{aligned}
&\left|\frac{\tau \mathrm{N}_n(A_{R,1})}{\sigma^2\left(\sigma^2+\tau \mathrm{N}_n(A_{R,1})\right)}-\frac{\tau \mathrm{N}_n(A_{L,2})}{\sigma^2\left(\sigma^2+\tau \mathrm{N}_n(A_{L,2})\right)}\right| \\
=&\left|\frac{\tau(\mathrm{N}_n(A_{R,1})-\mathrm{N}_n(A_{L,2}))}{(\sigma^2+\tau \mathrm{N}_n(A_{R,1}))(\sigma^2+\tau \mathrm{N}_n(A_{L,2}))}\right| \\
\leq&\left|\frac{\tau}{(\sigma^2+\tau \mathrm{N}_n(A_{R,1}))(\sigma^2+\tau \mathrm{N}_n(A_{L,2}))}\right|\left(|\mathrm{N}_n(A_{R,1})|+|\mathrm{N}_n(A_{L,2})|\right) \\
\leq&\frac{2\tau(1-\sqrt{\delta}+\delta^2)n}{\left(\sigma^2+\tau(1-\sqrt{\delta}-\delta^2)n\right)^2}=g(\delta,n)\to 0 \text{ when } n \text{ is large.}
\end{aligned} \tag{C.22}$$

Note that the upper bounds in equation (C.17) and (C.19) can be arbitrarily small if $\delta \to 0$, but the upper bound in equation (C.22) relies on making n large. Use the tail bound of non-central χ^2 distribution, result of supplementary materials of Scornet et al. (2015), and similar to J_{22}

$$|J_{21}| \leq g(\delta, n)M, \tag{C.23}$$

which can be arbitrarily small when n is large.

Now we switch to J_1, corresponding to $i \mid X_i^{(1)} \in [0, x_1]$, we proceed with similar decomposition.

$$
\begin{aligned}
J_1 =& \frac{\tau}{\sigma^2\left(\sigma^2+\tau \mathrm{N}_n(A_{L,1})\right)} \frac{1}{n}\left(\mathrm{N}_n(A_{L,1}) \sum_{i:\mathbf{x}_i^{(1)} \leq x_1} y_i^2-\left(\mathrm{N}_n(A_{L,1})-1\right) \sum_{i:\mathbf{x}_i^{(1)} \leq x_1}\left(y_i-\bar{y}_{A_{L,1}}\right)^2\right) \\
& -\frac{\tau}{\sigma^2\left(\sigma^2+\tau \mathrm{N}_n(A_{L,2})\right)} \frac{1}{n}\left(\mathrm{N}_n(A_{L,2}) \sum_{i:\mathbf{x}_i^{(1)} \leq x_1} y_i^2-\left(\mathrm{N}_n(A_{L,2})-1\right) \sum_{i:\mathbf{x}_i^{(1)} \leq x_1}\left(y_i-\bar{y}_{A_{L,2}}\right)^2\right) \\
=& \frac{\tau \mathrm{N}_n(A_{L,1})}{\sigma^2\left(\sigma^2+\tau \mathrm{N}_n(A_{L,1})\right)}\left(\frac{1}{n} \sum_{i:\mathbf{x}_i^{(1)} \leq x_1} y_i^2\right)-\frac{\tau \mathrm{N}_n(A_{L,2})}{\sigma^2\left(\sigma^2+\tau \mathrm{N}_n(A_{L,2})\right)}\left(\frac{1}{n} \sum_{i:\mathbf{x}_i^{(1)} \leq x_1} y_i^2\right) \\
& +\frac{\tau\left(\mathrm{N}_n(A_{L,2})-1\right)}{\sigma^2\left(\sigma^2+\tau \mathrm{N}_n(A_{L,2})\right)}\left(\frac{1}{n} \sum_{i:\mathbf{x}_i^{(1)} \leq x_1}\left(y_i-\bar{y}_{A_{L,2}}\right)^2\right) \\
& -\frac{\tau\left(\mathrm{N}_n(A_{L,1})-1\right)}{\sigma^2\left(\sigma^2+\tau \mathrm{N}_n(A_{L,1})\right)}\left(\frac{1}{n} \sum_{i:\mathbf{x}_i^{(1)} \leq x_1}\left(y_i-\bar{y}_{A_{L,1}}\right)^2\right) \\
=& J_{11}+J_{12}.
\end{aligned}
\tag{C.24}
$$

$$
\begin{aligned}
\left|J_{12}\right| =& \left|\frac{\tau\left(\mathrm{N}_n(A_{L,2})-1\right)}{\sigma^2\left(\sigma^2+\tau \mathrm{N}_n(A_{L,2})\right)}\left(\frac{1}{n} \sum_{i:\mathbf{x}_i^{(1)} \leq x_1}\left(y_i-\bar{y}_{A_{L,2}}\right)^2\right)\right. \\
& \left.-\frac{\tau\left(\mathrm{N}_n(A_{L,1})-1\right)}{\sigma^2\left(\sigma^2+\tau \mathrm{N}_n(A_{L,1})\right)}\left(\frac{1}{n} \sum_{i:\mathbf{x}_i^{(1)} \leq x_1}\left(y_i-\bar{y}_{A_{L,1}}\right)^2\right)\right| \\
\leq & \left|\frac{\tau\left(\mathrm{N}_n(A_{L,2})-1\right)}{\sigma^2\left(\sigma^2+\tau \mathrm{N}_n(A_{L,2})\right)}\right| \times\left|\frac{1}{n} \sum_{i:\mathbf{x}_i^{(1)} \leq x_1}\left(y_i-\bar{y}_{A_{L,2}}\right)^2-\frac{1}{n} \sum_{i:\mathbf{x}_i^{(1)} \leq x_1}\left(y_i-\bar{y}_{A_{L,1}}\right)^2\right| \\
& +\left|\frac{\tau\left(\mathrm{N}_n(A_{L,2})-1\right)}{\sigma^2\left(\sigma^2+\tau \mathrm{N}_n(A_{L,2})\right)}-\frac{\tau\left(\mathrm{N}_n(A_{L,1})-1\right)}{\sigma^2\left(\sigma^2+\tau \mathrm{N}_n(A_{L,1})\right)}\right| \times\left|\frac{1}{n} \sum_{i:\mathbf{x}_i^{(1)} \leq x_1}\left(y_i-\bar{y}_{A_{L,1}}\right)^2\right| .
\end{aligned}
\tag{C.25}
$$

Same as J_{22},

$$\begin{aligned} |J_{12}| &\leq C(\delta) \times \left| \frac{1}{n} \sum_{i:\mathbf{x}_i^{(1)} \in [x_1, x_2]} (y_i - \bar{y}_{A_{R,1}})^2 \right| + 2C(\delta)M \\ &\leq C(\delta) \times 5(||m||_\infty \sqrt{\delta} + \alpha) + 2C(\delta)M. \end{aligned} \tag{C.26}$$

The second equation above use result of equation (9) of supplementary material of Scornet et al. (2015). Similar to J_{21}, we have

$$\begin{aligned} |J_{11}| &= \left| \frac{\tau \mathrm{N}_n(A_{L,1})}{\sigma^2 \left(\sigma^2 + \tau \mathrm{N}_n(A_{L,1})\right)} \left(\frac{1}{n} \sum_{i:\mathbf{x}_i^{(1)} \leq x_1} y_i^2 \right) - \frac{\tau \mathrm{N}_n(A_{L,2})}{\sigma^2 \left(\sigma^2 + \tau \mathrm{N}_n(A_{L,2})\right)} \left(\frac{1}{n} \sum_{i:\mathbf{x}_i^{(1)} \leq x_1} y_i^2 \right) \right| \\ &\leq \left| \frac{\tau \mathrm{N}_n(A_{L,1})}{\sigma^2 \left(\sigma^2 + \tau \mathrm{N}_n(A_{L,1})\right)} - \frac{\tau \mathrm{N}_n(A_{L,2})}{\sigma^2 \left(\sigma^2 + \tau \mathrm{N}_n(A_{L,2})\right)} \right| \times \left| \frac{1}{n} \sum_{i:\mathbf{x}_i^{(1)} \leq x_1} y_i^2 \right| \\ &\leq g(\delta, n)M. \end{aligned} \tag{C.27}$$

