UNIVERSITY OF CALIFORNIA

Los Angeles

Utilization of Low Dimensional Structure to Improve the Performance of Nonparametric Estimation in High Dimensions

A dissertation submitted in partial satisfaction
of the requirements for the degree
Doctor of Philosophy in Biostatistics

by

Daniel Joshua Conn

2018

ABSTRACT OF THE DISSERTATION

Utilization of Low Dimensional Structure to Improve the Performance of Nonparametric Estimation in High Dimensions

by

Daniel Joshua Conn
Doctor of Philosophy in Biostatistics
University of California, Los Angeles, 2018
Professor Gang Li, Co-Chair
Professor Christina Michelle Ramirez, Co-Chair

When the number of covariates is small, nonparametric regression methods serve a number of useful purposes. In this setting, nonparametric regression methods often demonstrate better predictive performance than parametric models. This is because nonparametric methods have the virtue of being able to detect nonlinear structure and complex interactions. In settings where the sample size is small or the level of noise is high, it may be the case that parametric models outperform nonparametric methods. However, even in this setting, nonparametric methods can be useful for diagnosing problems of model misspecification. Unfortunately, when the number of covariates is large, the curse of dimensionality, in its many forms, renders many of the most commonly used nonparametric regression methods unstable and prone to overfitting. We have developed two methods that, in some sense, overcome the curse of dimensionality. Both methods implicitly assume the existence of lower dimensional structure. First, we have developed a variant of random forests, called fuzzy forests. Fuzzy forests reduce the bias observed in random forest variable importance measures by clustering covariates into distinct groups such that the correlation of covariates within a group is high and the correlation between groups is low. Fuzzy forests is expected to work well when the true regression function exhibits an additive structure. Second, we have extended a machine learning method called metric learning to right-censored survival outcomes. If the true regression function is multi-index, we have shown that a closely related metric learning estimator achieves a rate of convergence dependent on the number of indices rather than the number of covariates.

The dissertation of Daniel Joshua Conn is approved.

Beth D Jamieson-Karavodin

Hua Zhou

Robert Erin Weiss

Christina Michelle Ramirez, Committee Co-Chair

Gang Li, Committee Co-Chair

University of California, Los Angeles

2018

To my parents, Richard and Cathryn Conn, for their loving support.

Table of Contents

1 Introduction **1**

1.1 Introduction 1

2 Fuzzy Forests: A New Random Forest Based Variable Selection Method **4**

2.1 Introduction 4

2.2 Variable Importance Measures and the Fuzzy Forests Algorithm . 6

2.2.1 Motivation for Variable Importance Measures 6

2.2.2 An Introduction to Random Forests 9

2.2.3 A Brief Review of WGCNA 12

2.2.4 The Fuzzy Forests Algorithm 15

2.2.5 A Justification for the Fuzzy Forests Algorithm 18

2.3 The fuzzyforest Package 23

2.4 Simulations 26

2.4.1 Application 32

2.5 Discussion 34

3 Kernel Regression in High Dimensions **39**

3.1 Introduction 39

3.2 Main Results 42

3.2.1 Preliminaries 42

3.2.2 An Oracle Property of the Kernel Regression Estimator . . 46

3.3 Simulations 51

3.4 Real Data Examples 54

3.5 Discussion 55

3.6 Proofs 57

3.6.1 Definitions and Results Regarding Brackets 57

3.6.2 Rate of Convergence of a Kernel Regression Estimator with Unbounded Support 58

3.6.3 Proof of Theorem 1 60

3.6.4 Proof of Theorem 2 69

4 Metric Learning for Right Censored Survival Outcomes 73

4.1 Introduction 73

4.2 Extending Metric Learning to Right-Censored Survival Data . . . 75

4.2.1 Metric Learning 75

4.2.2 Metric Learning for Right-Censored Outcomes 78

4.2.3 Further Refinements 80

4.3 Simulations 81

4.4 Data Analysis 84

4.4.1 Serum Free Light Chain Data Set 87

4.4.2 Sample of Non-small-cell Lung Cancer Cases from the SEER Database 88

4.4.3 Time Until Weaning for Breast-fed Newborns 88

4.5 Discussion 89

5 Discussion 91

5.1 Further Extensions of Random Forests and Fuzzy Forests 91

5.1.1 Correcting the Bias in Fuzzy Forests VIMs 91

5.2 Future Work on Cross-Validation and Model Selection 95

5.2.1 Further Application of General Optimization Techniques to Minimizing the Cross-Validation Criterion 95

Appendices 97

A Further Technical Results for Chapter 3 98

A.1 Proof of Lemma 2 98

A.2 Derivation of Gradient for Kernel Regression 103

References 105

List of Figures

2.1 Flow chart of fuzzy forests algorithm 16

2.2 Importance of VIMs within each module. The height of the bars represents the proportion of features in each module. The proportion of each bar colored in red represents the proportion of features that are selected within each module. In this case, modules 0 and 11 contain most of the selected features. 27

2.3 Fuzzy forests are compared to random forests with $p = 100$ and $n = 100$. The height of each point represents the proportion of times each feature was selected in 100 simulations. $X_1, X_2, X_3, X_{76}, X_{77}$, and X_{78} are important features. All other features were not important. X_1, X_2, and X_3 are correlated features. X_4 is correlated with X_1, X_2, and X_3 but is not important. X_{79} is independent but is not important. X_1, X_2 X_{76}, and X_{77} all have the same importance. X_3 and X_{78}, equally important, have lower importance than the other important features. 29

2.4 Fuzzy forests are compared to random forests with $p = 1,000$ and $n = 100$. The height of each point represents the proportion of times each feature was selected in 100 simulations. $X_1, X_2, X_3, X_{901}, X_{902}$, and X_{903} are important features. All other features were not important. X_1, X_2, and X_3 are correlated features. X_4 is correlated with X_1, X_2, and X_3 but is not important. X_{904} is independent but is not important. X_1, X_2, X_{901}, and X_{902} all have the same importance. X_3 and X_{903}, equally important, have lower importance than the other important features. 30

2.5 Fuzzy forests are compared to random forests with $p = 100$ and $n = 250$. The height of each point represents the proportion of times each feature was selected in 100 simulations. $X_1, X_2, X_3, X_4, X_{76}, X_{77},$, X_{78}, and X_{79} are important features. All other features were not important. X_1, X_2, X_3, X_4 are correlated with one another. X_5 is correlated with X_1, X_2, X_3, X_4 but is not important. X_{80} is independent and not important. 31

2.6 Fuzzy forests are compared to random forests with $p = 100$ and $n = 500$. The height of each point represents the proportion of times each feature was selected in 100 simulations. $X_1, X_2, X_3, X_4, X_{76}, X_{77},$, X_{78}, and X_{79} are important features. All other features were not important. X_1, X_2, X_3, X_4 are correlated features. X_5 is correlated with X_1, X_2, X_3, X_4 but is not important. X_{80} is independent but is not important. 32

2.7 Plot to determine WGCNA tuning parameter. This plot shows that 8 is the smallest power such that the scale free topology criterion is approximately met. 36

2.8 Importance of VIMs within each module. The height of the bars represents the proportion of features in each module. The proportion of each bar colored in red represents the proportion of features that are selected as important within each module. 37

2.9 This plot displays the importance of the top 10 selected features after fitting fuzzy forests. The variables are ranked from top to bottom. Red features indicate that controllers have higher values for the feature compared to responders. Black features indicate that responders have lower values for the feature compared to controllers. 38

List of Tables

2.1 Popular VIM and feature selection (FS) methods 12

3.1 RMSEs and 95% CIs for kerned regression estimator with matrix bandwidth, scalar bandwidth, and EDR under a single index model (Index # =1) and a five-index model (Index # =5). 53

3.2 Estimated RMSEs for three commonly used data sets. BW=bandwidth matrix. 55

4.1 Simulation results for linear Weibull model, $n = 500$ 83

4.2 Simulation results for linear Weibull model, $n = 1,000$ 84

4.3 Simulation results for nonlinear model, n=500 85

4.4 Simulation results for nonlinear model, n=1,000 86

4.5 RMSE and Harrell's c-Index of methods for `flchain` data set. . . . 87

4.6 RMSE and Harrell's c-Index of methods for subset of SEER data set. 88

4.7 RMSE and Harrell's c-Index of methods for subset of data set on time until weaning for breast-fed newborns. 89

5.1 True VIMs for simulation study 94

5.2 Regression tree VIMs with splits along coordinate axes versus regression tree VIMs with splits along hyperplanes, $n = 500$ 94

5.3 Regression tree VIMs with splits along coordinate axes versus regression tree VIMs with splits along hyperplanes, $n = 1,000$ . . . 95

5.4 Regression tree VIMs with splits along coordinate axes versus regression tree VIMs with splits along hyperplanes, $n = 2,000$ . . . 96

Acknowledgments

I would like to offer my thanks to my advisors, Dr. Gang Li and Dr. Christina Ramirez, and my committee members, Dr. Beth Jamieson, Dr. Hua Zhou, and Dr. Robert Weiss for their guidance and generosity of spirit.

I would also like to thank my family and friends whose support has been invaluable to me. I love you all.

Vita

2010	B.A. Mathematics. Department of Mathematics, Columbia, New York, NY.
2011-2018	Doctoral Candidate (Biostatistics), University of California, Los Angeles, CA.

Publications

Ramirez, C. M., Sinclair, E., Epling, L., Lee, S. A., Jain, V., Hsue, P. Y., Hatano, H., Conn D., Hecht, F. M., J. N., and McCune, J. M. (2016), “Immunologic Profiles Distinguish Aviremic HIV-Infected Adults," *AIDS*, 30(10):1553-1562.

Conn, D. and Ramirez, C. M. (2016), “Random Forests and Fuzzy Forests in Biomedical Research." In Alvarez, R. M. (ed.), *Computational Social Science*, Analytical Methods for Social Research, Chapter 6, 166-195. New York: Cambridge University Press.

Conn, D., Ngun, T., Ramirez, C. M., and Li, G. (2018), “Fuzzy Forests: Extending Random Forest Feature Selection for Correlated, High-Dimensional Data," *Journal of Statistical Software* (Accepted, to appear in 2018).

CHAPTER 1

Introduction

1.1 Introduction

It is widely known that many popular statistical methods fail in high-dimensional settings. For example, in the context of multiple linear regression, if the number of covariates (p) is larger than the sample size (n) the least squares estimate of the regression coefficients fails to be unique. In fields such as genetics it is common to analyze data sets where the number of covariates is a magnitude larger than the sample size. This is the so-called $p >> n$ setting. The dimension of a statistical problem depends not only on the number of covariates and sample size; the dimension of a statistical problem also depends on the complexity of the proposed model. In the context of nonparametric regression, the true regression function cannot generally be determined by a finite number of parameters. Therefore, even if p is moderately sized ($p = 5$ or $p = 10$), the sample size required to accurately estimate the true regression function may be astronomical. A nonparametric regression model is high dimensional even if the sample size is much larger than the number of covariates. This degradation of performance for many nonparametric regression methods, even for moderately sized p, is one of many manifestations of the curse of dimensionality. The methods we have developed seek to overcome the curse of dimensionality by taking advantage of lower dimensional structure in the data (if such lower dimensional structure exists).

Random forests is a popular machine learning algorithm introduced in [1]. In addition to being renowned for its predictive performance, random forests provide nonparametric measures of variable importance. These variable importance measures (VIMs) estimate the extent to which the regression function depends on each covariate. These VIMs are calculated by assessing the decrease in predictive accuracy when the levels of a particular covariate are permuted across observations. VIMs can be used to rank the importance of covariates and to carry out variable selection. Unfortunately, it has been demonstrated that these VIMs can be quite biased in the presence of correlated covariates [2, 3, 4]. Because of this bias, random forest variable selection methods that utilize these VIMs will also be biased. Conditional variable importance measures, proposed in [2], have been shown to reduce the bias. However, these conditional VIMs have the drawback of being computationally infeasible when n and p are moderately large.

We have developed the fuzzy forests variable selection algorithm to reduce the bias observed in random forest variable selection [5]. In a simulation study, we compare fuzzy forests to variable selection carried out via random forest VIMs and conditional VIMs. We also present a fuzzy forests analysis with the goal of understanding which aspects of the immune system allow a certain subset of HIV infected patients to control the virus without taking anti-retroviral therapy (ART). Such patients are called elite controllers. In particular, we analyze a flow cytometry data set and determine which immunological phenotypes distinguish elite controllers from immunological responders (subjects on ART with undetectable levels of virus).

Our other contributions primarily concern a machine learning technique called metric learning which is closely related to the classical Nadaraya-Watson kernel regression estimator. Metric learning is a framework for regression and classification in which the data is used to estimate a distance metric that will lead to good predictive performance [6, 7, 8]. Once this distance metric is estimated it can be used in another algorithm such as kernel regression or k-nearest neighbors. In regression, observations that are close to one another according to this distance metric will have similar estimated means. The metric learning algorithm we study is an extension of kernel regression with a matrix-valued bandwidth parameter [9]. The matrix-valued bandwidth is chosen by using gradient descent to minimize the K-fold cross-validation criterion.

We have shown that if the true regression function is a multi-index regression model [10, 11, 12, 13] and if minimization of the K-fold cross-validation criterion is used to select the bandwidth matrix, the estimated regression function overcomes the curse of dimensionality in the sense that the convergence rate of the estimator depends on the index number rather than the number of covariates. This result relies on a more general theorem on the efficacy of K-fold cross-validation when the cross-validation criterion is minimized over a continuum of tuning parameters rather than a discrete grid of tuning parameters. This general result is important as it provides a theoretical basis for the use of general optimization techniques such as gradient descent to minimize the K-fold cross-validation criterion.

We then extend metric learning to right censored survival data by taking advantage of a synthetic variable transformation [14, 15] method and an iterative Buckley-James type imputation method [16].

In Chapter 2 the fuzzy forests algorithm is introduced and its performance is assessed via a simulation experiment. Chapter 3 presents our result on the performance of the Nadaraya-Watson regression estimator when K-fold cross-validation is used to select the bandwidth matrix. In Chapter 4 we introduce our extension of metric learning to right-censored survival analysis data. In Chapter 5 we summarize our contributions and outline potential areas of future research.

CHAPTER 2

Fuzzy Forests: A New Random Forest Based Variable Selection Method

2.1 Introduction

In the era of high-throughput technologies such as multi-color flow cytometry and next generation sequencing, high dimensional data has become increasingly common in biomedical research. However, the ability to generate data has vastly outpaced our ability to analyze it. In the biomedical sciences as well as the *Omics* fields it is common for the number of features (p) to be much larger than the number of observations (n), the so-called $p >> n$ problem. This problem is exacerbated by the fact that the features are often highly correlated and the correlation structure is often unknown a priori.

Identifying important features in this situation has been an area of intense research within the statistics and machine learning community. While model based feature selection algorithms such as the LASSO [17, 18, 19] or SCAD (smoothly clipped absolute deviation) [20, 21] may detect important features in the presence of correlation [22], this comes at the cost of making parametric assumptions that may not hold in practice.

Random forests are a popular ensemble machine learning algorithm. Random forests are nonparametric, nonlinear, embarrassingly parallelizable, easy to implement, and have been described as one of the best "off-the-shelf" classifiers [23]. Random forest variable importance measures (VIMs) offer a flexible alternative to model based feature selection algorithms [1]. While random forest VIMs have demonstrated the ability to accurately capture the true importance of features in settings where the features are independent, it is well known that random forest VIMs are biased when features are correlated with one another [24, 2, 3].

Fuzzy forests handle correlated features by taking a piecewise approach. We first estimate the correlation structure of the data and partition the set of features into distinct modules such that the correlation within each module is high and the correlation between modules is low. We then use recursive feature elimination random forests (RFE-RF) [25] to select the most important features from each module. The surviving features from each module are combined and one final RFE-RF is then applied, selecting and ranking the most important ones. The fact that fuzzy forests carry out separate feature selection algorithms on distinct groups of correlated covariates distinguishes it from other commonly used random forest based feature selection methods. We believe that fuzzy forests will be useful to a wide variety of researchers including those in biology, medicine, psychology, social sciences, and any application in which there is high dimensional data with correlation.

The general fuzzy forests algorithm allows for the use of a variety of methods for partitioning the features into distinct clusters. The **fuzzyforest** package allows the analyst to input their own clustering of the features. Commonly, such a partition of the features would be derived by considering the correlation matrix of the features.

The particular implementation of the fuzzy forests algorithm given in the **R** package **fuzzyforest** also gives the analyst the option of utilizing the functionality of Weighted Gene Coexpression Network Analysis via the package **WGCNA** [26] to partition covariates into distinct clusters. WGCNA is a rigorous framework for detecting correlation networks [27]. Although WGCNA has been used primarily in genetics, it has also been applied successfully in contexts such as brain imaging and cancer biology [26].

The conditional variable importance measures introduced in [2] have also been proposed as a means for reducing the bias in random forest VIMs. However, the calculation of conditional variable importance measures is computationally intensive. In this article, we compare feature selection from random forests, conditional inference forests, and fuzzy forests, using packages **randomForest**[28], **party** [29, 24, 2], and **fuzzyforest**, respectively. We find that fuzzy forests offer a computationally feasible alternative to conditional inference forests for feature selection in the presence of highly correlated features.

2.2 Variable Importance Measures and the Fuzzy Forests Algorithm

2.2.1 Motivation for Variable Importance Measures

In this section we introduce basic notation and discuss VIMs. The VIMs that we discuss in this section describe important aspects of the true regression function and are well-defined outside of the context of random forests. We assume that our data comes in the form of n independently and identically distributed (iid) pairs $(X, Y) \sim G_{(X,Y)}$. Here, X is a p dimensional feature vector, with vth element $X^{(v)}$, and Y is a scalar outcome. Let $X_i^{(v)}$ denote the value of the vth feature for the ith subject and let $X_i = (X_i^{(1)}, \ldots, X_i^{(p)})^\top$ be the feature vector for the ith subject. Finally, the distribution of X and the marginal distribution of $X^{(v)}$ are denoted as G_X and $G_{X^{(v)}}$, respectively.

In the case of regression, we are interested in modeling the conditional mean of Y given a feature vector X. We denote this conditional mean as $E[Y|X]$ or $f(X)$. We assume that $Y|X$ has distribution equal to that of $f(X) + \epsilon$, where Y is continuous and the ϵ are independent of X and iid with variance σ^2. In the case of regression, a prediction for a new observation X_{new} would be obtained by evaluating the conditional mean at X_{new}: $f(X_{new})$.

For binary classification, we are again interested in modeling the conditional mean of Y given a feature vector X, however, Y is restricted to take the value 0 or 1. Thus, $Y|X$ is a Bernoulli trial with mean $E[Y|X = x] = P(Y = 1|X = x)$. In the case of binary classification, the predicted outcome for a new observation would be 1 if $f(X_{new}) = P(Y = 1|X = X_{new}) > 0.5$, and 0 otherwise. See Chapters 2 and 5 of [30] and the Chapter 1 of [31] for justification and criticism of the use of this rule for prediction. Random forests are also able to handle the case of multi-class classification [1].