J_3 have the same bound as J_1. Collect all terms, we have

$$\begin{aligned} &|J_1| \leq g(\delta, n)M + C(\delta) \times 25(||m||_\infty \sqrt{\delta} + \alpha) + 2C(\delta)M \\ &|J_2| \leq g(\delta, n)M + C(\delta) \times 4(||m||_\infty + \alpha) \left((\delta + \delta^2)(2||m||_\infty + \alpha) + \alpha \right) + 2C(\delta)M \\ &|J_3| \leq g(\delta, n)M + C(\delta) \times 25(||m||_\infty \sqrt{\delta} + \alpha) + 2C(\delta)M \\ &|L_{n,1}(1, x_1) - L_{n,1}(1, x_2)| \leq |J_1| + |J_2| + |J_3| \end{aligned} . \tag{C.28}$$

Consequently, for all n large enough and δ small enough, we have

$$|L_{n,1}(1, x_1) - L_{n,1}(1, x_2)| \leq 3\alpha. \tag{C.29}$$

Second case

Assume that $x_1, x_2 \in A_{L,\sqrt{\delta}}$, take same arguments as above, we have

$$\mathrm{N}_n(A_{L,1}), \mathrm{N}_n(A_{L,2}) \leq (\sqrt{\delta} + \delta^2)n. \tag{C.30}$$

Different from the first case, now both x_1 and x_2 are close to the left edge, which is corresponding to term J_1. Note that $|J_2|$ and $|J_3|$ are the same as the first case since the control over region A_C and $A_{R,1} \times A_{R,2}$ and not changed.

$$\begin{aligned}
|J_{12}| = & \left| \frac{\tau(\mathrm{N}_n(A_{L,2}) - 1)}{\sigma^2 \left(\sigma^2 + \tau \mathrm{N}_n(A_{L,2})\right)} \left(\frac{1}{n} \sum_{i:\mathbf{x}_i^{(1)} \leq x_1} (y_i - \bar{y}_{A_{L,2}})^2 \right) \right. \\
& \left. - \frac{\tau(\mathrm{N}_n(A_{L,1}) - 1)}{\sigma^2 \left(\sigma^2 + \tau \mathrm{N}_n(A_{L,1})\right)} \left(\frac{1}{n} \sum_{i:\mathbf{x}_i^{(1)} \leq x_1} (y_i - \bar{y}_{A_{L,1}})^2 \right) \right| \\
\leq & \left| \frac{\tau(\mathrm{N}_n(A_{L,2}) - 1)}{\sigma^2 \left(\sigma^2 + \tau \mathrm{N}_n(A_{L,2})\right)} \right| \times \left| \frac{1}{n} \sum_{i:\mathbf{x}_i^{(1)} \leq x_1} (y_i - \bar{y}_{A_{L,2}})^2 - \frac{1}{n} \sum_{i:\mathbf{x}_i^{(1)} \leq x_1} (y_i - \bar{y}_{A_{L,1}})^2 \right| \\
& + \left| \frac{\tau(\mathrm{N}_n(A_{L,2}) - 1)}{\sigma^2 \left(\sigma^2 + \tau \mathrm{N}_n(A_{L,2})\right)} - \frac{\tau(\mathrm{N}_n(A_{L,1}) - 1)}{\sigma^2 \left(\sigma^2 + \tau \mathrm{N}_n(A_{L,1})\right)} \right| \times \left| \frac{1}{n} \sum_{i:\mathbf{x}_i^{(1)} \leq x_1} (y_i - \bar{y}_{A_{L,1}})^2 \right|.
\end{aligned} \tag{C.31}$$

We have

$$\left| \frac{\tau(\mathrm{N}_n(A_{L,2}) - 1)}{\sigma^2 \left(\sigma^2 + \tau \mathrm{N}_n(A_{L,2})\right)} \right| \leq \left| \frac{\tau \mathrm{N}_n(A_{L,2})}{\sigma^2 \tau \mathrm{N}_n(A_{L,2})} \right| = \frac{1}{\sigma^2} \tag{C.32}$$

$$
\begin{aligned}
&\left|\frac{1}{n}\sum_{i:\mathbf{x}_i^{(1)}\le x_1}(y_i-\bar{y}_{A_{L,2}})^2-\frac{1}{n}\sum_{i:\mathbf{x}_i^{(1)}\le x_1}(y_i-\bar{y}_{A_{L,1}})^2\right| \\
&=2|\bar{y}_{A_{L,1}}-\bar{y}_{A_{L,2}}|\times\frac{1}{n}\left|\sum_{i:\mathbf{x}_i^{(1)}<x_1}\left(y_i-\frac{\bar{y}_{A_{L,1}}+\bar{y}_{A_{L,2}}}{2}\right)\right| \\
&\le 4(||m||_\infty+\alpha)\left(\frac{(||m||_\infty+\alpha)\mathrm{N}_n(A_{L,1})}{n}+\frac{1}{n}\left|\sum_{i:\mathbf{x}_i^{(1)}<x_1}m(\mathbf{x}_i)+\epsilon_i\right|\right) \\
&\le 4(||m||_\infty+\alpha)\left((||m||_\infty+\alpha)(\sqrt{\delta}+\delta^2)+\frac{\mathrm{N}_n(A_{L,1})}{n}(||m||_\infty+\alpha)\right) \\
&\le 4(||m||_\infty+\alpha)\left((||m||_\infty+\alpha+1)(\sqrt{\delta}+\delta^2)\right)
\end{aligned}
\tag{C.33}
$$

$$
\begin{aligned}
&\left|\frac{\tau(\mathrm{N}_n(A_{L,2})-1)}{\sigma^2\left(\sigma^2+\tau\mathrm{N}_n(A_{L,2})\right)}-\frac{\tau(\mathrm{N}_n(A_{L,1})-1)}{\sigma^2\left(\sigma^2+\tau\mathrm{N}_n(A_{L,1})\right)}\right|\times\left|\frac{1}{n}\sum_{i:\mathbf{x}_i^{(1)}\le x_1}(y_i-\bar{y}_{A_{L,1}})^2\right| \\
=&\left|\frac{\tau(\mathrm{N}_n(A_{L,2})-1)}{\sigma^2\left(\sigma^2+\tau\mathrm{N}_n(A_{L,2})\right)}-\frac{\tau(\mathrm{N}_n(A_{L,1})-1)}{\sigma^2\left(\sigma^2+\tau\mathrm{N}_n(A_{L,1})\right)}\right|\times\frac{\mathrm{N}_n(A_{L,1})}{n}\left|\frac{1}{\mathrm{N}_n(A_{L,1})}\sum_{i:\mathbf{x}_i^{(1)}\le x_1}(y_i-\bar{y}_{A_{L,1}})^2\right| \\
\le&\frac{2}{\sigma^2}(\sqrt{\delta}+\delta^2)M.
\end{aligned}
\tag{C.34}
$$

As a result

$$
|J_{12}|\le\frac{1}{\sigma^2}4(||m||_\infty+\alpha)\left((||m||_\infty+\alpha+1)(\sqrt{\delta}+\delta^2)\right)+\frac{2}{\sigma^2}(\sqrt{\delta}+\delta^2)M\to 0. \tag{C.35}
$$

$$
\begin{aligned}
|J_{11}|&=\left|\frac{\tau\mathrm{N}_n(A_{L,1})}{\sigma^2\left(\sigma^2+\tau\mathrm{N}_n(A_{L,1})\right)}\left(\frac{1}{n}\sum_{i:\mathbf{x}_i^{(1)}\le x_1}y_i^2\right)-\frac{\tau\mathrm{N}_n(A_{L,2})}{\sigma^2\left(\sigma^2+\tau\mathrm{N}_n(A_{L,2})\right)}\left(\frac{1}{n}\sum_{i:\mathbf{x}_i^{(1)}\le x_1}y_i^2\right)\right| \\
&\le\left|\frac{\tau\mathrm{N}_n(A_{L,1})}{\sigma^2\left(\sigma^2+\tau\mathrm{N}_n(A_{L,1})\right)}-\frac{\tau\mathrm{N}_n(A_{L,2})}{\sigma^2\left(\sigma^2+\tau\mathrm{N}_n(A_{L,2})\right)}\right|\times\left|\frac{1}{n}\sum_{i:\mathbf{x}_i^{(1)}\le x_1}y_i^2\right| \\
&\le\frac{2}{\sigma^2}(\sqrt{\delta}+\delta^2)M\to 0.
\end{aligned}
\tag{C.36}
$$

Consequently we conclude that for all $n > N$ and all δ small enough,

$$|L_{n,1}(1, x_1) - L_{n,1}(1, x_2)| \leq 3\alpha. \tag{C.37}$$

The other cases $\{x_1, x_2 \in A_{R,\sqrt{\delta}}\}$, $\{x_1 \in A_{L,\sqrt{\delta}}, x_2 \in A_{C,\sqrt{\delta}}\}$ and $\{x_1 \in A_{C,\sqrt{\delta}}, x_2 \in A_{R,\sqrt{\delta}}\}$ can be proved in the same way. Details are omitted.

C.2 Proof of Lemma 3 for the case $k = 2$

Preliminary results

Similarly, Laurent and Massart (2000) gives tail bound of χ^2 distribution,

$$\mathbb{P}[\chi_n^2 \geq 5n] \leq \exp(-n).$$

By the tail bound above, it's straightforward to show that

Suppose $\mathbf{x}$ follows χ^2 distribution with degrees of freedom k and non-central parameter λ

$$P(\mathbf{x} \geq x) \leq \frac{\sqrt{\pi}}{2e}\Phi(\sqrt{x})I_{\frac{k}{2}}(1)M_{k-1}, \tag{C.38}$$

where I_v is a modified Bessel function of the first kind, $M_{k-1} = E(y^{k-1})$ and y is a Gaussian $(\mu, 1)$ random variable truncated on $(\sqrt{x}, \infty)$. So we can claim that with probability $1 - \rho$, the term $\frac{1}{n}\sum_{i=1}^n y_i^2$ is bounded.

Follow the notation of Scornet et al. (2015), let $d_1' = (1, x_1')$ and $d_2' = (2, x_2'$ be such that $|x_1 - x_1'| \leq \delta$ and $|x_2 - x_2'| \leq \delta$.

There exist a constant $C_\rho > 0$ and N_1 such that, with probability $1 - \rho$, for all $n > N_1$,

$$\max_{1 \leq i \leq n} |\epsilon_i| \leq C_\rho \sqrt{\log(n)} \tag{C.39}$$

and

$$\max_{1 \leq i \leq n} |\epsilon_i^2| \leq C_\rho^2 \log(n). \tag{C.40}$$

Fix $\rho > 0$, there exist N_2 such that, with probability $1 - \rho$, for all $n > N_2$ and all $A_n = [a_n^{(1)}, b_n^{(1)}] \times [a_n^{(2)}, b_n^{(2)}] \subset [0, 1]^2$ satisfying $\mathrm{N}_n(A_n) > \sqrt{n}$,

$$\left| \frac{1}{\mathrm{N}_n(A_n)} \sum_{i:\mathbf{x}_i \in A_n} \epsilon_i \right| \leq \alpha \tag{C.41}$$

and

$$\frac{1}{\mathrm{N}_n(A_n)} \sum_{i:\mathbf{x}_i \in A_n} \epsilon_i^2 \leq \tilde{\sigma}^2. \tag{C.42}$$

Furthermore, it's easy to verify

$$\left| \frac{1}{\mathrm{N}_n(A_n)} \sum_{i:\mathbf{x}_i \in A_n} y_i \right| \leq ||m||_\infty + \alpha \tag{C.43}$$

and

$$\left| \frac{1}{\mathrm{N}_n(A_n)} \sum_{i:\mathbf{x}_i \in A_n} y_i^2 \right| \leq ||m||_\infty^2 + \tilde{\sigma}^2 + 2\alpha||m||_\infty. \tag{C.44}$$

Similar to the $k = 1$ case, we denote partition of space as

$$\begin{cases} A_{R,1} = [x_1, 1] \times [0,1]^{p-1} \\ A_{B,2} = [x_1, 1] \times [0, x_2] \times [0,1]^{p-1} \\ A_{H,2} = [x_1, 1] \times [x_2, 1] \times [0,1]^{p-1} \\ A'_{B,2} = [x'_1, 1] \times [0, x'_2] \times [0,1]^{p-1} \\ A'_{H,2} = [x'_1, 1] \times [x'_2, 1] \times [0,1]^{p-1}. \end{cases} \tag{C.45}$$

See Figure C.2 for an illustration of notations.

Let $d_1 = (1, x_1)$, $d_2 = (2, x_2)$, $d'_1 = (1, x'_1)$ and $d'_2 = (2, x'_2)$ be four cutpoints and $|x_1 - x'_1| < \delta$, $|x_2 - x'_2| < \delta$, then

$$L_n(d_1, d_2) - L_n(d'_1, d'_2) = L_n(d_1, d_2) - L_n(d'_1, d_2) + L_n(d'_1, d_2) - L_n(d'_1, d'_2). \tag{C.46}$$

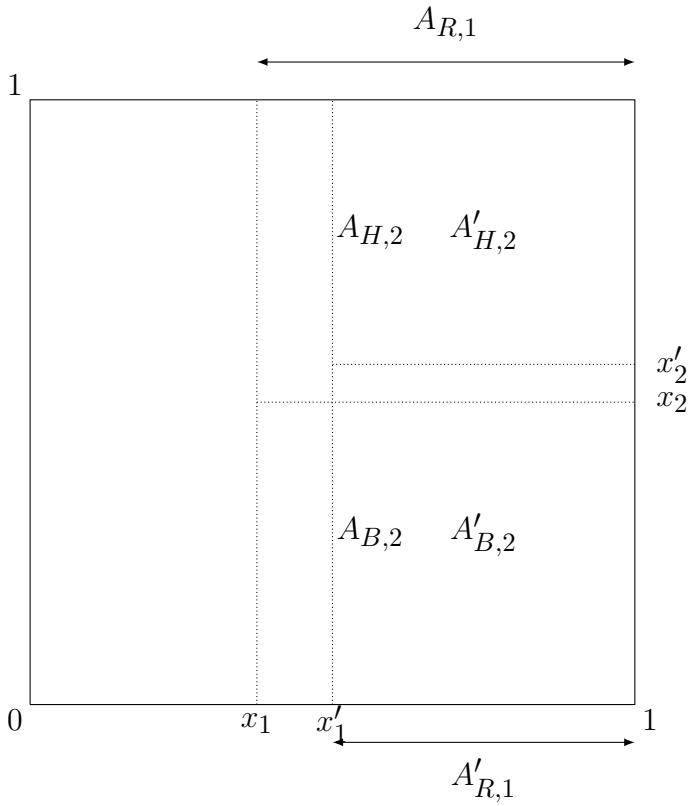


Figure C.2: Illustration of notations for $k = 2$.