For both classification and regression, we say that feature $X^{(v)}$ is unimportant if $E[Y|X]$ does not depend on $X^{(v)}$. The problem of feature selection requires more than a “black box" estimate of $f(X)$. It requires an understanding of how $f(X)$ depends on each individual feature.

If p is low dimensional ($p = 1, 2$), we can simply plot our estimate of $f(X)$ to understand how it varies as a function of X. On the other hand, if p is moderate or large, the estimate of $f(X)$ may be difficult to interpret. This problem of interpretability may be alleviated by assuming $f(X)$ has a specific parametric form such that $f_\gamma(X)$ is known up to a finite dimensional parameter γ. In the case of linear regression, where $f_\gamma(X_i) = \gamma_0 + \sum_{v=1}^{p} \gamma_v X_i^{(v)}$, γ is a vector of regression coefficients and we may measure the importance of one feature versus another by examining the absolute magnitude of their corresponding coefficients (assuming the features have all been standardized).

A chosen parametric model, $f_\gamma(X)$, is often intended as a decent parametric approximation to $f(X)$. Unfortunately, this parametric approximation may fail to capture salient characteristics of $f(X)$ for a variety of reasons. Notably, $f_\gamma(X)$ might miss important interactions between features, or, in the case of the above linear regression model, the true $f(X)$ may be nonlinear in such a way that the best linear approximation fails to capture. In contrast, random forests are nonlinear and nonparametric. Therefore, the resulting random forest VIMs, defined below, naturally take interactions and nonlinear structure into account.

With large sample sizes, nonparametric regression methods are more likely to detect nonlinearities and interactions, however with smaller sample sizes or greater levels of noise, linear models can easily outperform nonparametric models [32]. In practice, nonparametric methods and parametric methods complement one another. For example, nonparametric methods can be used to diagnose incorrect parametric modeling assumptions (see Chapter 6 of [33], Chapter 4 of [34], and Chapter 17 of [35]).

We would like to end this section with a discussion of the general goals of feature selection and how they relate to estimation of VIMs. The ultimate objective of fuzzy forests is to select a small subset of features such that the selected features will have relatively high VIM in comparison with the rest of the features. Another potential goal of feature selection is to select a subset of features with the ultimate goal of predicting outcomes for new observations. In this latter case, feature selection might be advisable as a means of improving predictive capabilities (some predictive algorithms may be adversely effected by the presence of unimportant features). Using feature selection with the goal of prediction may also be useful if it is advantageous to reduce the number of measurements taken on each individual.

In the presence of correlation, feature selection methods such as fuzzy forests that are designed to select features with the highest VIMs may yield different results than feature selection methods designed to optimize predictive accuracy. For example, suppose two features are highly correlated and only one feature is important while the other is not. If the goal is maximizing predictive accuracy, either feature may be selected without adversely effecting predictive ability. The two features effectively serve as proxies for one another.

2.2.2 An Introduction to Random Forests

The random forests algorithm is a popular ensemble method that has been applied in the settings of both classification and regression [1]. The random forests algorithm works by combining the predictions of an ensemble of classification or regression trees. Each tree is grown on a separate bootstrap sample of the data. The number of trees grown in this manner is denoted as NTREE. The subjects that are not selected in a particular bootstrap sample are said to be "out of bag." Roughly one third of subjects will be out of bag for each tree. These out of bag subjects play the important role of serving as a validation set for each tree, allowing the user to obtain estimates of the prediction error that are not overly optimistic.

Call the kth tree $\hat{f}_k(X)$. In the case of regression trees, $\hat{f}(X) = \frac{1}{\text{NTREE}} \sum_{k=1}^{\text{NTREE}} \hat{f}_k(X)$. In the case of classification, $\hat{f}(X)$ is the majority vote of the *ntree* predictions given by $\hat{f}_k(X)$. Each tree, by itself, may be highly unstable, leading to highly variable estimates of $f(X)$, however, by averaging multiple trees over many bootstrap samples, the variance of our estimate for $f(X)$ may be significantly reduced. The algorithm described thus far is known as bagging (bootstrap-aggregating). This algorithm is a special case of random forests.

A further element of randomness is introduced by random forests. Before a node in a particular tree is split, a subset of features is chosen at random. The best splitting rule, derived from only these randomly selected features, is then used to split the node. The number of randomly selected features at each stage is commonly called MTRY. If MTRY $= p$, then random forests are equivalent to bagging. High values of MTRY tend to lead to just a few important features getting selected at the majority of nodes. Lower values of MTRY allow more features to play a role in the estimation of $f(X)$. In the case of regression, a common default value of MTRY is $\lfloor p/3 \rfloor$ and, in the case of classification, $\sqrt{p}$ is a common choice

[28].

Multiple random forest VIMs have been developed. In this article we will exclusively focus on unscaled random forest permutation VIMs. Random forest permutation VIMs are obtained by testing how predictive accuracy suffers when the values of an individual feature are randomly permuted.

For example, suppose a particular feature is important in determining the value of the outcome. Randomly permuting the values of this feature destroys its relationship with the outcome. Because the connection between this particular feature and the outcome has been obscured, there should be a subsequent decrease in predictive accuracy when predictions are made using this permuted data. If there was no relationship to begin with, the predictive accuracy obtained using the permuted data should be comparable to the predictive accuracy obtained using the original, unpermuted, data. The random forest permutation VIM measures the average decline in predictive performance for each feature across multiple trees.

An advantage of the permutation VIM is that it is understood what parameter the permutation VIM is estimating. Thus, the permutation VIM leads to a formal definition of "importance" given by Equation 5.1.

We now describe the calculation of the random forest permutation VIM for the vth feature. Let $OOB_k \subset \{1, \ldots, n\}$ be the indices for the out of bag sample from the kth tree and let $|OOB_k|$ be the number of out of bag samples. Let $\pi_k = (\pi_{k1}, \ldots, \pi_{kn})$ be a random permutation of OOB_k and let $\tilde{X}_i = (X_i^{(1)}, \ldots, X_{\pi_{ki}}^{(v)}, \ldots, X_i^{(p)})^\top$ be the feature vector for the ith subject where the vth feature has been permuted. In the case of regression, the variable importance of the vth feature from the kth tree is defined as

$$\widehat{VIM}_k(v) = \frac{\sum_{i \in OOB_k} (y_i - \hat{f}_k(\tilde{X}_i))^2 - (y_i - \hat{f}_k(X_i))^2}{|OOB_k|} \tag{2.1}$$

The random forest permutation VIM for the vth feature is defined as

$$\widehat{VIM}(v) = \frac{\sum_{k=1}^{\text{NTREE}} \widehat{VIM}_k(v)}{ntree} \tag{2.2}$$

We note that a number of other VIMs are in common use. The **randomForest** package implements two types of VIMs. **randomForest** implements the permutation based VIM discussed above. It also implements a VIM based on the mean decrease in "impurity" in the child nodes after splitting a node on a particular feature. The measure of impurity will depends on whether classification or regression trees are being used. For example, in the case of regression, the within-node variance is a measure of impurity. For classification, the Gini-index is the default measure of node impurity.

In the package **party**, an additional VIM, called the conditional VIM is implemented. The conditional VIM, developed in [2], has been shown to reduce the bias in random forest VIMs, however, calculation of the conditional VIM is computationally quite expensive, particularly when the sample size is large.

We summarize a number of VIMs and feature selection methods in Table 2.1 below. There is a distinction between VIMs and feature selection methods. Calculation of VIMs alone does not immediately lead to a unique set of features to select, however VIMs may play the central role in a feature selection procedure. For example, calculating random forest VIMs and keeping the VIMs that rank in the top 5% defines a feature selection procedure. The fuzzy forests algorithm is a more complex feature selection procedure that relies on the calculation of VIMs.

[4] presents a clear discussion of the nature and source of bias in random forest permutation VIMs. In their article, [4] conducts a simulation study in which the true model is linear with a group of positively correlated important features and

Method	Type	Nonparametric	**R** Package
Permutation Importance	VIM	T	**randomForest/party**
Mean Decrease in Node Impurity	VIM	T	**randomForest**
Conditional VIM	VIM	T	**party**
Fuzzy Forests	FS	T	**fuzzyforest**
LASSO	FS and VIM	F	**glmnet**
SCAD	FS and VIM	F	**ncvreg**

Table 2.1: Popular VIM and feature selection (FS) methods

a group of independent important features. They find in this simulation study that permutation VIMs favor the group of correlated features.

It is worth noting that the correlation of features alone does not guarantee heavily biased permutation VIMs. For example, as demonstrated in [4], in a null model, a model in which none of the features are important for the outcome, the resulting permutation VIMs are not particularly biased. The simulations of [4] suggest that correlation of features will induce bias in the VIMs if the correlation structure induces marginal correlations that do not reflect the importance of the features.

2.2.3 A Brief Review of WGCNA

In genetics, statistical network models play a significant role in uncovering important regulatory mechanisms or processes. Weighted genetic co-expression network analysis (WGCNA), first developed to detect networks of highly correlated genes, has seen great success in many biological applications. The **R** package **WGCNA** is a robust and well-documented implementation of the WGCNA framework [36, 26] that was originally designed to detect correlation networks in the context of genetics. WGCNA has been used extensively outside of the context of gene expression data. For example, it has seen use in the analysis of fMRI data [37]. We believe that WGCNA has fairly wide applicability as, at its core, it relies on an application of hierarchical clustering methods to functions of the correlation matrix.

We expect that researchers already familiar with the **WGCNA** package will easily adopt the fuzzy forests algorithm and we expect that newcomers to **WGCNA** will be able to make good use of **WGCNA**'s fine documentation and tutorials. WGCNA takes in the matrix of features and uses the correlation structure to partition the features into distinct groups such that the correlation between features in the same group is large and the correlation between features in separate groups is small. In the context of WGCNA, these groups of features are called modules. WGCNA constructs a network of features, each feature representing one node, via the correlation matrix of features. It determines modules based off of this network.

Formally, the user first specifies a similarity matrix with (u, v)th entry $s_{uv} = S(X^{(u)}, X^{(v)})$ for features u and v. The function $S(X^{(u)}, X^{(v)})$ is called the similarity function and often takes values between 0 and 1. Common similarity functions include $|\widehat{Corr}(X^{(u)}, X^{(v)})|$ and $(1 + \widehat{Corr}(X^{(u)}, X^{(v)}))/2$ [27], where $\widehat{Corr}(X^{(u)}, X^{(v)}))$ is the sample correlation between features u and v. The former choice of similarity function leads to an unsigned network where features have high similarity score if they have strong positive or negative correlation with one another. The latter choice of similarity function leads to a signed network where features are deemed most similar if they have strong positive correlation and are deemed highly dissimilar if they have strong negative correlation. A signed network takes into account the sign of the correlation.

This similarity matrix is then transformed into an adjacency matrix $A = [a_{uv}]$ via an adjacency function $a_{uv} = \alpha(s_{uv})$. The adjacency function determines how similarities translate into properties of the network. The hard threshold function, denoted by $signum(s_{uv}, \tau)$, where τ is defined to be the threshold, is the simplest choice of adjacency function: if $s_{uv} \geq \tau$ then $a_{uv} = signum(s_{uv}, \tau) = 1$, otherwise $a_{uv} = 0$. Nodes are either classified as connected or unconnected. In practice, a soft-thresholded network is often more plausible than a hard-thresholded one. The power function $a_{uv} = s_{uv}^{\beta}$ is a common choice of soft-thresholding adjacency function. Large values of β yield behavior closer to a hard-thresholded network. Setting $\beta = 1$ is equivalent to using the similarity function. Once an adjacency function is calculated, a hierarchical clustering tree algorithm is used to define the clusters of features.

It is common to apply this hierarchical clustering algorithm to the topological overlap matrix rather than the adjacency matrix. The topological overlap between two nodes is defined as

$$\omega_{uv} = \frac{q_{uv} + a_{uv}}{min\{c_u, c_v\} + 1 - a_{uv}} \tag{2.3}$$

where $q_{uv} = \sum_{r=1}^{p} a_{ur}a_{rv}$ and $c_u = \sum_{r=1}^{p} a_{ur}$ is the connectivity of the uth feature [38] . The topological overlap between two nodes can be high even if a_{uv} is low. This occurs when the two nodes are strongly connected to the same set of nodes. Use of the topological overlap matrix rather than the adjacency matrix may lead to more distinct modules [27].

In many biological contexts, it is suspected that only a few features are highly connected to many other features. This prior knowledge leads to the scale-free criterion for determining which value of β to select. Let $r(c_u)$ be resulting density function from fitting a histogram, with equal size bin widths, to the observed connectivities. We call $r(c_u)$ the frequency function of the connectivities. A network is said to have generalized scale-free topology if $r(c_u) \propto c_u^{\beta}$ [27], where β is a non-negative real number and $\propto$ is the symbol for “proportional to". If the scale-free topology criterion is suspected to hold, one should select a value of β such that the R^2 between $\log_{10}(r(c_u))$ and $\log_{10}(c_u)$ is high.

2.2.4 The Fuzzy Forests Algorithm

The fuzzy forests algorithm is an extension of random forests designed to obtain less biased feature selection in the presence of correlated features. In this section, we describe the algorithm. First we give a summary of the procedure.

In the first step of fuzzy forests, the features are partitioned into distinct groups or modules, such that the correlation of features within modules is high and the correlation of features between modules is low. Our package, **fuzzyforest**, facilitates the use of **WGCNA** to determine the modules although it is possible to use alternative methods to partition the features. Once features have been subdivided into distinct modules, fuzzy forests eliminates features in two steps: a screening step and a selection step. In the screening step, RFE-RF is used on each module to eliminate the least important features within each module. In the selection step, a final RFE-RF is used on the surviving features.

A detailed explication of RFE-RF is given below. RFE-RF sequentially eliminates features with the lowest VIMs until a pre-specified percentage of features remain. By sequentially eliminating the least important features, RFE-RF is able to better focus on determining which features are the most important.

The screening step of fuzzy forests achieves two goals. First, it reduces the number of features that have to be analyzed at one time. Second, the bias caused by correlated features is alleviated. In [3], it is observed that unimportant features that are correlated with an important feature are more likely to be chosen at the root of tree than uncorrelated important features. The high importance of these unimportant correlated features comes at the cost of the important uncorrelated features. When we analyze each module separately, features in different groups are no longer competing against one another.

In biological applications, modules might represent different biological components or demographic information about the subjects. By carrying out RFE-RF

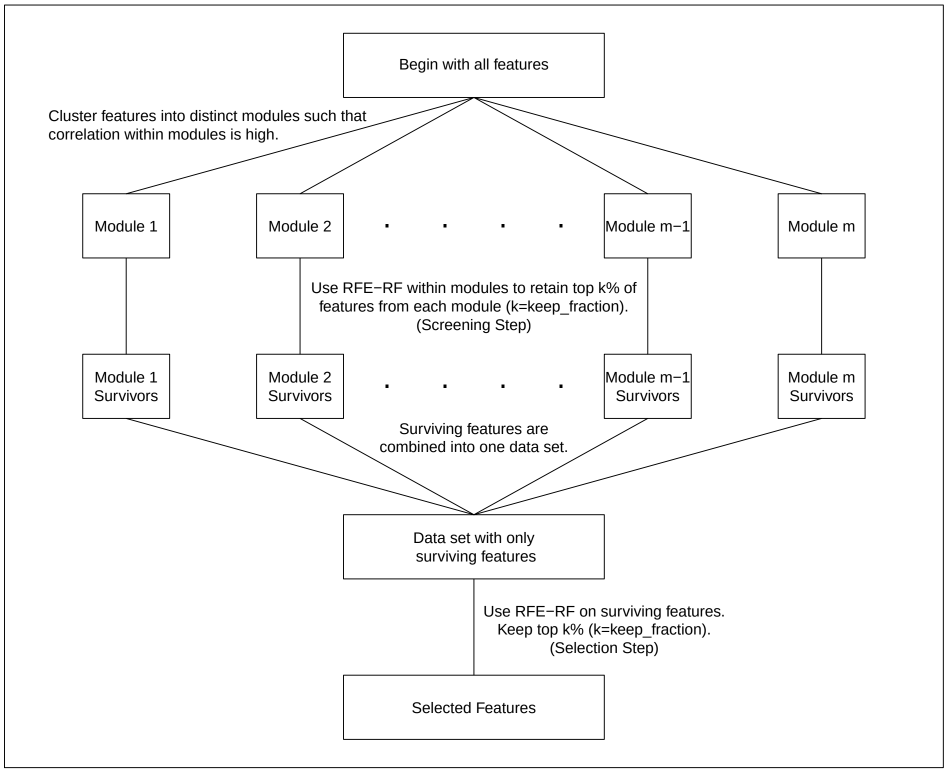
Begin with all features
Cluster features into distinct modules such that correlation within modules is high.
Module 1
Module 2
Module m−1
Module m
Use RFE−RF within modules to retain top k% of features from each module (k=keep_fraction). (Screening Step)
Module 1 Survivors
Module 2 Survivors
Module m−1 Survivors
Module m Survivors
Surviving features are combined into one data set.
Data set with only surviving features
Use RFE−RF on surviving features. Keep top k% (k=keep_fraction). (Selection Step)
Selected Features

Figure 2.1: Flow chart of fuzzy forests algorithm

on the features that survived the screening step, the selection step effectively allows these systems to interact with one another.

A flow chart of the fuzzy forests algorithm is given in Figure 2.1.

We now provide a detailed description of the screening step and RFE-RF. Denote the set of modules by $P = \{P_1, \ldots, P_m\}$. Let $p_l = |P_l|$ so that $\sum_{l=1}^{m} p_l = p$, where m is the number of modules. For each element of the partition, P_l, RFE-RF is used to screen out unimportant or less important features.

We now describe the RFE-RF procedure in the context of screening features in a particular partition P_l. At the start of the procedure, a random forest is fit using all of the features in P_l and the least important features are eliminated. For example, the features with VIM in the bottom 25% might be dropped. Call the reduced set of features in P_l, after this first random forest, $P_l^{(1)}$. A second random forest is then fit using only features in $P_l^{(1)}$. The least important features from this latest random forest are then eliminated leading to a further reduced set of features $P_l^{(2)} \subset P_l^{(1)} \subset P_l$. The subset obtained after iteration t is denoted as $P_l^{(t)}$ and let $p_l^{(t)}$ be the number of features in $P_l^{(t)}$. Features are eliminated in this manner until a user-specified stopping criteria is reached. For example, features may be eliminated until 5% of the original features in P_l remain.

The user must specify a few tuning parameters at the screening step. First, the user must specify what fraction of features are to be dropped after each step of the RFE-RF. We call this fraction the DROP_FRACTION. The user must also specify a stopping criteria. In **fuzzyforest** the user specifies what fraction of the original p_l features, in each module P_l, to retain. This fraction is called the KEEP_FRACTION. The first time the number of features drops below KEEP_FRACTION $* p_l$, the RFE-RF stops and the top $\lfloor \text{KEEP\_FRACTION} * p_l \rfloor$ features are selected. More precisely, for the first iteration t such that $p_l^{(t)} <$ KEEP_FRACTION $* p_l$, we retain the top $\lfloor \text{KEEP\_FRACTION} * p_l \rfloor$ features from $P_l^{(t-1)}$.

For each RFE-RF, MTRY and NTREE must be appropriately selected. Since the number of features varies across the forests, MTRY and NTREE must be a function of the current number of features. Suppose we are at iteration t and are about to fit a random forest to obtain $P_l^{(t+1)} \subset P_l^{(t)}$, in the case of regression, **fuzzyforest** sets MTRY $= \left\lfloor p_l^{(t)} * \text{MTRY\_FACTOR}/3 \right\rfloor$. For classification, **fuzzyforest** sets MTRY $= \left\lfloor \sqrt{p_l^{(t)} * \text{MTRY\_FACTOR}} \right\rfloor$. In both cases, MTRY_FACTOR must be specified by the user, with the default being 1. The parameter NTREE must be set high enough to be able to pick up the effects of important variables, however if NTREE is set too high, the iterative series of random forests takes longer to fit. The package **fuzzyforest** sets NTREE $= \max(\text{MIN\_NTREE}, \left\lfloor p_l^{(t)} * \text{NTREE\_FACTOR} \right\rfloor)$, where

MIN_NTREE is a minimal number of trees grown for each forest and NTREE_FACTOR allows the number of trees to increase with the number of features.

The final step consists of one last RFE-RF to allow for interactions between features in different modules. Note that a separate choice of DROP_FRACTION, MTRY_FACTOR, MIN_TREE, and NTREE_FACTOR may be used for the final selection step. The user specifies how many features to keep in the final selection step. If certain features are, a priori, known to be important (perhaps demographic characteristics), **fuzzyforest** allows the user to let these features skip the initial round of screening.

Finally, we compare the RFE-RF procedure described above to the RFE-RF procedure described in [25] and implemented in the package **varSelRF** [39]. In the procedure presented in [25], [25] prefer that the VIMs not be recalculated at each iteration, with the intent of preventing overfitting. In the RFE-RF procedure described above, permutation VIMs are calculated at each iteration. This is because we wanted to allow the ranking of VIMs to change as unimportant features are dropped. The ultimate focus of fuzzy forests is to select features with the highest ranking VIMs. We do note that **varSelRF** does allow for the option of recalculating VIMs at each iteration and an RFE-RF procedure in which VIMs were calculated at each iteration was proposed by [40]. Classification as opposed to regression also appears to be the primary focus of the RFE-RF procedure in **varSelRF**.

2.2.5 A Justification for the Fuzzy Forests Algorithm

The screening of features within distinct modules is motivated by the following heuristic observations concerning the theoretical properties of VIMs. As noted previously, correlation between features can cause bias because the correlation structure can induce high marginal correlation between features and the outcome that do not reflect the importance of the features. This is particularly problematic when important features are correlated with one another. In this case, the marginal correlation between these features and the outcome will be greatly amplified by the correlation, causing the permutation VIMs to ignore or underrate the independent and important features.

Intuitively, dividing features into distinct, correlated modules and carrying out feature selection within each module provides an advantage because the correlation structure within a module is relatively uniform. Although the correlation within a module is high, the uniform nature of the correlation structure is not expected to lead to particularly misleading VIMs. Importantly, feature selection on the independent features is unaffected by feature selection on the correlated features. In the following discussion, we formally define the parameter being estimated by permutation VIMs and we discuss conditions under which the VIMs calculated within a module are equivalent to VIMs calculated using the full set of features.

The estimation of VIMs is formally investigated by [41]. The random forest VIM is discussed in [42] and [43]. Intuitively, the random forest VIM of the vth feature measures how much $f(X_i)$ changes when the vth entry of X_i, $X_i^{(v)}$, is replaced by an independent realization, $\tilde{X}_i^{(v)}$, generated with distribution $G_{X^{(v)}}$. Formally, the random forest permutation VIM of feature v estimates the following parameter

$$VIM(v) = E(f(X_i^{(1)}, \ldots, X_i^{(v)}, \ldots, X_i^{(p)}) - f(X_i^{(1)}, \ldots, \tilde{X}_i^{(v)}, \ldots, X_i^{(p)}))^2. \quad (2.4)$$

First, note that the expression is the same for all choices of index, i, because the (X_i, Y_i) are iid with distribution $G_{(X,Y)}$. Next note that f is fixed and the expectation is with respect to the random variables $X_i = (X_i^{(1)}, \ldots, X_i^{(v)}, \ldots, X_i^{(p)})$ and $\tilde{X}_i^{(v)}$. The random vector X_i has distribution G_X and $\tilde{X}_i^{(v)}$, generated independently of X_i, has distribution $G_{X^{(v)}}$. If the value of $f(X_i)$ changes greatly when

$X_i^{(v)}$ is replaced by $\tilde{X}_i^{(v)}$, it implies that the vth feature is important. In the case where $f_\gamma(X) = \gamma_0 + \sum_{v=1}^{p} \gamma_v X^{(v)}$ is a linear model, with standardized features ($\text{Var}(X_i^{(v)}) = 1$), $\text{VIM}(v) = 2\gamma_v^2$.

Let $G_{P^{(l)}}$ denote the joint distribution of the features in the module $P^{(l)}$ and let $X^{P^{(l)}} \sim G_{P^{(l)}}$. In general, the conditional expectation, $E[A|B]$, of one random variable A with respect to another random variable, B, is defined as the function $h(B)$ that minimizes $E[(A-h(B))^2]$ or, written more compactly, $\text{argmin}_h E[(A - h(B))^2]$. When random forests are fit using only the features in module $P^{(l)}$, the estimated regression function converges to

$$\begin{aligned}
\text{argmin}_h E[(Y - h(X^{P^{(l)}}))^2] &= \text{argmin}_h E[(f(X) + \epsilon - h(X^{P^{(l)}}))^2] && (2.5)\\
&= \text{argmin}_h E[\epsilon^2 + 2\epsilon(f(X) - h(X^{P^{(l)}})) + (f(X) - h(X^{P^{(l)}}))^2] && (2.6)\\
&= \text{argmin}_h E[2\epsilon(f(X) - h(X^{P^{(l)}})) + (f(X) - h(X^{P^{(l)}}))^2] && (2.7)\\
&= \text{argmin}_h E[(f(X) - h(X^{P^{(l)}}))^2] && (2.8)\\
&= E[f(X)|X^{P^{(l)}}]. && (2.9)
\end{aligned}$$

Note that $E[2\epsilon(f(X) - h(X^{P^{(l)}}))] = 0$ because ϵ is independent of X and has mean 0.

Assume that features in separate modules $X^{P^{(1)}}, \ldots, X^{P^{(m)}}$ are independent and suppose that $f(X) = \sum_{j=1}^{m} f_j(X^{P^{(j)}})$. The form of the regression function, $f(X)$, allows for interactions within modules but no interactions between modules. We now demonstrate that if we fit random forests using only the features in $P^{(l)}$, we are no longer estimating $E[Y|X] = f(X)$, instead we are estimating

$$E[f(X)|X^{P^{(l)}}] = \sum_{j=1}^{m} E[f_j(X^{P^{(j)}})|X^{P^{(l)}}] = f_l(X^{P^{(l)}}) + \sum_{j \neq l}^{m} E_{X_{P^{(l)}}}[f_j(X^{P^{(j)}})]. \tag{2.10}$$

As a result, the VIMs obtained by fitting a separate random forest to each module $P^{(l)}$, are equal to the VIMs obtained by fitting a random forest using the full set of features.

This is seen by the following argument

$$E[Y|X^{P^{(l)}}] = \text{argmin}_h E[(Y - h(X^{P^{(l)}}))^2] \tag{2.11}$$

$$= \text{argmin}_h E[\{(Y - f(X)) - (h(X^{P^{(l)}}) - f(X))\}^2]. \tag{2.12}$$

This last term equals:

$$\text{argmin}_h \{E[(Y-f(X))^2] - 2E[(Y-f(X))(h(X^{P^{(l)}})-f(X))] + E[(h(X^{P^{(l)}})-f(X))^2]\}. \tag{2.13}$$

Now, the first of the above expectations does not depend on h. The second expectation equals 0:

$$E[(Y - f(X))(h(X^{P^{(l)}}) - f(X))] = E[E[(Y - f(X))(h(X^{P^{(l)}}) - f(X))|X]] \tag{2.14}$$

$$= E[(h(X^{P^{(l)}}) - f(X))E[(Y - f(X))|X]] \tag{2.15}$$

$$= 0. \tag{2.16}$$

This leaves only the third expectation remaining. Thus, $E[Y|X^{P^{(l)}}] = \text{argmin}_h E[(h(X^{P^{(l)}}) - f(X))^2]$. By the definition of conditional expectation, this last term equals $E[f(X)|X^{P^{(l)}}]$. Note that by the independence of the modules, we have $E[f_j(X^{P^{(j)}})|X^{P^{(l)}}] =$

$E_{X_{P^{(l)}}}[f_j(X^{P^{(j)}})]$ for all $j \neq l$. This yields Equation 2.10.

Suppose feature v is in partition $P^{(l)}$, the VIM obtained by fitting a random forest to only those features in $P^{(l)}$ is estimating the following quantity:

$$VIM^*(v) = E(f_l(X_i^{(l_1)}, \ldots, X_i^{(v)}, \ldots, X_i^{(l_m)}) - f_l(X_i^{(l_1)}, \ldots, \tilde{X}_i^{(v)}, \ldots, X_i^{(l_m)}))^2. \tag{2.17}$$

Here, $X_i^{l_k}$ is the kth element of partition $P^{(l)}$. As in Equation 5.1, $X_i^{(v)}$ and $\tilde{X}_i^{(v)}$ are iid from $G_{X^{(v)}}$. We see from this equation that $VIM^*(v) = VIM(v)$ if the true regression function is additive across modules and if the modules are independent of each other. If our assumptions are met, the VIMs obtained by analyzing each module separately are asymptotically the same as those that would have been obtained if VIMs were obtained by analyzing all features at once.

These observations suggest that if we assume strict additivity and independence of the modules, then obtaining VIMs from each module separately should suffice. However, if these assumptions are not met, the VIMs obtained by analyzing each module separately are, in general, different than the VIMs obtained by fitting a single random forest on all of the features at once. We stress that the above derivation depends on the additivity assumption and the assumption of independence of modules.

If there are interactions between features in different modules, the VIMs calculated within modules will be asymptotically biased. However, under the circumstances that the most important VIMs in each module are also the features that are most likely to be heavily involved in interactions between modules, carrying out feature selection on each module separately should still allow for the selection of important features.

The final RFE-RF, applied at the selection step serves to relax this restrictive additivity assumption, allowing for interactions between features that were found to be important within modules. However, it is important to note that when features from separate modules are combined, the potential for bias due to correlation between features is reintroduced. Thus, the estimated VIMs may still be biased and must be interpreted with caution.

While the implicit assumptions underlying fuzzy forests are strict, we point out that random forests, as well as conditional inference forests may also demonstrate bias. In the case of random forests, as discussed above, the simulation results of [2] and [4] suggest that random forests will be unbiased if the marginal correlations between features and the outcome largely reflect the true VIMs. We believe that this is an even more stringent assumption than the assumptions made in Section 2.5. We believe that fuzzy forests will have less biased feature selection properties than random forests because the conditions under which random forest feature selection is roughly unbiased are even more stringent than fuzzy forests. The simulations carried out below also demonstrate that feature selection using conditional permutation VIMs can also demonstrate bias.

2.3 The fuzzyforest package

The package **fuzzyforest** has two functions for fitting fuzzy forests. The first is `wff`, the second is `ff`. The function `wff` automatically carries out a WGCNA analysis on the features. Then it uses these newly derived modules as input to fuzzy forests. The WGCNA analysis is carried out via the `blockwiseModules` function, from the package **WGCNA**.

The second function `ff` assumes that the features have already been partitioned into separate modules. For example, it may be advantageous to use hierarchical clustering directly on the correlation matrix and to cut the tree by visual inspection via calls to `hclust` and `cutree` in the **stats** package. This procedure may give the user more flexibility in which distance metric to use and in how to cluster the features. The package **pvclust** calculates p-values to assess the uncertainty in clusters of features and can be used to find a stable clustering of the features. Another common use case for `ff` is to carry out the fuzzy forests algorithm using the output of **WGCNA**, thereby allowing more customization of options for **WGCNA**.

A number of tuning parameters must be specified before fuzzy forests can be run. These tuning parameters are organized into separate control objects. Tuning parameters related to **WGCNA** are specified with an object of type `WGCNA_control`. Similarly, tuning parameters related to the screening step and the selection step are specified through objects of type `screen_control` and `select_control`.

We demonstrate the workings of **fuzzyforest** with an analysis of a data set concerning gene expression in liver tissue in female mice. The data set can be found in the tutorial website for **WGCNA**:
http://labs.genetics.ucla.edu/horvath/CoexpressionNetwork/Rpackages/WGCNA/Tutorials/.
The number of mice is 131 and the number of genes is 3,600. We examine how the expression of these genes correlates with the weight(g) of the mice. In the following code, the data set is called `Liver_Expr`.

```
R> weight <- Liver_Expr[, 1]
R> expression_levels <- Liver_Expr[, -1]
```

We first use **WGCNA** to select the power that leads to a network with approximately scale-free topology. We set $\beta = 6$ (β is equivalent to `power` in the code below) and set other tuning parameters for **WGCNA** in the following call. Note that the resulting number of modules can be sensitive to `minModuleSize`.

```
R> WGCNA_params <- WGCNA_control(power = 6, TOMType = "unsigned",
+    minModuleSize = 30, numericLabels = TRUE, pamRespectsDendro = FALSE)
```

Then we set tuning parameters for the selection step and the screening step:

```
R> mtry_factor <- 1; drop_fraction <- .25; number_selected <- 10
R> keep_fraction <- .05; min_ntree <- 5000; ntree_factor <- 5
R> final_ntree <- 5000;
R> screen_params <- screen_control(drop_fraction = drop_fraction,
+    keep_fraction = keep_fraction, min_ntree = min_ntree,
+    mtry_factor = mtry_factor, ntree_factor = ntree_factor)
> select_params <- select_control(drop_fraction = drop_fraction,
+   number_selected = number_selected, min_ntree = min_ntree,
+   mtry_factor = mtry_factor, ntree_factor = ntree_factor)
```

Finally, we use `wff` to fit fuzzy forests to the data set.

```
R> wff_fit <- wff(expression_levels, weight, WGCNA_params=WGCNA_params,
+    screen_params = screen_params, select_params = select_params,
+    final_ntree = final_ntree, num_processors = 4)
```

The function `wff` returns an object of type `fuzzy_forest`. Objects of type `fuzzy_forest` have the usual generic methods. The function `print` prints the selected features as well as module memberships. The function `predict(fuzzy_forest, new_data)` takes in a `data.frame` or `matrix` and produces predictions based on the selected features. The generic `predict` method for fuzzy forests produces a vector of predicted values for the set of observations in the data set used to fit fuzzy forests or, if a new, independent data set is provided as the value of the argument `new_data`, predicted values for observations in the new data set. Although the fuzzy forests algorithm was designed with feature selection in mind, it is possible to fit random forests using the selected features and to use the resulting model for prediction.

A `data.frame` with the selected features can be obtained by accessing the element `feature_list` from the `fuzzy_forest` object.

```
> wff_fit$feature_list
```

```
   feature_name variable_importance module_membership
1   MMT00026944              5.4857                 6
2   MMT00019254              4.8648                 6
3   MMT00067823              3.5857                 3
4   MMT00006001              3.5310                 3
5   MMT00074983              3.1431                 7
6   MMT00061313              2.3732                 3
7   MMT00070342              2.3454                 3
8   MMT00065159              2.3377                 3
9   MMT00078732              2.2109                 6
10  MMT00067261              2.1083                 3
```

Before the analysis is run, the user selects the desired number of important features as the end output of fuzzy forests. The number of features selected can be thought of as a tuning parameter. The predictive accuracy on a validation set can then be used to determine the optimal number of features to select.

As it is often useful to ascertain which modules are contributors to the signal of the outcome, we create a visual representation of all modules and the distribution of important features across the modules. The function `modplot` yields a visual display of which modules are important. The height of the bars represents what percentage of the total p features fall into a particular module. The area of the bar which is red represents the percentage of each module that is selected by fuzzy forests. Applying the function `modplot` to the object `wff_fit` above, we obtain the graph in Figure 2.2.

2.4 Simulations

In this section we demonstrate the performance of fuzzy forests in a number of simulation scenarios. These simulations are designed to compare fuzzy forests to

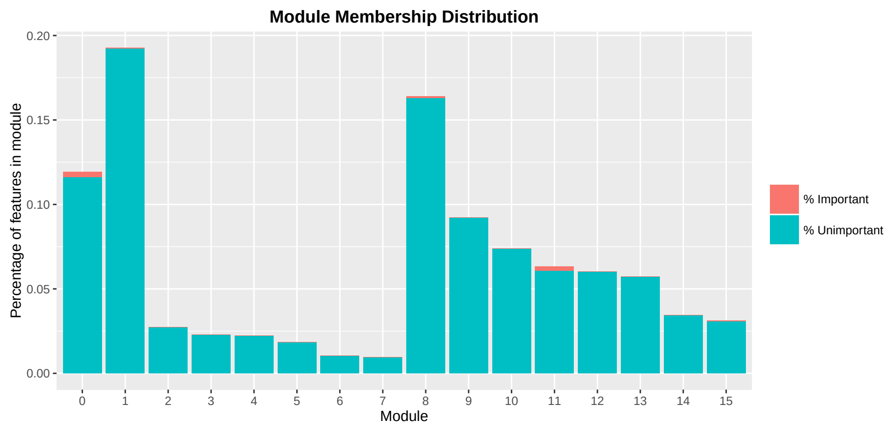

Figure 2.2: Importance of VIMs within each module. The height of the bars represents the proportion of features in each module. The proportion of each bar colored in red represents the proportion of features that are selected within each module. In this case, modules 0 and 11 contain most of the selected features.

random forests and conditional inference forests when the features are correlated. We carry out two simulations. In the first simulation, data is generated from a linear model. In the second simulation, data is generated from a nonlinear model. For random forests, feature selection is carried out by selecting the features with top 10 permutation VIMs. For conditional inference forests, feature selection is carried out by selecting the features with top 10 conditional VIMs. The first simulation is closely related to the simulations given in [4] and [2], the key distinction being that the simulation below includes additional “noise" features that are unrelated to the outcome.