$$
\begin{aligned}
&L_n(d_1, d_2) - L_n(d_1', d_2) \\
=&\frac{\tau}{\sigma^2\left(\sigma^2 + \tau \mathrm{N}_n(A_{B,2})\right)} \frac{1}{\mathrm{N}_n(A_{R,1})} \left(\mathrm{N}_n(A_{B,2}) \sum_{i:\mathbf{x}_i^{(2)} \le x_2} y_i^2 \mathbb{1}_{\mathbf{x}_i^{(1)} > x_1} \right. \\
&\qquad \left. -(\mathrm{N}_n(A_{B,2}) - 1) \sum_{i:\mathbf{x}_i^{(2)} \le x_2} (y_i - \bar{y}_{A_{B,2}})^2 \mathbb{1}_{\mathbf{x}_i^{(1)} > x_1} \right) \\
&+ \frac{\tau}{\sigma^2\left(\sigma^2 + \tau \mathrm{N}_n(A_{H,2})\right)} \frac{1}{\mathrm{N}_n(A_{R,1})} \left(\mathrm{N}_n(A_{H,2}) \sum_{i:\mathbf{x}_i^{(2)} > x_2} y_i^2 \mathbb{1}_{\mathbf{x}_i^{(1)} > x_1} \right. \\
&\qquad \left. -(\mathrm{N}_n(A_{H,2}) - 1) \sum_{i:\mathbf{x}_i^{(2)} > x_2} (y_i - \bar{y}_{A_{H,2}})^2 \mathbb{1}_{\mathbf{x}_i^{(1)} > x_1} \right) \\
&- \frac{\tau}{\sigma^2\left(\sigma^2 + \tau \mathrm{N}_n(A_{B,2}')\right)} \frac{1}{\mathrm{N}_n(A_{R,1}')} \left(\mathrm{N}_n(A_{B,2}') \sum_{i:\mathbf{x}_i^{(2)} \le x_2} y_i^2 \mathbb{1}_{\mathbf{x}_i^{(1)} > x_1'} \right. \\
&\qquad \left. -(\mathrm{N}_n(A_{B,2}') - 1) \sum_{i:\mathbf{x}_i^{(2)} \le x_2} (y_i - \bar{y}_{A_{B,2}'})^2 \mathbb{1}_{\mathbf{x}_i^{(1)} > x_1'} \right) \\
&- \frac{\tau}{\sigma^2\left(\sigma^2 + \tau \mathrm{N}_n(A_{H,2}')\right)} \frac{1}{\mathrm{N}_n(A_{R,1}')} \left(\mathrm{N}_n(A_{H,2}') \sum_{i:\mathbf{x}_i^{(2)} > x_2} y_i^2 \mathbb{1}_{\mathbf{x}_i^{(1)} > x_1'} \right. \\
&\qquad \left. -(\mathrm{N}_n(A_{H,2}') - 1) \sum_{i:\mathbf{x}_i^{(2)} > x_2'} (y_i - \bar{y}_{A_{H,2}'})^2 \mathbb{1}_{\mathbf{x}_i^{(1)} > x_1'} \right) \\
&+ \frac{\gamma_{x_1,x_2}}{\mathrm{N}_n(A_{R,1})} - \frac{\gamma_{x_1',x_2}}{\mathrm{N}_n(A_{R,1}')} \\
&= A_1 + B_1
\end{aligned}
\tag{C.47}
$$

$$
\begin{aligned}
A_1 =& \frac{\tau}{\sigma^2\left(\sigma^2+\tau \mathrm{N}_n(A_{H,2})\right)} \frac{1}{\mathrm{N}_n(A_{R,1})} \left(\mathrm{N}_n(A_{H,2}) \sum_{i:\mathbf{x}_i^{(2)}>x_2} y_i^2 \mathbb{1}_{\mathbf{x}_i^{(1)}>x_1} \right. \\
& \left. -(\mathrm{N}_n(A_{H,2})-1) \sum_{i:\mathbf{x}_i^{(2)}>x_2} (y_i - \bar{y}_{A_{H,2}})^2 \mathbb{1}_{\mathbf{x}_i^{(1)}>x_1} \right) \\
& - \frac{\tau}{\sigma^2\left(\sigma^2+\tau \mathrm{N}_n(A'_{H,2})\right)} \frac{1}{\mathrm{N}_n(A'_{R,1})} \left(\mathrm{N}_n(A'_{H,2}) \sum_{i:\mathbf{x}_i^{(2)}>x_2} y_i^2 \mathbb{1}_{\mathbf{x}_i^{(1)}>x'_1} \right. \\
& \left. -(\mathrm{N}_n(A'_{H,2})-1) \sum_{i:\mathbf{x}_i^{(2)}>x'_2} (y_i - \bar{y}_{A'_{H,2}})^2 \mathbb{1}_{\mathbf{x}_i^{(1)}>x'_1} \right) \\
=& A_{1,1} + A_{1,2} + A_{1,3}
\end{aligned}
\tag{C.48}
$$

$$
\begin{aligned}
A_{1,1} =& \frac{\tau}{\sigma^2\left(\sigma^2+\tau \mathrm{N}_n(A_{H,2})\right)} \frac{1}{\mathrm{N}_n(A_{R,1})} \left(\mathrm{N}_n(A_{H,2}) \sum_{i:\mathbf{x}_i^{(2)}>x_2} y_i^2 \mathbb{1}_{\mathbf{x}_i^{(1)}>x'_1} \right. \\
& \left. -(\mathrm{N}_n(A_{H,2})-1) \sum_{i:\mathbf{x}_i^{(2)}>x_2} (y_i - \bar{y}_{A_{H,2}})^2 \mathbb{1}_{\mathbf{x}_i^{(1)}>x'_1} \right) \\
& - \frac{\tau}{\sigma^2\left(\sigma^2+\tau \mathrm{N}_n(A'_{H,2})\right)} \frac{1}{\mathrm{N}_n(A_{R,1})} \left(\mathrm{N}_n(A'_{H,2}) \sum_{i:\mathbf{x}_i^{(2)}>x_2} y_i^2 \mathbb{1}_{\mathbf{x}_i^{(1)}>x'_1} \right. \\
& \left. -(\mathrm{N}_n(A'_{H,2})-1) \sum_{i:\mathbf{x}_i^{(2)}>x_2} (y_i - \bar{y}_{A'_{H,2}})^2 \mathbb{1}_{\mathbf{x}_i^{(1)}>x'_1} \right)
\end{aligned}
\tag{C.49}
$$

$$
\begin{aligned}
A_{1,2} =& \frac{\tau}{\sigma^2\left(\sigma^2+\tau \mathrm{N}_n(A'_{H,2})\right)} \frac{1}{\mathrm{N}_n(A_{R,1})} \left(\mathrm{N}_n(A'_{H,2}) \sum_{i:\mathbf{x}_i^{(2)}>x_2} y_i^2 \mathbb{1}_{\mathbf{x}_i^{(1)}>x'_1} \right. \\
& \left. -(\mathrm{N}_n(A'_{H,2})-1) \sum_{i:\mathbf{x}_i^{(2)}>x_2} (y_i - \bar{y}_{A'_{H,2}})^2 \mathbb{1}_{\mathbf{x}_i^{(1)}>x'_1} \right) \\
& - \frac{\tau}{\sigma^2\left(\sigma^2+\tau \mathrm{N}_n(A'_{H,2})\right)} \frac{1}{\mathrm{N}_n(A'_{R,1})} \left(\mathrm{N}_n(A'_{H,2}) \sum_{i:\mathbf{x}_i^{(2)}>x_2} y_i^2 \mathbb{1}_{\mathbf{x}_i^{(1)}>x'_1} \right. \\
& \left. -(\mathrm{N}_n(A'_{H,2})-1) \sum_{i:\mathbf{x}_i^{(2)}>x'_2} (y_i - \bar{y}_{A'_{H,2}})^2 \mathbb{1}_{\mathbf{x}_i^{(1)}>x'_1} \right)
\end{aligned} \tag{C.50}
$$