In all simulations, X_i is generated from a multivariate normal distribution. The error terms, ϵ_i, are normal with mean 0 and standard deviation 0.5. The marginal distribution of each feature is standard normal. Features are subdivided into distinct modules. The correlation between features in different modules is 0. If the features in a module are correlated with one another, the correlation between features within the same module is set to 0.8. In each simulation, features in the final module are independent of each other (i.e., with correlation 0). Thus, within modules, the covariance structure is compound symmetry. In fitting fuzzy forests, we do not assume the modules are known. We use WGCNA to estimate the modules.

For the simulation from a linear model, we carry out two simulation scenarios. For the first scenario, the number of parameters p is set to 100. In this scenario, there are 4 modules. The features in the 4th module are independent of each other and of the features in the other modules. The features in the other three modules are correlated with one another. Namely, $\{X^{(1)}, \ldots, X^{(25)}\}$, $\{X^{(26)}, \ldots, X^{(50)}\}$, and $\{X^{(51)}, \ldots, X^{(75)}\}$ constitute 3 distinct modules each containing 25 features. The final module is $\{X^{(76)}, \ldots, X^{(100)}\}$. In this scenario, $Y_i = X_i^\top \gamma + \epsilon_i$ and among the correlated features we have $\gamma_1 = \gamma_2 = 5$ and $\gamma_3 = 2$. Among the group of independent features, $\gamma_{76} = \gamma_{77} = 5$ and $\gamma_{78} = 2$. All other elements of γ are set to 0. In addition, the intercept term, γ_0, is set to 0.

To evaluate the feature selection performance, we compute the proportion of times the non-zero features were selected over 100 simulation runs. The results are displayed in Figure 2.3.

For random forests and conditional inference forests the results of this simulation are largely in line with the results from [4] and [2]. Random forests is much less likely to select independent covariates than conditional inference forests. Fuzzy forests select important features with slightly lower frequency than conditional inference forests, however its performance is generally comparable.

In the second scenario for the linear model, we have the same setup as before except we increase p to 1,000 while leaving n at 100. The group of correlated features now contains 900 features, grouped into the following modules: $\{X^{(1)}, \ldots, X^{(100)}\}, \ldots, \{X^{(801)}, \ldots, X^{(900)}\}$. Again, the correlation between features in the same module is 0.8. The correlation of features from different modules is 0. The remaining module, $\{X^{(901)}, \ldots, X^{(1{,}000)}\}$, consists of independent features. Once again, $\gamma_1 = \gamma_2 = 5$ and $\gamma_3 = 2$. The first 3 independent features are also non-zero: $\gamma_{901} = \gamma_{902} = 5$ and $\gamma_{903} = 2$. As seen in Figure 2.4, when

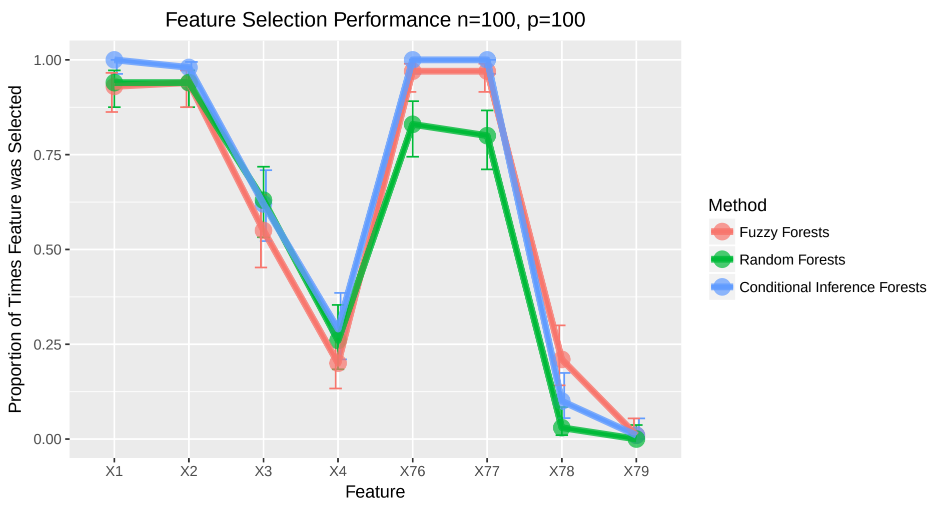

Figure 2.3: Fuzzy forests are compared to random forests with $p = 100$ and $n = 100$. The height of each point represents the proportion of times each feature was selected in 100 simulations. $X_1, X_2, X_3, X_{76}, X_{77}$, and X_{78} are important features. All other features were not important. X_1, X_2, and X_3 are correlated features. X_4 is correlated with X_1, X_2, and X_3 but is not important. X_{79} is independent but is not important. X_1, X_2 X_{76}, and X_{77} all have the same importance. X_3 and X_{78}, equally important, have lower importance than the other important features.

$p = 1,000$, random forests permutation VIMs largely ignore the independent features.

For the second simulation in which data was generated from a nonlinear model, we set $p = 100$ and let n vary from 250 to 500. The correlation structure in this simulation is identical to correlation structure described above for the linear simulation with $p = 100$. The true regression model, given in the equation below, was designed such that the true VIM for each of the features upon which f depends are approximately equal to 30 (the true VIMs were calculated via Monte Carlo simulations). Therefore all of the features should be selected with equal probability.

$$f(X) = X_1+X_2+2.92X_1X_2+\sqrt{15}X_3+X_4^3+X_{76}+X_{77}+3.74X_{76}X_{77}+\sqrt{15}X_{78}+X_{79}^3 \tag{2.18}$$

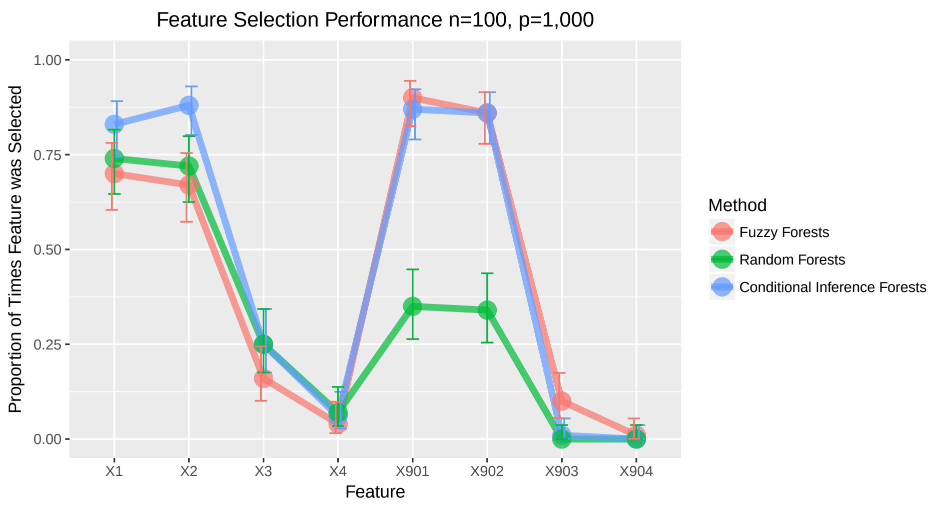

Figure 2.4: Fuzzy forests are compared to random forests with $p = 1,000$ and $n = 100$. The height of each point represents the proportion of times each feature was selected in 100 simulations. $X_1, X_2, X_3, X_{901}, X_{902}$, and X_{903} are important features. All other features were not important. X_1, X_2, and X_3 are correlated features. X_4 is correlated with X_1, X_2, and X_3 but is not important. X_{904} is independent but is not important. X_1, X_2, X_{901}, and X_{902} all have the same importance. X_3 and X_{903}, equally important, have lower importance than the other important features.

For the nonlinear scenario with $n = 250$, we were able to compute VIMs for random forests and fuzzy forests, as well as conditional VIMs. For the scenario with $n = 500$, we were unable to compute conditional VIMs as the computational burden was too great. In our experience, the computational burden of calculating conditional VIMs increases more quickly with the sample size as opposed to the number of covariates.

The results of the first nonlinear scenario are displayed in Figure 2.5. First of all, note that none of the methods select features with equal probability, even within modules. In general, the features that are part of interaction terms (X_1, X_2, X_{76}, and X_{77}), are chosen with lower probability than the other 4 important features. All of these tree-based method have more difficulty detecting interactions in comparison to the linear and cubic terms, even conditional inference forests.

As in the first nonlinear simulation scenario, the correlated features are favored over the independent features. In particular, although both X_5 and X_{80} have VIM of 0, X_5 is selected with higher probability. Random forests are most heavily biased in favor of the correlated features and were largely unable to detect the interacting features X_{76} and X_{77}. Fuzzy forests perform slightly worse than both random forests and conditional inference forest on the correlated features, however, they perform comparably to conditional inference forests on the independent features. Overall, conditional inference forests seem to yield the best performance.

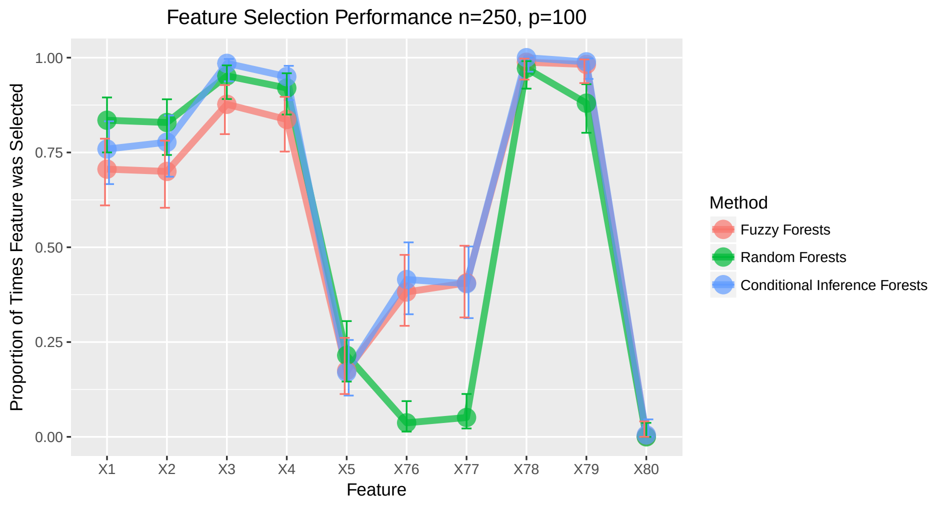

Figure 2.5: Fuzzy forests are compared to random forests with $p = 100$ and $n = 250$. The height of each point represents the proportion of times each feature was selected in 100 simulations. $X_1, X_2, X_3, X_4, X_{76}, X_{77},$, X_{78}, and X_{79} are important features. All other features were not important. X_1, X_2, X_3, X_4 are correlated with one another. X_5 is correlated with X_1, X_2, X_3, X_4 but is not important. X_{80} is independent and not important.

The results of the second scenario with $n = 500$ are displayed in Figure 2.6. As previously mentioned, calculation of conditional VIMs are computationally too burdensome. In this scenario, both random forests and fuzzy forests are able to select the correlated interacting features with higher probability (fuzzy forests, with smaller probability). Fuzzy forests also improve in its ability to select the independent interacting features with $n = 500$, while random forests are still largely unable to select these features.

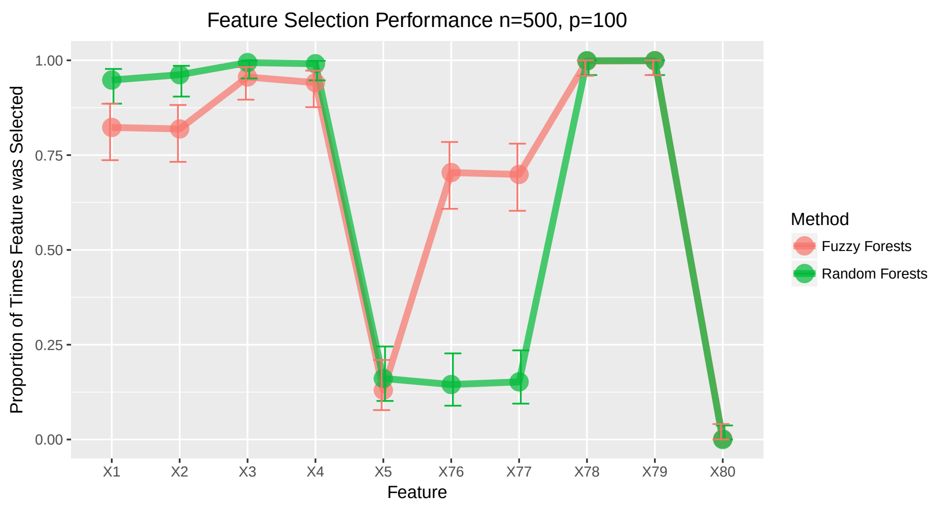

Figure 2.6: Fuzzy forests are compared to random forests with $p = 100$ and $n = 500$. The height of each point represents the proportion of times each feature was selected in 100 simulations. $X_1, X_2, X_3, X_4, X_{76}, X_{77},$, X_{78}, and X_{79} are important features. All other features were not important. X_1, X_2, X_3, X_4 are correlated features. X_5 is correlated with X_1, X_2, X_3, X_4 but is not important. X_{80} is independent but is not important.

2.4.1 Application

We demonstrate an application of **fuzzyforest** by using it to discover immunologic profiles that predict if an HIV patient will be able to control the virus without antiretroviral therapy (ART). An immunologic controller is defined as a patient able to achieve undetectable levels of the virus (< 50 copies/ml) without ART. Similarly, an immunologic responder is an aviremic patient, on ART, with sustained undetectable levels of the virus and CD4+ T cell counts above 350 cells/mm^3.

In this dataset there were 125 immunologic responders, 92 controllers ($n = 217$), and 313 features ($p = 313$). The features are derived from flow cytometry measurements. Flow cytometry may be used to measure the presence of various markers on the surface of a cell. The presence of up to 14 cell surface markers was measured. This yields up to 2^{14} possible binary combinations of markers, however, not all of these combinations were available. These markers assess immunological factors such as T cell maturation, activation, dysfunction, senescence, antigen-specificity and proliferation.

Features derived from flow cytometry measurements typically describe what proportion of cells in a sample display a subset of the aforementioned 14 markers. The presence of a cell surface marker is denoted by " $+$ " and the absence of the marker is denoted by " $-$." For example, one feature may measure the proportion of all lymphocytes in a sample that are CD4 positive (CD4+). A second feature may measure the proportion of lymphocytes that display both CD4 and CD38 (CD4+CD38+). Because the group of CD4+CD38+ T cells is nested within the group of CD4+ T cells, the proportion of lymphocytes that are CD4+ will be positively correlated the proportion of lymphocytes that are CD4+CD38+. The nested nature of different subgroups of lymphocytes leads to high levels of correlation between features.

For some markers, mean florescence intensity, a continuous measure of the extent to which a cell displays a particular marker, was also measured using flow cytometry.

We used **WGCNA** to partition the features into modules. We used the scale-free topology criterion to determine the power of the adjacency function. We set $\beta = 8$ based on the elbow of the curves in Figure 2.7. We found 11 modules. Each module is identified with a color. The choice of color is chosen randomly with the exception of the grey module. The grey module consists of features that are independent of the other modules. In our analysis, the largest module was the grey module with 140 features. It is commonly the case that the grey module is larger than the other modules. The smallest module, purple, was of size 10.

We used the resulting module memberships as input to the function `ff`. Because of the small size of the modules we set `keep_fraction` to 0.25. We tested multiple values for `number_selected`. The ranking of features was robust to settings of this parameter. We display the results when selecting 10 features.

The strongest predictors of virologic control without ART were HIV GAG-specific response and immune activation; see Figure 2.9. The immune systems of the controllers are highly reactive to proteins specific to HIV, i.e., gag. The selection of cell surface markers such as PD-1 suggests that controllers may have a higher percentage of T cells that may be dysfunctional. It is notable that, while controllers had overall higher levels of immune activation [44], they had lower activation in CD4+ central memory cells [45], as seen in the feature ranked 4th. These results are consistent with the nature of HIV pathogenesis. Indeed, it has been shown that limited infection of the central memory compartment is associated with lack of disease progression even in individuals who have detectable viremia [46].

2.5 Discussion

In this article we have presented the fuzzy forests algorithm as an extension of random forests that can provide less biased feature selection in the presence of correlation between features in a computationally feasible manner, especially when $p >> n$. Under these conditions, fuzzy forests are expected to outperform random forests. We found that, as expected, random forest VIMs were biased in favor of correlated features. Indeed when $p = 1,000$ while $n = 100$, random forests essentially ignored the independent variables that were important in the true model whereas fuzzy forests found them. The fuzzy forests algorithm is useful for screening large numbers of features or when it is desirable to find the most important features contributing to the signal.

We introduced an implementation of fuzzy forests in the **fuzzyforest** package. The **fuzzyforest** package has two functions for fitting the fuzzy forests algorithm. The first implementation, `wff` automatically carries out a WGCNA analysis to partition the features into separate modules. These modules are then used by fuzzy forests for feature selection. The second implementation, `ff` lets the user determine how features should be partitioned before fuzzy forests is used for feature selection.

We then used fuzzy forests to investigate immunologic phenotypes of patients who can control the virus without antiretrovirals. The set of important features was stable with respect to $mtry\_factor$ and other tuning parameters. The set of features found by fuzzy forests is biologically plausible and in part confirms findings from in vivo and other clinical studies, suggesting that fuzzy forests found the true underlying signal. It is expected that fuzzy forests will be useful in a wide variety of applications from gene studies, to flow cytometry to other studies where the data has high correlation and many potential predictors.

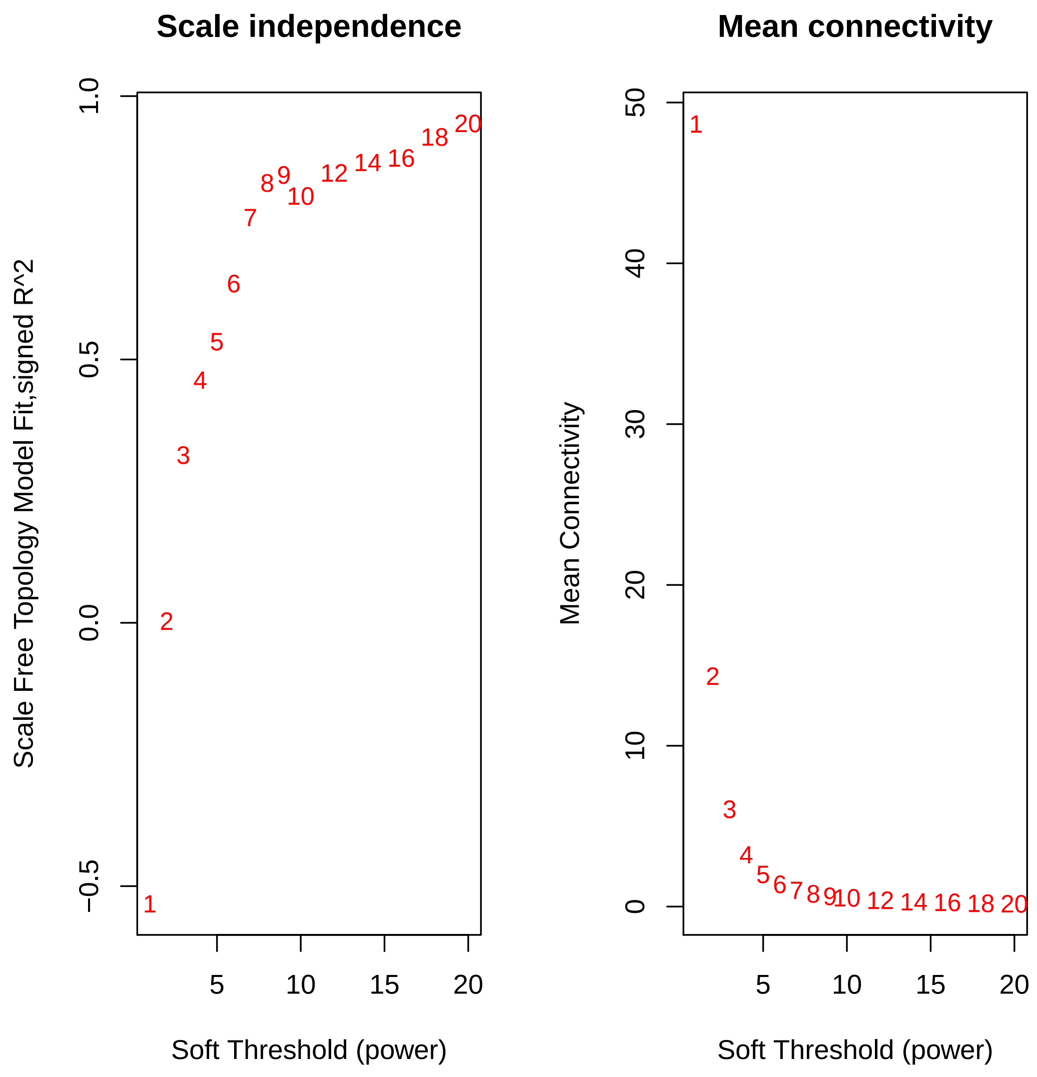

Figure 2.7: Plot to determine WGCNA tuning parameter. This plot shows that 8 is the smallest power such that the scale free topology criterion is approximately met.

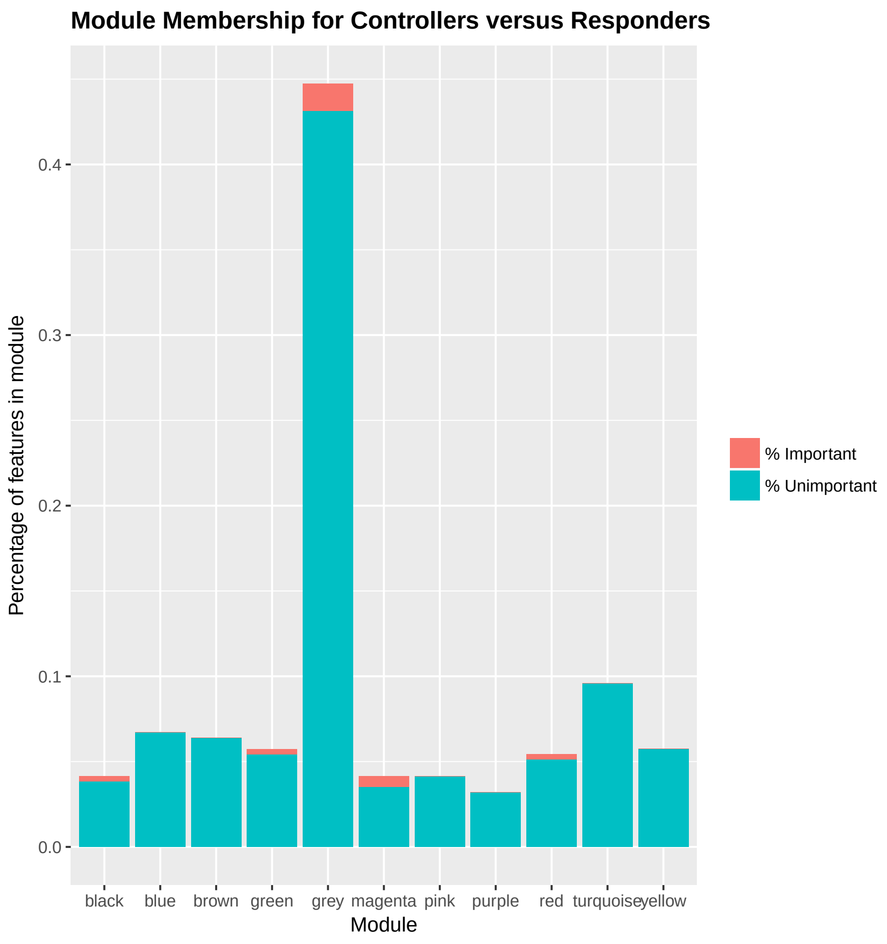

Figure 2.8: Importance of VIMs within each module. The height of the bars represents the proportion of features in each module. The proportion of each bar colored in red represents the proportion of features that are selected as important within each module.

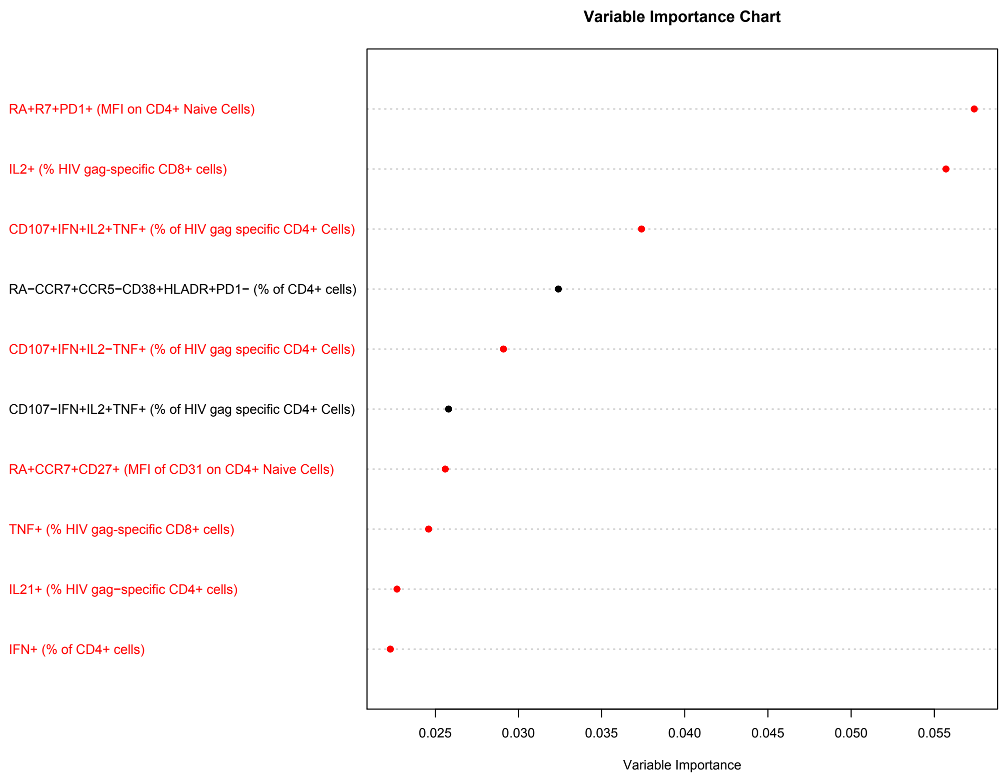

Figure 2.9: This plot displays the importance of the top 10 selected features after fitting fuzzy forests. The variables are ranked from top to bottom. Red features indicate that controllers have higher values for the feature compared to responders. Black features indicate that responders have lower values for the feature compared to controllers.

CHAPTER 3

Kernel Regression in High Dimensions

3.1 Introduction

The Nadaraya-Watson kernel regression estimator [47, 48] is a corner stone of nonparametric regression with widespread applications. Assume that one observes n independent and identically (iid) distributed random variables of the form $O_i = (X_i, Y_i), \sim P_0$, $i = 1, \ldots, n$, where $X_i \in \mathcal{R}^p$ are p-dimensional covariates and Y_i is a real-valued outcome. The Nadaraya-Watson kernel estimator of the multivariate regression function $\psi(x) \equiv E[Y|X = x]$ is commonly defined as

$$\psi_{H^*}(x) = \frac{\sum_{i=1}^{n} K(H^{*-1/2}(X_i - x))Y_i}{\sum_{i=1}^{n} K(H^{*-1/2}(X_i - x))}, \tag{3.1}$$

where H^* is a $p \times p$ symmetric positive definite matrix depending on n, and $K(u)$ is a kernel function such that $\int K(u)du = 1$. It is well-known that the performance of nonparametric regression methods degrades as the number of covariates, p, increases. This degradation in performance is often referred to as the "curse of dimensionality." For kernel regression, the curse of dimensionality can be seen to manifest itself when $H^* = h_n^* I_p$ by considering results such as Theorem 5.2 of [49] concerning the rate of the convergence of the kernel regression estimator ψ_{n,h_n^*} to ψ:

$$E \int (\psi_{n,h_n^*}(x) - \psi(x))^2 dP_{0,X}(x) = O(n^{-2/(p+2)}), \tag{3.2}$$

for an appropriately chosen sequence of scalar bandwidths, h_n^*. Also see the discussion at the end of Chapter 4 in [50]. We highlight that the above result is established under the commonly made assumption that the bandwidth tends to zero at a certain rate as n grows to infinity, which allows the kernel estimator to pick up local features and balance the trade-off between its bias and variance.

Although the Nadaraya-Watson kernel regression estimator has been well studied in the literature, its theoretical properties have been mostly derived for a general regression function $\psi(x)$ and require the bandwidth to go to zero along all p dimensions as n goes to infinity [51, 52, 53, 54, 55, 49]. To the best of our knowledge, very little is known about its theoretical behavior when there is an embedded low-rank structure in $\psi(x)$. The primary purpose of this article is to fill this theoretical gap by showing that as a fully nonparametric estimator, the Nadaraya-Watson kernel regression estimator indexed by a bandwidth matrix has an oracle property if $\psi(x)$ is multi-index and the bandwidths are allowed to diverge to infinity. Specifically, a multi-index model assumes that $\psi(x) = \phi(Tx)$ where T is a linear orthonormal function from $\mathcal{R}^p$ to $\mathcal{R}^m$ and ϕ is a map from $\mathcal{R}^m$ to $\mathcal{R}$, which is referred to as a single-index model when $m = 1$. We consider the following reparametrized form of the Nadaraya-Watson kernel estimator

$$\psi_H(x) = \frac{\sum_{i=1}^n K(H^{1/2}(X_i - x))Y_i}{\sum_{i=1}^n K(H^{1/2}(X_i - x))}. \tag{3.3}$$

Note that (3.3) is equivalent to the classical definition (3.1) with $H^* = H^{-1}$ if H is positive definite, but we will allow H to be positive semidefinite. Letting H be less than full rank, theoretically, allows the above estimator to take advantage of the low-dimensional structure in a multi index model. We will show in Theorem 2 that if the true regression function is a single or multi-index regression model with index number m, then K-fold cross-validation can be used to produce an estimator with rate of convergence that depends on m rather than p. Consequently, the Nadaraya-

Watson kernel regression estimator no longer suffers the curse of dimensionality when there exists an embedded low-rank structure in $\psi(x)$. If m is much less than p, the gain in predictive performance may be substantial.

Our theoretical result on the oracle property of the Nadaraya-Watson kernel estimator relies on an extension of an oracle inequality concerning sample splitting and K-fold cross-validation presented in [56] and [49], who considered sample splitting or cross-validation for selecting the best model from a discrete collection of models. In practice, for kernel regression with a bandwidth matrix, specifying a discrete set of bandwidth matrices over which the K-fold cross-validation criterion is to be minimized is likely to be overly burdensome and can quickly become formidable as the dimension grows. Our result allows for optimization of the K-fold cross-validation criterion with respect to H to take place over a bounded subset of the space of $p \times p$ positive semidefinite matrices. This fact is important because it provides theoretical justification for the use of general optimization techniques such as gradient descent that rely on selecting the optimal bandwidth matrix over a continuum of positive semidefinite matrices, rather than a discrete grid. We refer the reader to [57], [9], and [58] for examples of using optimization algorithms such as the projected gradient decent algorithm to minimize a non-convex objective function over the continuum of positive semidefinite matrices in high-dimensional settings (e.g. p in the range of 10-60).

The abovementioned oracle property provides a theoretical rationale for the application of the Nadaraya-Watson kernel regression estimator to high dimensional data when $\psi(x)$ follows a multi-index regression model with an *unknown* index m. It is worth noting that multiple estimation methods have been studied for the multiple index model with a pre-specified index m, see, e.g., [59, 60, 10, 11, 12, 61, 62, 63], and [50] among others. Data driven methods such as cross validation have been commonly used in practice to select the index m. The oracle property we establish in this paper for the fully nonparametric Nadaraya-

Watson kernel regression estimator, which does not require any prior knowlege of m, implies that its rate of convergence is comparable to an estimator based on the multi index model where m is known in advance. Finally, the application of the Nadaraya-Watson estimator with a matrix-valued bandwidth is not limited to multi-index regression models. For example, it has been popularly used in metric learning, which is a much more general framework for both supervised and unsupervised learning with applications in fields such as computer vision, information retrieval, and bioinformatics [8].

The rest of the paper is organized as follows. In Section 2, we first discuss some optimality criteria we use for assessing predictive performance of an estimator and introduce notations for cross-validation schemes. Then we present a general oracle inequality for estimators using K-fold cross-validation. This result is later used to prove an oracle property for the kernel regression estimator defined in (3.3). In Sections 3 and 4, we illustrate the performance of the cross-validated kernel regression estimator using a bandwidth matrix in comparison with that using a scalar bandwidth and with an estimation method for the multi index model in simulations and on commonly used benchmark data sets from the UC Irvine Machine Learning repository. Further discussions are provided in Section 5. Sections 6 contains some lemmas and the proofs of the theoretical results in Section 2.

3.2 Main Results

3.2.1 Preliminaries

3.2.1.1 Optimality Criteria for Predictive Performance

Given an estimator $\hat{\psi}$ of ψ based on the observed data, consider the squared error loss, $L(Y, \hat{\psi}(X)) = (Y - \hat{\psi}(X))^2$, for an additional independent observation,

$O = (X, Y) \sim P_0$. Define

$$\tilde{\theta}_n(\hat{\psi}) = E_O[L(Y, \hat{\psi}(X))] = \int L(y, \hat{\psi}(x)) dP_0(x, y), \tag{3.4}$$

to be the conditional risk [64, 56], which is also referred to as the test error or generalization error [32] or the integrated squared error (ISE) [65]. We refer to this measure of performance as the conditional risk because it is conditional on the training set used to estimate $\hat{\psi}(X)$).

Note that the conditional risk is a random variable as $\hat{\psi}$ depends on the observed data. We call $E\tilde{\theta}_n(\hat{\psi})$ the marginal risk, or the expected test error, or the mean integrated squared error ($MISE$).

Define $\psi(X) = E[Y|X]$ to be the true conditional expectation of Y given X and $\theta_{opt} = E_O[(Y - \psi(X))^2]$. Then, for any square integrable estimator $\hat{\psi}(X)$, we have

$$\tilde{\theta}_n(\hat{\psi}) \geq \theta_{opt},$$

since $\psi(X) = E[Y|X]$ minimizes $E_O[(Y - \eta(X))^2]$ over all square integrable functions $\eta(X)$ of X (see, e.g., Corollary 8.17 of [66]). Hence θ_{opt} is a lower bound for both the conditional risk, and the marginal risk, and θ_{opt} is the optimal conditional risk. Moreover, we note that selecting an estimator $\hat{\psi}$ that minimizes the conditional risk is equivalent to minimizing $\tilde{\theta}_n(\hat{\psi}) - \theta_{opt}$ and that $\tilde{\theta}_n(\hat{\psi}) - \theta_{opt} = \int (\hat{\psi}(x) - \psi(x))^2 dP_{0,X}(x)$, where $P_{0,X}$ is the marginal distribution of X. We refer to $\tilde{\theta}_n(\hat{\psi}) - \theta_{opt}$ as the conditional excess risk and $E\tilde{\theta}_n(\hat{\psi}) - \theta_{opt}$ as the marginal excess risk.

Because the true distribution P_0 of the observations is unknown, the conditional risk must be estimated. A natural estimator of the conditional risk would be $\int L(y, \hat{\psi}(x)) dP_n(x, y)$, where P_n is the empirical distribution of the observations. Unfortunately, this estimate may be highly optimistic because the data has been used twice to first produce the estimator, $\hat{\psi}$ and then obtain the estimate of the conditional risk. A sample splitting procedure, whereby a (training) subset of the data is set aside to produce $\hat{\psi}$ and a separate (validation) subset of the data is used to estimate $\tilde{\theta}_n(\hat{\psi})$, would produce an estimate of the conditional risk that is less prone to this negative bias. Cross-validation schemes for estimating the conditional risk are elaborations of the aforementioned sample splitting procedure, in which observations alternate in their role of training and validation as described in the next section.

3.2.1.2 A Formal Explication of Cross-Validation

Let $\{\psi_k(x) : k \in \Xi_n\}$ denote a collection of estimators of $\psi(x)$ indexed by k, where $\Xi_n \subset R^d$ for some positive integer d. Below we formally present the concept of cross-validation, giving attention to the case in which the index k runs through a continuous range of values.

We denote a split of the data into training and validation sets via the binary vector $S_n = (S_{n1}, \ldots, S_{nn})^T \in \{0,1\}^n$, where

$$S_{ni} = \begin{cases} 0, & \text{if observation } i \text{ is in the training set,} \\ 1, & \text{if observation } i \text{ is in the validation set,} \end{cases}$$

and the set $\{0,1\}^n$ represents all possible splits of the data into training and validation sets. Define P^0_{n,S_n} and P^1_{n,S_n} as the empirical distribution of the observations in the training and validation set, respectively. Let $\psi_k = \psi_k(X|P^0_{n,S_n})$ be an estimator produced by applying an estimation procedure to observations in the training set determined by S_n. Define the conditional expectation, given the observations in the training set, of a function $f(O, \psi_k(X|P^0_{n,S_n}))$ by $E_O[f(O,\psi_k)|P^0_{n,S_n}]$, where f is a function depending on an independent observation $O \sim P_0$ and the estimator $\psi_k(X|P^0_{n,S_n})$.