$$
\begin{aligned}
A_{1,3} =& \frac{\tau}{\sigma^2\left(\sigma^2+\tau \mathrm{N}_n(A_{H,2})\right)} \frac{1}{\mathrm{N}_n(A_{R,1})} \left(\mathrm{N}_n(A_{H,2}) \sum_{i:\mathbf{x}_i^{(2)}>x_2} y_i^2 \mathbb{1}_{\mathbf{x}_i^{(1)}\in[x_1,x'_1]} \right. \\
& \left. -(\mathrm{N}_n(A_{H,2})-1) \sum_{i:\mathbf{x}_i^{(2)}>x_2} (y_i - \bar{y}_{A_{H,2}})^2 \mathbb{1}_{\mathbf{x}_i^{(1)}\in[x_1,x'_1]} \right)
\end{aligned} \tag{C.51}
$$

$$
\begin{aligned}
A_{1,1} =& \frac{\tau}{\sigma^2\left(\sigma^2+\tau \mathrm{N}_n(A_{H,2})\right)} \frac{1}{\mathrm{N}_n(A_{R,1})} \Bigg(\mathrm{N}_n(A_{H,2}) \sum_{i:\mathbf{x}_i^{(2)}>x_2} y_i^2 \mathbb{1}_{\mathbf{x}_i^{(1)}>x_1'} \\
& \qquad -(\mathrm{N}_n(A_{H,2})-1) \sum_{i:\mathbf{x}_i^{(2)}>x_2} (y_i-\bar{y}_{A_{H,2}})^2 \mathbb{1}_{\mathbf{x}_i^{(1)}>x_1'} \Bigg) \\
& - \frac{\tau}{\sigma^2\left(\sigma^2+\tau \mathrm{N}_n(A'_{H,2})\right)} \frac{1}{\mathrm{N}_n(A_{R,1})} \Bigg(\mathrm{N}_n(A'_{H,2}) \sum_{i:\mathbf{x}_i^{(2)}>x_2} y_i^2 \mathbb{1}_{\mathbf{x}_i^{(1)}>x_1'} \\
& \qquad -(\mathrm{N}_n(A'_{H,2})-1) \sum_{i:\mathbf{x}_i^{(2)}>x_2} (y_i-\bar{y}_{A'_{H,2}})^2 \mathbb{1}_{\mathbf{x}_i^{(1)}>x_1'} \Bigg) \\
=& \left(\frac{\tau}{\sigma^2\left(\sigma^2+\tau \mathrm{N}_n(A_{H,2})\right)} \frac{\mathrm{N}_n(A_{H,2})}{\mathrm{N}_n(A_{R,1})} - \frac{\tau}{\sigma^2\left(\sigma^2+\tau \mathrm{N}_n(A'_{H,2})\right)} \frac{\mathrm{N}_n(A'_{H,2})}{\mathrm{N}_n(A_{R,1})} \right) \sum_{i:\mathbf{x}_i^{(2)}>x_2} y_i^2 \mathbb{1}_{\mathbf{x}_i^{(1)}>x_1'} \\
& + \Bigg(\frac{\tau}{\sigma^2\left(\sigma^2+\tau \mathrm{N}_n(A'_{H,2})\right)} \frac{(\mathrm{N}_n(A'_{H,2})-1)}{\mathrm{N}_n(A_{R,1})} \sum_{i:\mathbf{x}_i^{(2)}>x_2} (y_i-\bar{y}_{A'_{H,2}})^2 \mathbb{1}_{\mathbf{x}_i^{(1)}>x_1'} \\
& \qquad - \frac{\tau}{\sigma^2\left(\sigma^2+\tau \mathrm{N}_n(A_{H,2})\right)} \frac{(\mathrm{N}_n(A_{H,2})-1)}{\mathrm{N}_n(A_{R,1})} \sum_{i:\mathbf{x}_i^{(2)}>x_2} (y_i-\bar{y}_{A_{H,2}})^2 \mathbb{1}_{\mathbf{x}_i^{(1)}>x_1'} \Bigg)
\end{aligned}
\tag{C.52}
$$

which goes to zero using the same argument as $k = 1$ case.

$$
\begin{aligned}
A_{1,2} =& \frac{\tau}{\sigma^2\left(\sigma^2+\tau \mathrm{N}_n(A'_{H,2})\right)} \frac{1}{\mathrm{N}_n(A_{R,1})} \left(\mathrm{N}_n(A'_{H,2}) \sum_{i:\mathbf{x}_i^{(2)}>x_2} y_i^2 \mathbb{1}_{\mathbf{x}_i^{(1)}>x'_1} \right. \\
& \left. -(\mathrm{N}_n(A'_{H,2})-1) \sum_{i:\mathbf{x}_i^{(2)}>x_2} (y_i - \bar{y}_{A'_{H,2}})^2 \mathbb{1}_{\mathbf{x}_i^{(1)}>x'_1} \right) \\
& - \frac{\tau}{\sigma^2\left(\sigma^2+\tau \mathrm{N}_n(A'_{H,2})\right)} \frac{1}{\mathrm{N}_n(A'_{R,1})} \left(\mathrm{N}_n(A'_{H,2}) \sum_{i:\mathbf{x}_i^{(2)}>x_2} y_i^2 \mathbb{1}_{\mathbf{x}_i^{(1)}>x'_1} \right. \\
& \left. -(\mathrm{N}_n(A'_{H,2})-1) \sum_{i:\mathbf{x}_i^{(2)}>x'_2} (y_i - \bar{y}_{A'_{H,2}})^2 \mathbb{1}_{\mathbf{x}_i^{(1)}>x'_1} \right) \\
=& \left(\frac{\tau}{\sigma^2\left(\sigma^2+\tau \mathrm{N}_n(A'_{H,2})\right)} \frac{1}{\mathrm{N}_n(A_{R,1})} - \frac{\tau}{\sigma^2\left(\sigma^2+\tau \mathrm{N}_n(A'_{H,2})\right)} \frac{1}{\mathrm{N}_n(A'_{R,1})} \right) \\
& \times \left(\mathrm{N}_n(A'_{H,2}) \sum_{i:\mathbf{x}_i^{(2)}>x_2} y_i^2 \mathbb{1}_{\mathbf{x}_i^{(1)}>x'_1} - (\mathrm{N}_n(A'_{H,2})-1) \sum_{i:\mathbf{x}_i^{(2)}>x_2} (y_i - \bar{y}_{A'_{H,2}})^2 \mathbb{1}_{\mathbf{x}_i^{(1)}>x'_1} \right)
\end{aligned} \tag{C.53}
$$

$$
\begin{aligned}
|A_{1,2}| \leq & \left| \frac{\tau \mathrm{N}_n(A'_{H,2})}{\sigma^2\left(\sigma^2+\tau \mathrm{N}_n(A'_{H,2})\right)} \frac{\mathrm{N}_n(A'_{H,2})}{\mathrm{N}_n(A_{R,1})} - \frac{\tau \mathrm{N}_n(A'_{H,2})}{\sigma^2\left(\sigma^2+\tau \mathrm{N}_n(A'_{H,2})\right)} \frac{\mathrm{N}_n(A'_{H,2})}{\mathrm{N}_n(A'_{R,1})} \right| \\
& \times \left(\left| \frac{1}{\mathrm{N}_n(A'_{H,2})} \sum_{i:\mathbf{x}_i^{(2)}>x_2} y_i^2 \mathbb{1}_{\mathbf{x}_i^{(1)}>x'_1} \right| + \left| \frac{1}{\mathrm{N}_n(A'_{H,2})} \sum_{i:\mathbf{x}_i^{(2)}>x_2} (y_i - \bar{y}_{A'_{H,2}})^2 \mathbb{1}_{\mathbf{x}_i^{(1)}>x'_1} \right| \right)
\end{aligned} \tag{C.54}
$$