From now on, we will assume that all data splits devote the same proportion of observations, $1-\pi$, to training. A cross-validation scheme is defined by assigning a set of *nsplit* probability weights $w_1, \ldots, w_{nsplit}$ such that $w_j > 0$ and $\sum_{j=1}^{nsplit} w_j = 1$ to a subset of *nsplit* elements of $\{0,1\}^n$. The corresponding cross-validation criterion is then defined as

$$\hat{\theta}^{CV}_{n(1-\pi)}(k) = E_{S_n} \int L(y, \psi_k(x|P^0_{n,S_n}))dP^1_{n,S_n}(x,y) \tag{3.5}$$

$$= \sum_{j=1}^{nsplit} w_j \int L(y, \psi_k(x|P^0_{n,S_n^{(j)}}))dP^1_{n,S_n^{(j)}}(x,y). \tag{3.6}$$

The K-fold cross-validation scheme is defined by splitting the n observations into K distinct subsets. This partition of the observations results in K binary vectors $S_n^{(1)}, \ldots, S_n^{(K)}$ where $S_n^{(j)}$ is created by letting observations in the jth element of the partition serve as the validation set. After minor modification, we may take $\pi = \lfloor n/K \rfloor / n$. The K-fold cross-validation scheme then puts a probability weight of $w_j = 1/K$ on each of these K binary vectors.

A natural benchmark for a cross-validation scheme is

$$\tilde{\theta}^{CV}_{n(1-\pi)}(k) = E_{S_n} \int L(y, \psi_k(x|P^0_{n,S_n}))dP_0(x,y) \tag{3.7}$$

$$= \sum_{j=1}^{nsplit} w_j \int L(y, \psi_k(x|P^0_{n,S_n^{(j)}}))dP_0(x,y), \tag{3.8}$$

which is referred to as the cross-validation benchmark, or the “commensurate optimal benchmark" [56]). The cross-validation benchmark $\tilde{\theta}^{CV}_{n(1-\pi)}(k)$ can be regarded as a cross-validation criterion when an infinite number of observations are available for validation. Although minimization of the cross-validation benchmark is not equivalent to minimization of the conditional risk over estimators that use all n observations, for K-fold cross-validation, the following relationship holds: $E\tilde{\theta}^{CV}_{n(1-\pi)}(k) = E\tilde{\theta}_{n(1-\pi)}(k)$, where $\tilde{\theta}_{n(1-\pi)}(k)$ denotes the conditional risk of ψ_k. Thus, the cross-validation benchmark has mean equal to the marginal risk based on approximately $n(1-\pi)$ observations rather than n observations.

If minimization of the cross-validation criterion is carried out over a continuous range Ξ_n of $\mathcal{R}^d$, there may not exist a $\hat{k}_n \in \Xi_n$ such that $\hat{\theta}^{CV}_{n(1-\pi)}(\hat{k}_n) = \inf_{k\in\Xi_n} \hat{\theta}^{CV}_{n(1-\pi)}(k)$. For this reason, we will consider a γ-suboptimal point $\hat{k}_n \in \Xi_n$ as in [67] such that $\hat{\theta}^{CV}_{n(1-\pi)}(\hat{k}_n) \leq \inf\{\hat{\theta}^{CV}_{n(1-\pi)}(k) : k \in \Xi_n\} + \gamma$, for some prespecified $\gamma > 0$. Note that a γ-suboptimal point is not necessarily unique. If a minimizer exists in Ξ_n, then it can be considered as a γ-suboptimal point with $\gamma = 0$.

3.2.2 An Oracle Property of the Kernel Regression Estimator

Our main theoretical result relies on an extension of an oracle inequality presented in [56] and [49], who considered sample splitting or cross-validation for selecting the best model from a discrete collection of models. Our extension below allows for optimization of the K-fold cross-validation criterion to take place over a continuous bounded subset of the space of $p \times p$ positive semidefinite matrices and consequently enables selecting the best model from a continuum of models.

3.2.2.1 An Oracle Inequality for K-Fold Cross Validation

We now present an oracle inequality that demonstrates that K-fold cross-validation produces an estimator that has near optimal performance according to the cross-validation benchmark.

Let $\hat{k}_n$ and $\tilde{k}_n$ be the γ-suboptimal minimizers of the K-fold cross-validation criterion $\hat{\theta}^{CV}_{n(1-\pi)}(k)$ and its cross-validation benchmark $\tilde{\theta}^{CV}_{n(1-\pi)}(k)$, respectively, over Ξ_n.

The following assumptions will be needed.

(A1) There exists a constant $M > 0$ such that $Pr(|Y| < M) = 1$ and $sup_X|\psi_k(X|P_n)| \leq$

$M < \infty$ almost surely for all $k \in \Xi_n$, where M does not depend on n and the supremum is over the support of the marginal distribution of X.

(A2) Ξ_n is a bounded set.

(A3) Assume that with probability 1, $L(Y, \psi_k(X|P_n)) - L(Y, \psi(X))$, as a function of k, is Lipshitz continuous with Lipshitz constant C. This Lipshitz constant, C, does not depend on the training set, P_n.

(A4) Let $c_1(n\pi, d, \Xi_n, M, \delta) = \log\left\{\left(\frac{n\pi}{c_2(M,\delta)}\right)^d 4(4\sqrt{d}C^2 4(1+2\delta)\mathrm{diam}(\Xi_n))^d\right\}$ and let $c_2(M, \delta) = 8(1+\delta)^2(\frac{M_2}{\delta} + \frac{M_1}{3})$, where $M_1 = 8M^2$ and $M_2 = 16M^2$. Assume $\delta > 0$ is such that $1/c_2(M, \delta) \leq M_1 8(\frac{1}{(1+2\delta)} + \frac{1}{3})$ and take n large enough that $c_1(n\pi, d, \Xi_n, M, \delta) > 1$.

The derivation of our result in the theorem below requires that terms of the form $L(Y, \psi_k(X|P_n)) - L(Y, \psi(X))$, where P_n depends on the split, converge to their expectation uniformly over $k \in \Xi_n$. Intuitively, if $L(Y, \psi_k(X|P_n)) - L(Y, \psi(X))$ converges to its expectation uniformly in k, then the K-fold cross-validation criterion will converge to the K-fold cross-validation benchmark uniformly in k. Assumptions (A1), (A2), and (A3) ensure this occurs. Assumption (A4) is a technical condition primarily serving to simplify the result.

Theorem 1. *Assume the above assumptions (A1)-(A4) hold. Then,*

$$0 \leq E\tilde{\theta}^{CV}_{n(1-\pi)}(\hat{k}_n) - \theta_{opt} \leq (1+2\delta)(E\tilde{\theta}^{CV}_{n(1-\pi)}(\tilde{k}_n) - \theta_{opt}) + \tag{3.9}$$

$$\frac{4c_2(M,\delta)}{n\pi} c_1(n\pi, d, \Xi_n, M, \delta) + \tag{3.10}$$

$$(1+\delta)\gamma. \tag{3.11}$$

Remark 1. *Because $E\tilde{\theta}^{CV}_{n(1-\pi)}(\tilde{k}_n) - \theta_{opt}$ is a lower bound for $E\tilde{\theta}^{CV}_{n(1-\pi)}(\hat{k}_n) - \theta_{opt}$, the above inequality states that on average cross-validation selects an estimator with close to optimal performance measured by the cross-validation benchmark if*

$n\pi$ large. Note that the tightness of the above inequality depends on n, π, d, and diam(Ξ_n). The bound becomes looser for larger values of d and diam(Ξ_n). As $n\pi$ grows, the bound becomes tighter.

Remark 2. *Our finite sample result in Theorem 1 suggests a trade-off with regard to the choice of what proportion of observations to use for validation versus training for the upper-bound of the above inequality. The expression on the right hand side of the above inequality depends heavily on the number of observations used for validation: $n\pi$. If π is too small, the number of observations used for validation will be small and the term in (3.10) will be large, which signals difficulty in identifying the model that minimizes the cross-validation benchmark. If π is large, the expectation of the optimal cross-validation benchmark based on training sets of size $n(1-\pi)$ may be a poor approximation for the expectation of the optimal conditional risk based on a training set of size n. A similar trade-off has been observed with regard to the choice of K in [68, 32, 69]. [70], perhaps providing a more refined discussion of this issue, suggest that the choice of K should depend, in part, on the signal-to-noise ratio.*

The above oracle inequality in Theorem 1 is derived for any collection of estimators indexed by k that ranges over a continuous bounded subset Ξ_n of R^d, satisfying (A1)-(A4). In the next section, we apply Theorem 1 to the Nadaraya-Watsion kernel regression estimator defined in (3.3) with a Gaussian kernel $K(u) = \exp(-||u||^2)$, indexed by a bandwidth matrix H. We also show that for this Nadaraya-Watson kernel regression estimator, a minimizer (corresponding to $\gamma = 0$) exists for both the K fold cross-validation criterion and the K-fold cross-validation benchmark.

3.2.2.2 An Oracle Property for a Nadaraya-Watson Estimator with a Matrix Valued Bandwidth

As in [67], we represent the set of symmetric positive semidefinite matrices, S_+^p, as elements in $\mathcal{R}^{p(p+1)/2}$. Explicitly, we represent a particular matrix H as $(h_{1,1}, h_{2,2}, \ldots, h_{p,p},$
$h_{2,1}, h_{3,1}, \ldots, h_{p,p-1})'$. The Frobenius norm of a symmetric matrix, H, viewed as an element in $\mathcal{R}^{p(p+1)/2}$, is defined as $||H||_F = (\sum_{i=1}^p h_{ii}^2 + 2\sum_{j<i} h_{ij}^2)^{1/2}$. To ensure that Ξ_n is bounded (A2), we will let Ξ_n consist of the elements of $S_+^p \subset \mathcal{R}^{p(p+1)/2}$ with Frobenius norm less than or equal to λ_n for some $\lambda_n > 0$.

Theorem 2. *Consider the class of kernel regression estimators ψ_H, defined in (3.3), where $K(u) = \exp(-||u||^2)$ is the Gausian kernel function and H is selected via K-fold cross-validation. Assume that $P(|Y| < M) = 1$ and that (A4) holds. Assume further that there exists a constant $B > 0$ such that $P(||X|| < B) = 1$. Then, we have the following finite sample result.*

(a) Minimizers of the K-fold cross-validation criterion and the K-fold cross-validation benchmark with respect to H exist in Ξ_n.

(b) Denote by $\hat{H}_n \in \Xi_n$ a minimizer of the K-fold cross-validation criterion and $\tilde{H}_n \in \Xi_n$ a minimizer of the K-fold cross-validation benchmark. Then, the assumptions (A1), (A2), and (A3) are satisfied and consequently the inequality of Theorem 1 holds, with $\gamma = 0$, $d = p(p+1)/2$ and $C = 64\sqrt{p(p+1)/2}B^2M^2$.

(c) Let $\psi_{n(1-\pi),\hat{H}_n}(x|P^0_{n,S_n^{(j)}})$ be the kernel regression estimator obtained by using $\hat{H}_n$ from part (b) and the jth training sample for any $1 \leq j \leq K$. Assume $\psi(X) = \phi(Tx)$ is a multi-index model defined such that $\phi : \mathcal{R}^m \to \mathcal{R}$ is Lipshitz continuous with Lipshitz constant R and T is an $m \times p$ orthogonal matrix. Let $\lambda_n = \sqrt{p}V(\log(n)n)^{1/3}$, where $V > 0$ is a positive constant.

Then, as $n \to \infty$,

$$\begin{aligned} E\tilde{\theta}^{CV}_{n(1-\pi)}(\hat{H}_n) - \theta_{opt} &= E\int (\psi_{n(1-\pi),\hat{H}_n}(x|P^0_{n,S^{(j)}_n}) - \psi(x))^2 dP_{0,X}(x) \\ &= O((n(1-\pi))^{\frac{-2}{m+2}} \log(n(1-\pi))^{\frac{m}{m+2}}). \end{aligned}$$

We see that in contrast to the result (3.2), the rate of convergence of the above estimator in part (c) depends on $m \le p$. The gain in efficiency can be substantial when m is much smaller than p.

Remark 3. *(Optimality of K-fold cross-validation) If*

$$\frac{c_1(n\pi, d, \Xi_n, M, \delta)}{(n\pi)\{E\tilde{\theta}^{CV}_{n(1-\pi)}(\tilde{k}) - \theta_{opt}\}} \to 0 \text{ as } n \to \infty, \tag{3.12}$$

then it follows, by dividing both sides of the oracle inequality in Theorem 2.1, that

$$\frac{E\tilde{\theta}^{CV}_{n(1-\pi)}(\hat{k}) - \theta_{opt}}{E\tilde{\theta}^{CV}_{n(1-\pi)}(\tilde{k}) - \theta_{opt}} \to 1 \text{ as } n \to \infty. \tag{3.13}$$

This implies that the estimator produced by K-fold cross-validation has asymptotically optimal predictive performance as measured by the cross-validation benchmark. Furthermore, assuming that λ_n grows at the rate specified in Theorem 2, the above condition (3.12) will then be satisfied if $n\pi(E\tilde{\theta}^{CV}_{n(1-\pi)}(\tilde{H}_n) - \theta_{opt})$ increases at a polynomial rate, n^γ, where $0 < \gamma < 1$. As n^{-1} is a "parametric rate" of convergence we would expect $E\tilde{\theta}^{CV}_{n(1-\pi)}(\tilde{H}_n) - \theta_{opt}$ to decrease at a rate slower than n^{-1}. Thus, we expect the condition (3.12) to hold in many circumstances. In fact, Remark 4 of [71] provides an example of a class of distribution within which distributions can be found such that it holds.

Remark 4. *(Computational aspects) The K-fold cross-validation criterion is differentiable with respect to H for the Nadaraya-Watson estimator (3.3) we consider. The derivative is presented in the Supplementary material. To find $\hat{H}_n$,* *we have implemented a variant of the gradient descent algorithm presented in [9]. Our simulation results and the data analysis demonstrate that this algorithm is capable of finding acceptable local minimizers. Development of more sophisticated algorithms for finding global minimizers as well as improvement of computational speed warrants further research.*

3.3 Simulations

We present in this section the results of a simulation study to illustrate the oracle property of the kernel regression estimator using a matrix-valued bandwidth by comparing its performance with a kernel regression estimator using a scalar valued bandwidth and a multi-index regression method introduced in [12] and [13] under multiple scenarios for multi index models. The estimator of [13] is implemented in the **R** package EDR and is denoted as EDR. EDR relies on estimation of the gradient of the regression function. We use 10-fold cross-validation to estimate the optimal scalar-valued and matrix-valued bandwidth parameters. To find the matrix-valued bandwidth, we applied a gradient descent algorithm to minimize the 10-fold cross-validation criterion. A grid search was used to find the scalar-valued bandwidth. Tuning parameters for EDR were chosen via 10-fold cross-validation. We also demonstrate how the performance of kernel regression, parametrized by a bandwidth matrix, degrades as the number of covariates increases.

In all simulations, the covariates are independent and standard normal random variables. For the single index regression model we let

$$Y = 2X_1 + 2X_2 + X_3 + X_4 + \epsilon, \tag{3.14}$$

where $\epsilon \sim N(0, \sigma^2)$. The standard deviation of the error term, σ, has been set to 0.10 and 0.20. Additional “noise" covariates (covariates with coefficient equal to 0) have been added such that p varies from 5, 10, to 20. For the five-index regression model, we let

$$Y = X_1^3 + X_2^3 + \cos\left(\frac{\pi}{2}(X_1 + X_3)\right) + \sqrt{|X_4|} + \sqrt{|X_1 + X_5|} + \epsilon, \tag{3.15}$$

where $\epsilon \sim N(0, \sigma^2)$. The sample size, n, varies from 250, 500, to 1,000 and σ takes the values 0.10 and 0.20.

The performance of these estimators is measured by taking the square root of the mean squared error (RMSE) on a test set of size 10,000. Each simulation scenario was repeated 100 times. The mean of the RMSEs over the 100 simulations as well as 95% confidence intervals for each simulation scenario are presented in Table 3.1.

Both EDR and the kernel regression estimator using a matrix-valued bandwidth outperform the kernel regression estimator using a scalar-valued bandwidth over all simulation scenarios. The rate of convergence of the kernel regression estimator using a scalar-valued bandwidth is slow enough that the RMSE changes only by very small amounts as the sample size increases. We found that the scalar bandwidth selected by 10-fold cross-validation led to significant over-smoothing for all simulation scenarios.