Same as before, the second term is bounded and

$$
|A_{1,2}| \leq M \left| \frac{\mathrm{N}_n(A'_{H,2})}{\mathrm{N}_n(A_{R,1})} - \frac{\mathrm{N}_n(A'_{H,2})}{\mathrm{N}_n(A'_{R,1})} \right| \to 0 \tag{C.55}
$$

$$
\begin{aligned}
|A_{1,3}| \leq & \left| \frac{\tau}{\sigma^2\left(\sigma^2+\tau \mathrm{N}_n(A_{H,2})\right)} \frac{\mathrm{N}_n(A_{H,2})}{\mathrm{N}_n(A_{R,1})} \mathrm{N}_n\left(\left\{\mathbf{x}_i^{(1)} \in [x_1, x_1']\right\} \times \left\{\mathbf{x}_i^{(2)} > x_2\right\}\right)\right| \\
& \times \left| \frac{1}{\mathrm{N}_n\left(\left\{\mathbf{x}_i^{(1)} \in [x_1, x_1']\right\} \times \left\{\mathbf{x}_i^{(2)} > x_2\right\}\right)} \sum_{i:\mathbf{x}_i^{(2)}>x_2} y_i^2 \mathbb{1}_{\mathbf{x}_i^{(1)} \in [x_1, x_1']} \right| \\
& + \left| \frac{\tau}{\sigma^2\left(\sigma^2+\tau \mathrm{N}_n(A_{H,2})\right)} \frac{\mathrm{N}_n(A_{H,2})-1}{\mathrm{N}_n(A_{R,1})} \mathrm{N}_n\left(\left\{\mathbf{x}_i^{(1)} \in [x_1, x_1']\right\} \times \left\{\mathbf{x}_i^{(2)} > x_2\right\}\right)\right| \qquad \text{(C.56)} \\
& \times \left| \frac{1}{\mathrm{N}_n\left(\left\{\mathbf{x}_i^{(1)} \in [x_1, x_1']\right\} \times \left\{\mathbf{x}_i^{(2)} > x_2\right\}\right)} \sum_{i:\mathbf{x}_i^{(2)}>x_2} (y_i - \bar{y}_{A_{H,2}})^2 \mathbb{1}_{\mathbf{x}_i^{(1)} \in [x_1, x_1']} \right| \\
= & A_{1,3,1} + A_{1,3,2}.
\end{aligned}
$$

Note that $\frac{\tau \mathrm{N}_n(A_{H,2})}{\sigma^2\left(\sigma^2+\tau \mathrm{N}_n(A_{H,2})\right)}$ is bounded by a constant M as n is large,

$$
\left| \frac{\tau}{\sigma^2\left(\sigma^2+\tau \mathrm{N}_n(A_{H,2})\right)} \frac{\mathrm{N}_n(A_{H,2})}{\mathrm{N}_n(A_{R,1})} \mathrm{N}_n\left(\left\{\mathbf{x}_i^{(1)} \in [x_1, x_1']\right\} \times \left\{\mathbf{x}_i^{(2)} > x_2\right\}\right)\right| \leq M \frac{\delta^2+\delta}{\delta^2-\sqrt{\delta}} \to 0. \quad \text{(C.57)}
$$

So we have $A_{1,3,1} \to 0$ if n is large and δ is small.

If $\mathrm{N}_n\left(\left\{\mathbf{x}_i^{(1)} \in [x_1, x_1']\right\} \times \left\{\mathbf{x}_i^{(2)} > x_2\right\}\right) < \sqrt{n}$,

$$
\left| \frac{1}{\mathrm{N}_n\left(\left\{\mathbf{x}_i^{(1)} \in [x_1, x_1']\right\} \times \left\{\mathbf{x}_i^{(2)} > x_2\right\}\right)} \sum_{l:\mathbf{x}_i^{(2)}>x_2} (y_i - \bar{y}_{A_{H,2}})^2 \mathbb{1}_{\mathbf{x}_i^{(1)} \in [x_1, x_1']} \right| \leq \frac{C_\rho^2 \log(n)}{\sqrt{n}}. \quad \text{(C.58)}
$$

If $\mathrm{N}_n\left(\left\{\mathbf{x}_i^{(1)} \in [x_1, x_1']\right\} \times \left\{\mathbf{x}_i^{(2)} > x_2\right\}\right) > \sqrt{n}$, note that $|1-x_1| \geq \xi$, $\mathrm{N}_n(A_{R,1}) > \mathrm{N}_n(\xi) > (\xi - \delta^2)n$, $\mathrm{N}_n\left(\left\{\mathbf{x}_i^{(1)} \in [x_1, x_1']\right\} \times \left\{\mathbf{x}_i^{(2)} > x_2\right\}\right) \leq \mathrm{N}_n\left(\left\{\mathbf{x}_i^{(1)} \in [x_1, x_1']\right\}\right) \leq (\delta+\delta^2)n$. As a result

$$
\left| \frac{\mathrm{N}_n\left(\left\{\mathbf{x}_i^{(1)} \in [x_1, x_1']\right\} \times \left\{\mathbf{x}_i^{(2)} > x_2\right\}\right)}{\mathrm{N}_n(A_{R,1})} \right| \leq \frac{\delta-\delta^2}{\xi+\delta^2} \leq \frac{\delta}{\xi} \qquad \text{(C.59)}
$$

$$\left|\frac{\mathrm{N}_n\left(\left\{\mathbf{x}_i^{(1)} \in [x_1, x_1']\right\} \times \left\{\mathbf{x}_i^{(2)} > x_2\right\}\right)}{\mathrm{N}_n(A_{R,1})}\right|$$

$$\times \left|\frac{1}{\mathrm{N}_n\left(\left\{\mathbf{x}_i^{(1)} \in [x_1, x_1']\right\} \times \left\{\mathbf{x}_i^{(2)} > x_2\right\}\right)} \sum_{l:\mathbf{x}_i^{(2)} > x_2} (y_i - \bar{y}_{A_{H,2}})^2 \mathbb{1}_{\mathbf{x}_i^{(1)} \in [x_1, x_1']}\right| \tag{C.60}$$

$$\leq \frac{\delta}{\xi}\left(3(||m||_\infty + \alpha)^2 + ||m||_\infty^2 + \tilde{\sigma}^2 + 2||m||_\infty^2 \alpha\right).$$

Therefore $A_{1,3,2} \to 0$. Collecting all bounds, we conclude that $A_1 \to 0$. It is straightforward to prove with similar arguments that $B_1 \to 0$ and $L_n(d_1, d_2) - L_n(d_1', d_2') \to 0$.

REFERENCES

Albert, J. H. and Chib, S. (1993) Bayesian analysis of binary and polychotomous response data. *Journal of the American statistical Association*, **88**, 669–679.

Bartlett, P. L. and Traskin, M. (2007) Adaboost is consistent. *Journal of Machine Learning Research*, **8**, 2347–2368.

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

Birge, J. R., Candogan, O., Chen, H. and Saban, D. (2019a) Optimal commissions and subscriptions in networked markets. *Forthcoming in Manufacturing & Service Operations Management*.

Birge, J. R., Chen, H. and Keskin, N. B. (2019b) Markdown policies for demand learning with forward-looking customers. *Available at SSRN 3299819*.

Blei, D. M., Ng, A. Y. and Jordan, M. I. (2003) Latent dirichlet allocation. *Journal of machine Learning research*, **3**, 993–1022.

Breiman, L. (1996) Bagging predictors. *Machine learning*, **24**, 123–140.

— (1997) Arcing the edge. *Tech. rep.*, Statistics Department, University of California at Berkeley.

— (2001) Random forests. *Machine learning*, **45**, 5–32.

Breiman, L., Friedman, J., Olshen, R. and Stone, C. J. (1984) *Classification and regression trees*. Chapman and Hall/CRC.

Buntine, W. L. (1990) *A theory of learning classification rules*. Ph.D. thesis, Citeseer.

Cao, J. and Dowd, C. (2019) Estimation and inference for synthetic control methods with spillover effects. *arXiv preprint arXiv:1902.07343*.

Cao, J., Gu, C. and Wang, Y. (2020) Principal component and static factor analysis. In *Macroeconomic Forecasting in the Era of Big Data*, 229–266. Springer.

Cao, J. and Lu, S. (2019) Synthetic control inference for staggered adoption: Estimating the dynamic effects of board gender diversity policies. *arXiv preprint arXiv:1912.06320*.

Cao, J., Shi, X. and Shum, M. (2019) On the empirical content of the beckerian marriage model. *Economic Theory*, **67**, 349–362.

Chen, T. and Guestrin, C. (2016) XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 785–794. ACM.

Chipman, H., George, E. I., Gramacy, R. B. and McCulloch, R. (2013) Bayesian treed response surface models. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, **3**, 298–305.

Chipman, H. A., George, E. I. and McCulloch, R. E. (1998) Bayesian CART model search. *Journal of the American Statistical Association*, **93**, 935–948.

— (2007) Bayesian ensemble learning. In *Advances in neural information processing systems*, 265–272.

Chipman, H. A., George, E. I., McCulloch, R. E. et al. (2010) BART: Bayesian additive regression trees. *The Annals of Applied Statistics*, **4**, 266–298.

Chollet, F. et al. (2015) Keras.

Coram, M., Lalley, S. P. et al. (2006) Consistency of Bayes estimators of a binary regression function. *The Annals of Statistics*, **34**, 1233–1269.

Denison, D. G., Holmes, C. C., Mallick, B. K. and Smith, A. F. (2002) *Bayesian methods for nonlinear classification and regression*, vol. 386. John Wiley & Sons.

Denison, D. G., Mallick, B. K. and Smith, A. F. (1998) A Bayesian CART algorithm. *Biometrika*, **85**, 363–377.

Feng, G. and He, J. (2019) Factor investing: Hierarchical ensemble learning. *arXiv preprint arXiv:1902.01015*.

Feng, G., He, J. and Polson, N. G. (2018) Deep learning for predicting asset returns. *arXiv preprint arXiv:1804.09314*.

Feng, G., Polson, N., Wang, Y. and Xu, J. (2017) Sparse regularization in marketing and economics. *arXiv preprint arXiv:1709.00379*.

Feng, G., Polson, N. and Xu, J. (2016) The market for english premier league (epl) odds. *Journal of Quantitative Analysis in Sports*, **12**, 167–178.

— (2019) Deep learning in characteristics-sorted factor models. *Available at SSRN 3243683*.

Fernández-Delgado, M., Cernadas, E., Barro, S. and Amorim, D. (2014) Do we need hundreds of classifiers to solve real world classification problems? *The journal of machine learning research*, **15**, 3133–3181.

Freund, Y. (1995) Boosting a weak learning algorithm by majority. *Information and computation*, **121**, 256–285.

Freund, Y. and Schapire, R. E. (1997) A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of computer and system sciences*, **55**, 119–139.

Freund, Y., Schapire, R. E. et al. (1996) Experiments with a new boosting algorithm. In *icml*, vol. 96, 148–156. Citeseer.

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

— (2002) Stochastic gradient boosting. *Computational Statistics & Data Analysis*, **38**, 367–378.

Glynn, C., He, J., Polson, N. G. and Xu, J. (2019) Bayesian inference for polya inverse gamma models. *arXiv preprint arXiv:1905.12141*.

Gordon, L. and Olshen, R. A. (1980) Consistent nonparametric regression from recursive partitioning schemes. *Journal of Multivariate Analysis*, **10**, 611–627.

Gramacy, R. B. and Lee, H. K. H. (2008) Bayesian treed Gaussian process models with an application to computer modeling. *Journal of the American Statistical Association*, **103**, 1119–1130.

Guo, R., Cheng, L., Li, J., Hahn, P. R. and Liu, H. (2018) A survey of learning causality with data: Problems and methods. *arXiv preprint arXiv:1809.09337*.

Gupta, V., Yu, M. and Kolar, M. (2019a) Constrained high dimensional statistical inference. *arXiv preprint arXiv:1911.07319*.

— (2019b) Simultaneous inference for pairwise graphical models with generalized score matching. *arXiv preprint arXiv:1905.06261*.

Györfi, L., Kohler, M., Krzyzak, A. and Walk, H. (2006) *A distribution-free theory of nonparametric regression*. Springer Science & Business Media.

Hahn, P. R. and Carvalho, C. M. (2015) Decoupling shrinkage and selection in bayesian linear models: a posterior summary perspective. *Journal of the American Statistical Association*, **110**, 435–448.

Hahn, P. R., Carvalho, C. M. and Mukherjee, S. (2013) Partial factor modeling: predictor-dependent shrinkage for linear regression. *Journal of the American Statistical Association*, **108**, 999–1008.

Hahn, P. R., Carvalho, C. M., Puelz, D., He, J. et al. (2018a) Regularization and confounding in linear regression for treatment effect estimation. *Bayesian Analysis*, **13**, 163–182.

Hahn, P. R., He, J. and Lopes, H. (2018b) Bayesian factor model shrinkage for linear iv regression with many instruments. *Journal of Business & Economic Statistics*, **36**, 278–287.

Hahn, P. R., He, J. and Lopes, H. F. (2019) Efficient sampling for gaussian linear regression with arbitrary priors. *Journal of Computational and Graphical Statistics*, **28**, 142–154.

Hahn, P. R., Murray, J. S., Carvalho, C. M. et al. (2020) Bayesian regression tree models for causal inference: regularization, confounding, and heterogeneous effects. *Bayesian Analysis*.

Hastie, T., Tibshirani, R., Friedman, J. and Franklin, J. (2005) The elements of statistical learning: data mining, inference and prediction. *The Mathematical Intelligencer*, **27**, 83–85.

Hastie, T., Tibshirani, R. et al. (2000) Bayesian backfitting (with comments and a rejoinder by the authors. *Statistical Science*, **15**, 196–223.

Hazan, T., Papandreou, G. and Tarlow, D. (2016) *Perturbations, Optimization, and Statistics*. MIT Press.

He, J. and Hahn, P. R. (2020) Stochastic tree ensembles for regularized nonlinear regression. *arXiv preprint arXiv:2002.03375*.

He, J., Yalov, S. and Hahn, P. R. (2019) XBART: Accelerated Bayesian additive regression trees. In *The 22nd International Conference on Artificial Intelligence and Statistics*, 1130–1138.

Hill, J., Linero, A. and Murray, J. (2020) Bayesian additive regression trees: A review and look forward. *Annual Review of Statistics and Its Application*, **7**.

Hill, J. L. (2011) Bayesian nonparametric modeling for causal inference. *Journal of Computational and Graphical Statistics*, **20**, 217–240.

Ishwaran, H., Kogalur, U. B., Blackstone, E. H., Lauer, M. S. et al. (2008) Random survival forests. *The annals of applied statistics*, **2**, 841–860.

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

Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q. and Liu, T.-Y. (2017) LightGBM: A highly efficient gradient boosting decision tree. In *Advances in Neural Information Processing Systems*, 3146–3154.

Kindo, B. P., Wang, H. and Peña, E. A. (2016) Multinomial probit Bayesian additive regression trees. *Stat*, **5**, 119–131.