For the single-index model, EDR consistently outperforms the kernel regression estimator indexed by a matrix-valued bandwidth. This is unsurprising as EDR relies on a local-linear approximation rather than a local constant approximation. For the five-index regression model, the kernel regression estimator generally outperforms EDR when the number of covariates is equal to 5 and 10. We believe that for complex multi-index models EDR has difficulty estimating the gradient. When p is 20 for the five-index model, we believe that EDR and kernel regression perform similarly as boundary effects harm the performance of the kernel regression estimator.

n	p	SD	m	Matrix Bandwidth	EDR	Scalar Bandwidth
250	5	0.10	1	0.118 (0.118,0.119)	0.105 (0.105,0.106)	1.829 (1.829,1.83)
250	5	0.20	1	0.216 (0.216,0.216)	0.209 (0.208,0.21)	1.837 (1.837,1.838)
250	10	0.10	1	0.119 (0.118,0.119)	0.105 (0.105,0.105)	1.829 (1.828,1.83)
250	10	0.20	1	0.22 (0.22,0.221)	0.209 (0.209,0.21)	1.837 (1.836,1.838)
250	20	0.10	1	0.122 (0.121,0.122)	0.107 (0.107,0.108)	1.829 (1.828,1.83)
250	20	0.20	1	0.25 (0.249,0.251)	0.214 (0.214,0.214)	1.837 (1.836,1.837)
500	5	0.10	1	0.108 (0.108,0.108)	0.103 (0.102,0.104)	1.827 (1.827,1.828)
500	5	0.20	1	0.208 (0.208,0.208)	0.205 (0.204,0.206)	1.836 (1.835,1.837)
500	10	0.10	1	0.109 (0.109,0.109)	0.103 (0.103,0.103)	1.827 (1.827,1.828)
500	10	0.20	1	0.209 (0.209,0.209)	0.206 (0.205,0.206)	1.835 (1.835,1.836)
500	20	0.10	1	0.11 (0.11,0.11)	0.103 (0.103,0.103)	1.827 (1.826,1.828)
500	20	0.20	1	0.218 (0.217,0.219)	0.207 (0.206,0.207)	1.836 (1.835,1.836)
1000	5	0.10	1	0.104 (0.104,0.104)	0.101 (0.101,0.102)	1.826 (1.825,1.827)
1000	5	0.20	1	0.204 (0.204,0.204)	0.202 (0.202,0.203)	1.835 (1.834,1.835)
1000	10	0.10	1	0.104 (0.104,0.104)	0.102 (0.101,0.102)	1.826 (1.825,1.827)
1000	10	0.20	1	0.204 (0.204,0.205)	0.203 (0.202,0.203)	1.835 (1.834,1.836)
1000	20	0.10	1	0.105 (0.104,0.105)	0.102 (0.102,0.102)	1.827 (1.826,1.828)
1000	20	0.20	1	0.206 (0.206,0.206)	0.203 (0.203,0.203)	1.835 (1.834,1.835)
250	5	0.10	5	0.39 (0.388,0.391)	0.54 (0.516,0.564)	0.87 (0.869,0.87)
250	5	0.20	5	0.433 (0.432,0.435)	0.562 (0.552,0.572)	0.886 (0.886,0.887)
250	10	0.10	5	0.448 (0.447,0.45)	0.544 (0.524,0.564)	0.869 (0.869,0.87)
250	10	0.20	5	0.496 (0.495,0.497)	0.56 (0.55,0.57)	0.886 (0.886,0.887)
250	20	0.10	5	0.549 (0.546,0.551)	0.532 (0.528,0.536)	0.869 (0.869,0.87)
250	20	0.20	5	0.602 (0.599,0.604)	0.584 (0.577,0.591)	0.886 (0.886,0.887)
500	5	0.10	5	0.346 (0.345,0.347)	0.488 (0.482,0.495)	0.868 (0.868,0.869)
500	5	0.20	5	0.395 (0.394,0.396)	0.529 (0.521,0.538)	0.885 (0.885,0.886)
500	10	0.10	5	0.372 (0.371,0.373)	0.515 (0.493,0.536)	0.868 (0.868,0.869)
500	10	0.20	5	0.424 (0.423,0.425)	0.546 (0.52,0.573)	0.886 (0.885,0.886)
500	20	0.10	5	0.474 (0.472,0.476)	0.477 (0.472,0.482)	0.869 (0.868,0.869)
500	20	0.20	5	0.524 (0.522,0.525)	0.518 (0.51,0.526)	0.885 (0.885,0.886)
1000	5	0.10	5	0.306 (0.305,0.307)	0.48 (0.463,0.498)	0.868 (0.868,0.868)
1000	5	0.20	5	0.362 (0.361,0.362)	0.51 (0.501,0.519)	0.885 (0.885,0.885)
1000	10	0.10	5	0.315 (0.314,0.315)	0.472 (0.459,0.485)	0.868 (0.867,0.868)
1000	10	0.20	5	0.373 (0.372,0.374)	0.533 (0.499,0.568)	0.885 (0.885,0.885)
1000	20	0.10	5	0.399 (0.398,0.4)	0.419 (0.413,0.424)	0.868 (0.868,0.869)
1000	20	0.20	5	0.455 (0.454,0.456)	0.484 (0.471,0.498)	0.885 (0.885,0.885)

Table 3.1: RMSEs and 95% CIs for kerned regression estimator with matrix bandwidth, scalar bandwidth, and EDR under a single index model (Index # =1) and a five-index model (Index # =5).

3.4 Real Data Examples

In this section we demonstrate the estimated predictive performance of kernel regression using a bandwidth matrix versus kernel regression using a scalar bandwidth with the goal of understanding whether the asymptotic results presented in this article are indicative of finite sample performance on commonly explored data sets. For reference, we also compare the predictive performance of these kernel regression estimators to that of a linear regression estimator with no interactions and no nonlinear terms. Large differences in estimated prediction accuracy between the nonparametric kernel regression methods and the linear model may provide an indication of whether the linear model is failing to account for any interactions or nonlinearities although this comparison does not constitute a formal test.

To estimate the predictive accuracy of each method we used testing sets of approximate size $n \times 0.25$. Observations were split into 4 groups. This yielded 4 splits of the data into a set of size $n \times 0.75$ for training and a set of size $n \times 0.25$ for estimating the RMSE via a testing set. A final estimate of the RMSE was obtained by averaging the 4 estimates of the RMSE associated with each split. Within each training set 10-fold cross-validation was used to select the bandwidth matrix and scalar bandwidth for the two regression estimators.

We tested these methods on 3 data sets: the Boston housing data set, obtained via the R package MASS, the concrete compressive strength data set, and the auto-mpg data set, with the latter two being available from the UC Irvine Machine Learning Repository (https://archive.ics.uci.edu/ml). Continuous covariates were centered and scaled to have variance equal to 1. The results are presented in Table 3.2.

It is seen from Table 3.2 that the kernel regression estimator indexed by a bandwidth matrix yielded a smaller estimated RMSE than both the kernel re-

Data Set	n	p	EDR	Matrix BW	Scalar BW	Linear Regression
Boston housing	506	11	3.54	3.69	9.07	4.96
concrete	1030	8	6.77	6.97	16.60	10.55
auto-mpg	398	5	2.88	2.71	7.72	3.48

Table 3.2: Estimated RMSEs for three commonly used data sets. BW=bandwidth matrix.

gression estimator indexed by a scalar and the linear regression estimator. The kernel regression estimator indexed by a scalar bandwidth had the highest RMSE for all 3 data sets. The improvement in performance from using a bandwidth matrix over a scalar bandwidth suggests that there is likely a lower-dimensional structure which the bandwidth matrix is able to take advantage of. The lower RMSE of the kernel regression estimator indexed by a bandwidth matrix than linear regression is indicative of possible nonlinear structure or interactions which the linear regression estimator fails to account for.

3.5 Discussion

Previous theoretical study of cross-validation for the Nadaraya-Watson estimator has focused on some special bandwidth matrix structure [51, 52, 53, 54, 55, 49] under various optimality criteria, assumptions about the kernel, and assumptions about the data distribution. For instance, the results on cross-validation and kernel regression presented in [52, 53, 54], and [55] concern leave-one-out cross-validation and all consider the case when $H^* = h^* I_p$, where $h^* = 1/h$. The kernel estimator using such a scalar bandwidth, however, does not have the flexibility to take advantage of any low dimensional structure and thus suffers the curse of dimensionality. It is also problematic when some of the covariates are measured on different scales. We have derived a finite sample oracle inequality for the K-fold cross-validated Nadaraya-Watson kernel regression estimator indexed by a general bandwidth matrix and demonstrated that the resulting upper bound achieves a lower-dimensional rate of convergence when the true regression model is single or multi-index. The kernel estimator indexed by a general bandwidth matrix is also invariant to the scales of the covariates. In addition to the theoretical results, we have also corroborated the oracle property of the kernel regression estimator with a matrix-valued bandwidth by demonstrating its significant gain in efficiency over the kernel regression estimator with a scalar-valued bandwidth in numerical studies. Lastly, although it is not a primary focus of the article, we have conducted a simulation study to compare the performance of the kernel regression estimator using a matrix-valued bandwidth with a multi-index regression method, referred to as EDR here, introduced in [12] and [13] when the true regression model is single or multi-index. We have found that there exist scenarios for both estimators where they significantly outperform one another.

Our oracle inequality in Theorem 2.1 is a nontrivial extension of the result of [56] and [49] from a discrete set of models to a continuum of models for selecting the best model using cross validation. Similarly to [56] and [49], our result in Theorem 2.1 is not specific to kernel regression and thus potentially applies to a wide variety of estimation procedures in addition to kernel regression. We note that this generality is obtained with a small price. For example, our proof requires that the number of folds, K, be constant or grow slowly as a fraction of the sample size. In particular, this excludes leave-one-out cross-validation. The upper-bound in Theorem 2 may not be the tightest possible and it is possible that the actual rate of convergence of the estimator in Theorem 2.2 (c) could be faster without the $log(n)$ factor.

While the focus of the article is on K-fold cross-validation, our results easily generalize to other cross-validation schemes. For example, a cross-validation scheme that may lead to more stable model selection could be obtained by specifying more than just K splits in which approximately $n\pi$ observations are used for validation. An example of such a scheme is repeated K-fold cross-validation

([72]), wherein K-fold cross-validation is carried out repeatedly by using a different partition of the observations each time. It can be shown that our results also hold for repeated K-fold cross-validation.

3.6 Proofs

To prove the theorems of Section 2, we need to first discuss the concept of brackets and bracketing numbers and establish some lemmas.

3.6.1 Definitions and Results Regarding Brackets

Let $\mathcal{F}$ be a collection of functions $f(O)$. An L_2 ϵ-bracket determined by a pair of functions, l and u, such that $l(O) \leq u(O)$ and $E[|l(O) - u(O)|^2] \leq \epsilon^2$, is defined as the set of functions $f(O) \in \mathcal{F}$ such that $l(O) \leq f(O) \leq u(O)$. Such a bracket will be denoted as $[l, u]$. A minimal ϵ-covering of $\mathcal{F}$ with brackets is a collection of brackets such that each element of $\mathcal{F}$ is in at least one bracket and there exists no collection of ϵ brackets of smaller cardinality. The minimum number of brackets required to cover $\mathcal{F}$ is denoted by $N_{[]}(\epsilon, \mathcal{F})$.

By Jensen's inequality, we have $\epsilon^2 \geq E[|u - v|^2] \geq (E[|u - l|])^2$ which implies that $E[|u - l|] \leq \epsilon$.

Let $f_1, f_2 \in [l, u]$. We show that f_1 and f_2 have variances that are close to one another for small values of ϵ. This result will be used in the proof of Theorem 1.

Lemma 1. *Let f_1 and f_2 be elements of an L_2 ϵ-bracket, $[l, u]$. Then*

$$|\sigma^2_{f_1} - \sigma^2_{f_2}| \leq \epsilon(\sigma_{f_1} + \sigma_{f_2}) \tag{3.16}$$

Proof.

$$
\begin{aligned}
&E\Big\{\big[(f_1(O)-E[f_1(O)])-(f_2(O)-E[f_2(O)])\big]^2\Big\}\\
&=E\Big\{\big[f_1(O)-f_2(O)-E[f_1(O)-f_2(O)]\big]^2\Big\}\\
&\leq E\Big\{\big[(f_1(O)-f_2(O))-(E[f_1(O)-f_2(O)])\big]^2\Big\}+(E[f_1(O)-f_2(O)])^2\\
&=E[(f_1(O)-f_2(O))^2]\leq\epsilon^2.
\end{aligned}
$$

This implies that

$$\sqrt{E\Big\{\big[(f_1(O)-E[f_1(O)])-(f_2(O)-E[f_2(O)])\big]^2\Big\}}\leq\epsilon.$$

By the reverse triangle inequality, we have

$$\left|\sqrt{E[(f_1(O)-E[f_1(O)])^2]}-\sqrt{E[(f_2(O)-E[f_2(O)])^2]}\right|=\left|\sigma_{f_1}-\sigma_{f_2}\right|\leq\epsilon,$$

where $\sigma^2_{f_i}$ is the variance of $f_i (i=1,2)$.

Then $|\sigma^2_{f_1}$ - $\sigma^2_{f_2}|=|(\sigma_{f_1}-\sigma_{f_2})(\sigma_{f_1}+\sigma_{f_2})|\leq\epsilon(\sigma_{f_1}+\sigma_{f_2})$. □

3.6.2 Rate of Convergence of a Kernel Regression Estimator with Unbounded Support

In this section, we present a result extending Theorem 5.2 of [49] on the rate of convergence of the kernel regression estimator to handle the case where a Gaussian kernel is used. This result is used in the proof of Theorem 2.

In the next lemma, we restrict our attention to the special case where a single bandwidth parameter, $h>0$, is used. Therefore, instead of $k_{i,H}(x)=\exp(-(X_i-x)'H(X_i-x))$, we have $k_{i,h}(x)=\exp(-h(X_i-x)'(X_i-x))=\exp(-h||X_i-x||_2^2)$, where $||X_i-x||^2$ is the squared Euclidean distance between X_i and x. With h

fixed, we denote the resulting kernel estimator as ψ_n.

Consider a class of kernel regression estimators of the form

$$\psi_n(x) = \frac{\sum_{i=1}^n K((X_i - x)h)Y_i}{\sum_{i=1}^n K((X_i - x)h)}, \tag{3.17}$$

where $K : \mathcal{R}^d \to [0, \infty)$ is the kernel function. Theorem 5.2 of [49] provides an upper-bound on $E_O[(\psi_n(X) - \psi(X))^2]$ when $K(x)$ is a type of kernel with bounded support called a "boxed" kernel. Letting $S_{x,r}$ be a ball of radius r, centered at x, $K(x)$ is a boxed kernel if there exists $0 < r < r'$ and $b > 0$ such that $I_{\{x \in S_{0,r'}\}} \geq K(x) \geq bI_{\{x \in S_{0,r}\}}$. In the case of a Gaussian kernel, there does exist b and $r > 0$ such that $K(x) \geq bI_{\{x \in S_{0,r}\}}$. However, the Gaussian kernel has unbounded support, therefore, there exists no r' such that $I_{\{x \in S_{0,r'}\}} \geq K(x)$.

Lemma 2. *Assume X has support such that there exists $B > 0$ such that $P(||X|| < B) = 1$. Assume the true regression function, $\psi(x)$ is Lipshitz continuous over the support of X, with Lipshitz constant R, and that Var$(Y|X = x) \leq \sigma^2$ over the support of X. Then, we have*

$$E[(\psi_n(X) - \psi(X))^2] \leq \frac{R^2 \log(n)}{h} + \frac{2R^2B^2\tilde{c}h^{p/2}}{nb} + \frac{2\sigma^2\tilde{c}h^{p/2}}{nb}, \tag{3.18}$$

where $\tilde{c}$ depends on p and B. Furthermore, if h increases to infinity as

$$h_n^* = \left(\frac{A_1 \log(n) n}{A_2 \frac{p}{2}}\right)^{\frac{2}{p+2}}$$

where $A_1 = R^2$ and $A_2 = \frac{2\tilde{c}}{b}(R^2B^2 + \sigma^2)$, then the above bound yields

$$E[(\psi_n(X) - \psi(X))^2] \leq A \log(n)^{\frac{p}{p+2}} n^{\frac{-2}{p+2}},$$

where $A = A_1^{\frac{p}{p+2}} A_2^{\frac{2}{p+2}} \left((p/2)^{\frac{2}{p+2}} + (p/2)^{\frac{p}{p+2}}\right)$.

The proof of Lemma 2 is given in the supplementary materials.

3.6.3 Proof of Theorem 1

Proof. To simplify notation, for this finite sample result, we suppress the dependence of $\hat{k}_n$ and $\tilde{k}_n$ on the sample size, thus, let $\hat{k}_n = \hat{k}$ and $\tilde{k}_n = \tilde{k}$. We begin with the same decomposition as [56].

$$\begin{aligned}
0 &\le \tilde{\theta}^{CV}_{n(1-\pi)}(\hat{k}) - \theta_{opt} \\
&= E_{S_n} \int L(y, \psi_{\hat{k}}(x|P^0_{n,S_n})) - L(y, \psi(x)) dP_0(x,y) \\
&\quad - (1+\delta) E_{S_n} \int L(y, \psi_{\hat{k}}(x|P^0_{n,S_n})) - L(y, \psi(x)) dP^1_{n,S_n}(x,y) \\
&\quad + (1+\delta) E_{S_n} \int L(y, \psi_{\hat{k}}(x|P^0_{n,S_n})) - L(y, \psi(x)) dP^1_{n,S_n}(x,y) \\
&\le E_{S_n} \int L(y, \psi_{\hat{k}}(x|P^0_{n,S_n})) - L(y, \psi(x)) dP_0(x,y) \\
&\quad - (1+\delta) E_{S_n} \int L(y, \psi_{\hat{k}}(x|P^0_{n,S_n})) - L(y, \psi(x)) dP^1_{n,S_n}(x,y) \\
&\quad + (1+\delta) E_{S_n} \int L(y, \psi_{\tilde{k}}(x|P^0_{n,S_n})) - L(y, \psi(x)) dP^1_{n,S_n}(x,y) \\
&\quad + (1+\delta)\gamma \\
&= E_{S_n} \int L(y, \psi_{\hat{k}}(x|P^0_{n,S_n})) - L(y, \psi(x)) dP_0(x,y) \\
&\quad - (1+\delta) E_{S_n} \int L(y, \psi_{\hat{k}}(x|P^0_{n,S_n})) - L(y, \psi(x)) dP^1_{n,S_n}(x,y) \\
&\quad + (1+\delta) E_{S_n} \int L(y, \psi_{\tilde{k}}(x|P^0_{n,S_n})) - L(y, \psi(x)) dP^1_{n,S_n}(x,y) \\
&\quad - (1+2\delta) E_{S_n} \int L(y, \psi_{\tilde{k}}(x|P^0_{n,S_n})) - L(y, \psi(x)) dP_0(x,y) \\
&\quad + (1+2\delta) E_{S_n} \int L(y, \psi_{\tilde{k}}(x|P^0_{n,S_n})) - L(y, \psi(x)) dP_0(x,y) \\
&\quad + (1+\delta)\gamma
\end{aligned}$$

As in [56], denote the sum of the first and second terms above by $R_{n\hat{k}}$. Denote the sum of the third and fourth term as $T_{n,\tilde{k}}$. The fifth term is the cross-validation benchmark. Therefore, we have

$$0 \leq \tilde{\theta}^{CV}_{n(1-\pi)}(\hat{k}) - \theta_{opt} \leq (1+2\delta)\{\tilde{\theta}^{CV}_{n(1-\pi)}(\tilde{k}) - \theta_{opt}\} + R_{n\hat{k}} + T_{n,\tilde{k}} + (1+\delta)\gamma. \quad (3.19)$$

The objective is then to find upper bounds for $ER_{n\hat{k}}$ and $ET_{n,\tilde{k}}$. We will show that the same bound applies for both $ER_{n\hat{k}}$ and $ET_{n,\tilde{k}}$. At present, we find an upper bound for $R_{n\hat{k}}$.

Fixing our attention on a particular split into training and validation sets, say the jth split $S_n^{(j)}$, let $Z_{ki}^{S_n^{(j)}} = Z_{ki}^{S_n^{(j)}}(Y_i, X_i) = L(Y_i, \psi_k(X_i|P^0_{n,S_n^{(j)}})) - L(Y_i, \psi(X_i))$, where i is such that $S_{ni}^{(j)} = 1$. Let $Z_k^{S_n^{(j)}} = Z_k^{S_n^{(j)}}(Y, X) = L(Y, \psi_k(X|P^0_{n,S_n^{(j)}})) - L(Y, \psi(X))$ have the same distribution as $Z_{ki}^{S_n^{(j)}}$. Note that $|Z_{ki}^{S_n^{(j)}}| \leq M_1$ *a.s.* We also have $\int Z_k^{S_n^{(j)}}(x,y)dP_0(x,y) = E[Z_k^{S_n^{(j)}}|P^0_{n,S_n^{(j)}}] - \theta_{opt} \geq 0$.