Kolar, M., Gupta, V. and Ming, Y. (2017) An influence-receptivity model for topic based information cascades. In *2017 IEEE International Conference on Data Mining (ICDM)*, 1141–1146. IEEE.

Kolar, M., Gupta, V. and Yu, M. (2016) Statistical inference for pairwise graphical models using score matching. In *Advances in Neural Information Processing Systems*, 2829–2837.

Kolar, M., Yu, M. and Gupta, V. (2020) Recovery of simultaneous low rank and two-way sparse coefficient matrices, a nonconvex approach. *Electronic Journal of Statistics*, **14**, 413–457.

Kuhn, M. et al. (2008) Building predictive models in R using the caret package. *Journal of statistical software*, **28**, 1–26.

Laurent, B. and Massart, P. (2000) Adaptive estimation of a quadratic functional by model selection. *Annals of Statistics*, 1302–1338.

Laurent, H. and Rivest, R. L. (1976) Constructing optimal binary decision trees is np-complete. *Information processing letters*, **5**, 15–17.

Lin, Y. and Jeon, Y. (2006) Random forests and adaptive nearest neighbors. *Journal of the American Statistical Association*, **101**, 578–590.

Linero, A. R. (2017) A review of tree-based Bayesian methods. *Communications for Statistical Applications and Methods*, **24**.

— (2018) Bayesian regression trees for high-dimensional prediction and variable selection. *Journal of the American Statistical Association*, **113**, 626–636.

Linero, A. R., Sinha, D. and Lipsitz, S. R. (2019) Semiparametric mixed-scale models using shared bayesian forests. *Biometrics*.

Linero, A. R. and Yang, Y. (2018) Bayesian regression tree ensembles that adapt to smoothness and sparsity. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, **80**, 1087–1110.

Logan, B. R., Sparapani, R., McCulloch, R. E. and Laud, P. W. (2019) Decision making and uncertainty quantification for individualized treatments using Bayesian additive regression trees. *Statistical methods in medical research*, **28**, 1079–1093.

Loh, W.-Y., Cao, L. and Zhou, P. (2019) Subgroup identification for precision medicine: A comparative review of 13 methods. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, **9**, e1326.

Loh, W.-Y. and Zhou, P. (2020) *The GUIDE approach to subgroup identification*. Springer.

Lu, Y. and Nikolaev, V. V. (2019) Expected loan loss provisioning: An empirical model. *Chicago Booth Research Paper*.

Mehta, M., Agrawal, R. and Rissanen, J. (1996) SLIQ: A fast scalable classifier for data mining. In *International Conference on Extending Database Technology*, 18–32. Springer.

Murray, J. S. (2017) Log-linear Bayesian additive regression trees for multinomial Logistic and count regression models. *arXiv preprint arXiv:1701.01503*.

Newton, M., Polson, N. G. and Xu, J. (2018) Weighted bayesian bootstrap for scalable bayes. *arXiv preprint arXiv:1803.04559*.

Nielsen, D. (2016) *Tree boosting with xgboost-why does xgboost win" every" machine learning competition?* Master's thesis, NTNU.

van der Pas, S. and Ročková, V. (2017) Bayesian dyadic trees and histograms for regression. In *Advances in Neural Information Processing Systems*, 2089–2099.

Pedro, D. (2000) A unified bias-variance decomposition and its applications. In *17th International Conference on Machine Learning*, 231–238.

Pratola, M. (2016) Efficent Metropolis-Hastings proposal mechanism for Bayesian regression tree models. *Bayesian Analysis*, **11**, 885–911.

Pratola, M., Chipman, H., George, E. and McCulloch, R. (2017) Heteroscedastic BART using multiplicative regression trees. *arXiv preprint arXiv:1709.07542*.

Pratola, M. T., Chipman, H. A., Gattiker, J. R., Higdon, D. M., McCulloch, R. and Rust, W. N. (2014) Parallel Bayesian additive regression trees. *Journal of Computational and Graphical Statistics*, **23**, 830–852.

Puelz, D., Hahn, P. R. and Carvalho, C. M. (2015) Sparse mean-variance portfolios: A penalized utility approach. *arXiv preprint arXiv:1512.02310*.

Puelz, D., Hahn, P. R., Carvalho, C. M. et al. (2017) Variable selection in seemingly unrelated regressions with random predictors. *Bayesian Analysis*, **12**, 969–989.

Richard Hahn, P., Carvalho, C. M. and Scott, J. G. (2012) A sparse factor analytic probit model for congressional voting patterns. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, **61**, 619–635.

Rocková, V. (2019) A note on semi-parametric Bernstein-von Mises theorems for BART priors.

Rocková, V. and van der Pas, S. (2017) Posterior concentration for Bayesian regression trees and forests. *arXiv preprint arXiv:1708.08734*.

Ročková, V. and Saha, E. (2019) On theory for BART. In *The 22nd International Conference on Artificial Intelligence and Statistics*, 2839–2848.

Schapire, R. E. (1990) The strength of weak learnability. *Machine learning*, **5**, 197–227.

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

Shalev-Shwartz, S. and Ben-David, S. (2014) *Understanding machine learning: From theory to algorithms*. Cambridge university press.

Si, Y. and Zhou, P. (2019) Bayes-raking: Bayesian finite population inference with known margins. *arXiv preprint arXiv:1901.02117*.

Starling, J. E., Murray, J. S., Carvalho, C. M., Bukowski, R. and Scott, J. G. (2018) Functional response regression with funBART: an analysis of patient-specific stillbirth risk. *arXiv preprint arXiv:1805.07656*.

Starling, J. E., Murray, J. S., Lohr, P. A., Aiken, A. R., Carvalho, C. M. and Scott, J. G. (2019) Targeted smooth bayesian causal forests: An analysis of heterogeneous treatment effects for simultaneous versus interval medical abortion regimens over gestation. *arXiv preprint arXiv:1905.09405*.

Valiant, L. G. (1984) A theory of the learnable. *Communications of the ACM*, **27**, 1134–1142.

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

Wang, M., Yu, M., Yang, Z. and Wang, Z. (2020) Provable q-iteration with l infinity guarantees and function approximation.

Wright, M. N. and Ziegler, A. (2015) ranger: A fast implementation of random forests for high dimensional data in C++ and R. *arXiv preprint arXiv:1508.04409*.

Yang, Z., Yu, M., Kolar, M. and Wang, Z. (2019) Convergent policy optimization for safe reinforcement learning. In *Advances in Neural Information Processing Systems*, 3121–3133.

Yeager, D. S., Hanselman, P., Walton, G. M., Murray, J. S., Crosnoe, R., Muller, C., Tipton, E., Schneider, B., Hulleman, C. S., Hinojosa, C. P. et al. (2019) A national experiment reveals where a growth mindset improves achievement. *Nature*, **573**, 364–369.

Yu, M., Gupta, V. and Kolar, M. (2017a) Estimation of a low-rank topic-based model for information cascades. *arXiv preprint arXiv:1709.01919*.

Yu, M., Kolar, M. and Gupta, V. (2018a) Learning influence-receptivity network structure with guarantee. *arXiv preprint arXiv:1806.05730*.

Yu, M., Thompson, A. M., Ramamurthy, K. N., Yang, E. and Lozano, A. C. (2017b) Multitask learning using task clustering with applications to predictive modeling and gwas of plant varieties. *arXiv preprint arXiv:1710.01788*.

Yu, M., Yang, Z., Zhao, T., Kolar, M. and Wang, Z. (2018b) Provable gaussian embedding with one observation. In *Advances in Neural Information Processing Systems*, 6764–6774.

Zhang, T., Yu, B. et al. (2005) Boosting with early stopping: Convergence and consistency. *The Annals of Statistics*, **33**, 1538–1579.

Zhou, Z.-H. (2012) *Ensemble methods: foundations and algorithms*. CRC press.