For any k, let

$$R_{nk} = \frac{1}{K}\sum_{j=1}^{K} R_{nk}(S_n^{(j)}),$$

where

$$R_{nk}(S_n^{(j)}) = \int Z_k^{S_n^{(j)}}(x,y)dP_0(x,y) - (1+\delta)\{\frac{1}{n\pi}\sum_{\{i:S_{ni}^{(j)}=1\}} Z_{ki}^{S_n^{(j)}}(X_i,Y_i)\}.$$

After adding and subtracting $\delta \int Z_k^{S_n^{(j)}}(x,y)dP_0(x,y)$, we have

$$R_{nk}(S_n^{(j)}) = (1+\delta)\left\{\frac{1}{n\pi}\sum_{\{i:S_{ni}^{(j)}=1\}}(\int Z_k^{S_n^{(j)}}(x,y)dP_0(x,y) - Z_{ki})\right\} - \delta\int Z_k^{S_n^{(j)}}(x,y)dP_0(x,y). \quad (3.20)$$

Therefore,

$$\begin{aligned}
&P\Big(R_{n\hat{k}}(S_n^{(j)}) > s|P^0_{n,S_n^{(j)}}\Big)\\
&= P\Big((1+\delta)\{\frac{1}{n\pi}\sum_{\{i:S_{ni}^{(j)}=1\}} E[Z_{\hat{k}i}^{S_n^{(j)}}|P^0_{n,S_n^{(j)}}] - Z_{\hat{k}i}^{S_n^{(j)}}\} - \delta E[Z_{\hat{k}i}^{S_n^{(j)}}|P^0_{n,S_n^{(j)}}] > s|P^0_{n,S_n^{(j)}}\Big)\\
&\le P\Big(\sup_{k\in\Xi_n}\Big\{(1+\delta)\{\frac{1}{n\pi}\sum_{\{i:S_{ni}^{(j)}=1\}} E[Z_{ki}^{S_n^{(j)}}|P^0_{n,S_n^{(j)}}] - Z_{ki}^{S_n^{(j)}}\} - \delta E[Z_{ki}^{S_n^{(j)}}|P^0_{n,S_n^{(j)}}]\Big\} > s|P^0_{n,S_n^{(j)}}\Big)\\
&\le P\Big(\sup_{k\in\Xi_n}\Big\{(1+\delta)\{\frac{1}{n\pi}\sum_{\{i:S_{ni}^{(j)}=1\}} E[Z_{ki}^{S_n^{(j)}}|P^0_{n,S_n^{(j)}}] - Z_{ki}^{S_n^{(j)}}\} - \delta\frac{Var(Z_{ki}^{S_n^{(j)}}|P^0_{n,S_n^{(j)}})}{M_2}\Big\} > s|P^0_{n,S_n^{(j)}}\Big),
\end{aligned}$$

where in the last inequality we have used the inequality

$$Var(Z_{ki}^{S_n^{(j)}}|P^0_{n,S_n^{(j)}})/M_2 \le E[Z_{ki}^{S_n^{(j)}}|P^0_{n,S_n^{(j)}}]$$

(see Lemma 3 of [56]). We will find an exponential bound for the previous probability via bracketing numbers.

Let $\mathcal{F}_{P^0_{n,S_n^{(j)}}} = \{Z_k^{S_n^{(j)}} = Z_k^{S_n^{(j)}}(X,Y) : k \in \Xi_n\}$. Set $\epsilon = s/(4(1+2\delta))$. Let $[l_v, u_v]$ $(v = 1,\ldots,N(\epsilon))$ be a minimal L_2 ϵ-bracketing of $\mathcal{F}_{P^0_{n,S_n^{(j)}}}$ where $N(\epsilon) = N_{[]}(\epsilon, \mathcal{F}_{P^0_{n,S_n^{(j)}}})$. By $A1$ we may assume without loss of generality that $|l_v| \le M_1$ and that $|u_v| \le M_1$.

Note that $N_{[]}(\epsilon, \mathcal{F}_{P^0_{n,S_n^{(j)}}})$ depends on the training set defined by $S_n^{(j)}$. Given the training set used in split $S_n^{(j)}$, $N(\epsilon) = N_{[]}(\epsilon, \mathcal{F}_{P^0_{n,S_n^{(j)}}})$ is a fixed number rather than a random variable. We will find a bound for $N(\epsilon, \mathcal{F}_{P^0_{n,S_n^{(j)}}})$ that is independent of the training set. Choose a representative, $f_v \in [l_v, u_v]$ from each bracket and let $f_{vi} = f_v(O_i)$. Similarly, let $l_{vi} = l_v(O_i)$ and $u_{vi} = u_v(O_i)$. If $Z_k^{S_n^{(j)}} \in \mathcal{F}_{P^0_{n,S_n^{(j)}}}$ such that $Z_k^{S_n^{(j)}} \in [l_v, u_v]$ we have

$$\frac{1}{n\pi} \sum_{\{i:S_{ni}^{(j)}=1\}} (Z_{ki}^{S_n^{(j)}} - E[Z_{ki}^{S_n^{(j)}} | P^0_{n,S_n^{(j)}}])$$
$$\leq \frac{1}{n\pi} | \sum_{\{i:S_{ni}^{(j)}=1\}} f_{vi} - E[f_{vi} | P^0_{n,S_n^{(j)}}]| + \tag{3.21}$$
$$\frac{1}{n\pi} | \sum_{\{i:S_{ni}^{(j)}=1\}} (Z_{ki}^{S_n^{(j)}} - f_{vi}) - E[(Z_{ki}^{S_n^{(j)}} - f_{vi}) | P^0_{n,S_n^{(j)}}]|. \tag{3.22}$$

Consider the term in (3.22). By the triangle inequality and the fact that $[l_v, u_v]$ is an ϵ bracket we have

$$\frac{1}{n\pi} | \sum_{\{i:S_{ni}^{(j)}=1\}} (Z_{ki}^{S_n^{(j)}} - f_{vi}) - E[(Z_{ki}^{S_n^{(j)}} - f_{vi}) | P^0_{n,S_n^{(j)}}]|$$
$$\leq \frac{1}{n\pi} \sum_{\{i:S_{ni}^{(j)}=1\}} |Z_{ki}^{S_n^{(j)}} - f_{vi}| + E[|Z_{ki}^{S_n^{(j)}} - f_{vi}| | P^0_{n,S_n^{(j)}}]$$
$$\leq \frac{1}{n\pi} \sum_{\{i:S_{ni}^{(j)}=1\}} |u_{vi} - l_{vi}| + E[|u_{vi} - l_{vi}| | P^0_{n,S_n^{(j)}}]$$
$$= \frac{1}{n\pi} \sum_{\{i:S_{ni}^{(j)}=1\}} |u_{vi} - l_{vi}| - E[|u_{vi} - l_{vi}| | P^0_{n,S_n^{(j)}}]| + 2E[|u_{vi} - l_{vi}| | P^0_{n,S_n^{(j)}}]$$
$$\leq \frac{1}{n\pi} (\sum_{\{i:S_{ni}^{(j)}=1\}} |u_{vi} - l_{vi}| - E[|u_{vi} - l_{vi}| | P^0_{n,S_n^{(j)}}]|) + 2\epsilon.$$

Replacing the term in (3.22) by the final term in the above series of inequalities yields the inequality

$$\begin{aligned}
&\frac{1}{n\pi}|\sum_{\{i:S_{ni}^{(j)}=1\}}(Z_{ki}^{S_n^{(j)}}-E[Z_{ki}^{S_n^{(j)}}|P^0_{n,S_n^{(j)}}])| \\
&\leq \frac{1}{n\pi}|\sum_{\{i:S_{ni}^{(j)}=1\}} f_{vi}-E[f_{vi}|P^0_{n,S_n^{(j)}}]|+ \qquad (3.23)\\
&\quad \frac{1}{n\pi}\sum_{\{i:S_{ni}^{(j)}=1\}}(|u_{vi}-l_{vi}|-E[|u_{vi}-l_{vi}||P^0_{n,S_n^{(j)}}]|)+2\epsilon. \qquad (3.24)
\end{aligned}$$

Using Lemma 1 and the fact that $\sqrt{Var(Z_{ki}^{S_n^{(j)}}|P^0_{n,S_n^{(j)}})}\leq M_2$ and $\sqrt{Var(f_v|P^0_{n,S_n^{(j)}})}\leq M_2$, we have

$$Var(Z_{ki}^{S_n^{(j)}}|P^0_{n,S_n^{(j)}})\geq Var(f_v|P^0_{n,S_n^{(j)}})-2M_2\epsilon. \qquad (3.25)$$

Combining (3.23), (3.24), and (3.25), we have

$$\begin{aligned}
&(1+\delta)\{\frac{1}{n\pi}\sum_{\{i:S_{ni}^{(j)}=1\}}E[Z_{ki}^{S_n^{(j)}}|P^0_{n,S_n^{(j)}}]-Z_{ki}^{S_n^{(j)}}\}-\delta\frac{Var(Z_{ki}^{S^{(j)}}|P^0_{n,S_n^{(j)}})}{M_2}\\
&\leq (1+\delta)\frac{1}{n\pi}\sum_{\{i:S_{ni}^{(j)}=1\}}(|u_{vi}-l_{vi}|-E[|u_{vi}-l_{vi}||P^0_{n,S_n^{(j)}}]|)\\
&+(1+\delta)\frac{1}{n\pi}|\sum_{\{i:S_{ni}^{(j)}=1\}}f_{vi}-E[f_{vi}|P^0_{n,S_n^{(j)}}]|-\delta\frac{Var(f_v|P^0_{n,S_n^{(j)}})}{M_2}\\
&+2(1+\delta)\epsilon+2\delta\epsilon.
\end{aligned}$$

Making use of the fact that $s - 2(1+\delta)\epsilon - 2\delta\epsilon = s/2$, we have

$$
\begin{aligned}
&P\Bigg(\sup_{k\in\Xi_n}\Bigg\{(1+\delta)\Big\{\frac{1}{n\pi}\sum_{\{i:S_{ni}^{(j)}=1\}} E[Z_{ki}^{S_n^{(j)}}|P^0_{n,S_n^{(j)}}] - Z_{ki}^{S_n^{(j)}}\Big\} - \delta\frac{Var(Z_{ki}^{S^{(j)}}|P^0_{n,S_n^{(j)}})}{M_2}\Bigg\} > s|P^0_{n,S_n^{(j)}}\Bigg)\\
&\leq P\Bigg(\sup_{v\in\{1,\ldots,N(\epsilon)\}}\Bigg\{(1+\delta)\frac{1}{n\pi}\sum_{\{i:S_{ni}^{(j)}=1\}}(|u_{vi}-l_{vi}| - E[|u_{vi}-l_{vi}||P^0_{n,S_n^{(j)}}]|)+\\
&\qquad (1+\delta)\frac{1}{n\pi}|\sum_{\{i:S_{ni}^{(j)}=1\}} f_{vi} - E[f_{vi}|P^0_{n,S_n^{(j)}}]| - \delta\frac{Var(f_v|P^0_{n,S_n^{(j)}})}{M_2}\Bigg\} > \frac{s}{2}|P^0_{n,S_n^{(j)}}\Bigg)\\
&\leq N(\epsilon)\max_{v\in\{1,\ldots,N(\epsilon)\}} P\Bigg(\Bigg\{(1+\delta)\frac{1}{n\pi}\sum_{\{i:S_{ni}^{(j)}=1\}}(|u_{vi}-l_{vi}| - E[|u_{vi}-l_{vi}||P^0_{n,S_n^{(j)}}]|)+\\
&\qquad (1+\delta)\frac{1}{n\pi}|\sum_{\{i:S_{ni}^{(j)}=1\}} f_{vi} - E[f_{vi}|P^0_{n,S_n^{(j)}}]| - \delta\frac{Var(f_v|P^0_{n,S_n^{(j)}})}{M_2}\Bigg\} > \frac{s}{2}|P^0_{n,S_n^{(j)}}\Bigg)
\end{aligned}
$$

Fixing a particular value of v, we use the general inequality that for any pair of random variables A and B, $P(A+B > a+b) \leq P(A>a) + P(B>b)$:

$$
\begin{aligned}
&P\Bigg(\Bigg\{(1+\delta)\frac{1}{n\pi}\sum_{\{i:S_{ni}^{(j)}=1\}}(|u_{vi}-l_{vi}| - E[|u_{vi}-l_{vi}||P^0_{n,S_n^{(j)}}]|)+\\
&\qquad (1+\delta)\frac{1}{n\pi}|\sum_{\{i:S_{ni}^{(j)}=1\}} f_{vi} - E[f_{vi}|P^0_{n,S_n^{(j)}}]| - \delta\frac{Var(f_v|P^0_{n,S_n^{(j)}})}{M_2}\Bigg\} > \frac{s}{2}|P^0_{n,S_n^{(j)}}\Bigg)\\
&\leq P\Bigg((1+\delta)\frac{1}{n\pi}\sum_{\{i:S_{ni}^{(j)}=1\}}(|u_{vi}-l_{vi}| - E[|u_{vi}-l_{vi}||P^0_{n,S_n^{(j)}}]|) > \frac{s}{4}|P^0_{n,S_n^{(j)}}\Bigg) \qquad (3.26)\\
&+ P\Bigg((1+\delta)\frac{1}{n\pi}|\sum_{\{i:S_{ni}^{(j)}=1\}} f_{vi} - E[f_{vi}|P^0_{n,S_n^{(j)}}]| - \delta\frac{Var(f_v|P^0_{n,S_n^{(j)}})}{M_2} > \frac{s}{4}|P^0_{n,S_n^{(j)}}\Bigg)\\
&\qquad (3.27)
\end{aligned}
$$

First we obtain an upper-bound for (3.27). For simplicity let $Var(f_v|P^0_{n,S_n^{(j)}}) =$

$\sigma^2_{f_v}$. By Bernstein's inequality we have,

$$P\bigg((1+\delta)\frac{1}{n\pi}|\sum_{\{i:S^{(j)}_{ni}=1\}} f_{vi} - E[f_{vi}|P^0_{n,S^{(j)}_n}]| - \delta\frac{\sigma^2_{f_v}}{M_2} > \frac{s}{4}|P^0_{n,S^{(j)}_n}\bigg)$$
$$= P\bigg(\frac{1}{n\pi}|\sum_{\{i:S^{(j)}_{ni}=1\}} f_{vi} - E[f_{vi}|P^0_{n,S^{(j)}_n}]| > \frac{1}{1+\delta}(\delta\frac{\sigma^2_{f_v}}{M_2} + \frac{s}{4})|P^0_{n,S^{(j)}_n}\bigg)$$
$$= 2\exp\bigg(-\frac{n\pi}{2(1+\delta)^2}\frac{(\delta\frac{\sigma^2_{f_v}}{M_2}+\frac{s}{4})^2}{\sigma^2_{f_v} + \frac{M_1}{3(1+\delta)}(\delta\frac{\sigma^2_{f_v}}{M_2}+\frac{s}{4})}\bigg)$$

We simplify the following expression in the exponent:

$$\frac{(\delta\frac{\sigma^2_{f_v}}{M_2}+\frac{s}{4})^2}{\sigma^2_{f_v} + \frac{M_1}{3(1+\delta)}(\delta\frac{\sigma^2_{f_v}}{M_2}+\frac{s}{4})} = \frac{(\delta\frac{\sigma^2_{f_v}}{M_2}+\frac{s}{4})}{\frac{\sigma^2_{f_v}}{(\delta\frac{\sigma^2_{f_v}}{M_2}+\frac{s}{4})} + \frac{M_1}{3(1+\delta)}} \geq \frac{(\delta\frac{\sigma^2_{f_v}}{M_2}+\frac{s}{4})}{\frac{M_2}{\delta} + \frac{M_1}{3(1+\delta)}} \geq \frac{1}{4}\frac{s}{\frac{M_2}{\delta}+\frac{M_1}{3}}$$

As $c_2(M,\delta) = (1+\delta)^2 8(\frac{M_2}{\delta}+\frac{M_1}{3})$, we therefore have

$$P\bigg((1+\delta)\frac{1}{n\pi}|\sum_{\{i:S^{(j)}_{ni}=1\}} f_{vi} - E[f_{vi}|P^0_{n,S^{(j)}_n}]| - \delta\frac{\sigma^2_{f_v}}{M_2} > \frac{s}{4}|P^0_{n,S^{(j)}_n}\bigg)$$
$$\leq 2\exp\bigg(-\frac{n\pi}{c_2(M,\delta)}s\bigg). \tag{3.28}$$

Now consider the term in (3.26). Using Bernstein's inequality and the fact that $Var(|u_{vi} - l_{vi}|) \leq E|u_{vi} - l_{vi}|^2 \leq E|u_{vi} - l_{vi}|M_1 \leq M_1\epsilon$, we have

$$\begin{aligned}
&P(\frac{1}{n\pi}(\sum_{\{i:S_{ni}^{(j)}=1\}} |u_{vi} - l_{vi}| - E[|u_{vi} - l_{vi}||P^0_{n,S_n^{(j)}}, S_n^{(j)}]) > \frac{s}{4}|P^0_{n,S_n^{(j)}}, S_n^{(j)}) \\
&\leq 2\exp(-\frac{1}{2}\frac{n\pi(\frac{s}{4})^2}{M_1\epsilon + M_1\frac{s}{4}\frac{1}{3}}) \\
&= 2\exp(-\frac{1}{2}\frac{n\pi(\frac{s}{4})^2}{M_1\frac{s}{4(1+2\delta)} + M_1\frac{s}{4}\frac{1}{3}}) \\
&= 2\exp(-\frac{1}{M_1 8}\frac{n\pi}{\frac{1}{(1+2\delta)} + \frac{1}{3}}s).
\end{aligned} \tag{3.29}$$

By $A4$, δ is taken small enough so that $1/(c_2(M,\delta)) \leq M_1 8(\frac{1}{(1+2\delta)} + \frac{1}{3})$ and, in this case, the upper bound for (3.27) (Inequality 3.28) is larger than the upper bound for (3.26) (Inequality 3.29) and we have

$$\begin{aligned}
&P(R_{n,\hat{k}}(S_n^{(j)}) > s|P^0_{n,S_n^{(j)}}) \\
&\leq N(\epsilon)4\exp(-\frac{n\pi}{c_2(M,\delta)}s) \\
&\leq N(\frac{s}{4(1+2\delta)})4\exp(-\frac{n\pi}{c_2(M,\delta)}s).
\end{aligned}$$

By assumption $\mathcal{F}_{P^0_{n,S_n^{(j)}}} = \{Z_k : k \in \Xi_n\}$ are Lipshitz, with Lipshitz constant C independent of $P^0_{n,S_n^{(j)}}$. Therefore, by Example 19.7 in [73] and Example 27.1 of [74],

$$N(\frac{s}{4(1+2\delta)}) \leq \Big(\frac{4\sqrt{d}C^2 4(1+2\delta)diam(\Xi_n)}{s}\Big)^d.$$

Thus,

$$P(R_{n,\hat{k}}(S_n^{(j)}) > s|P^0_{n,S_n^{(j)}}) \leq 4\Big(\frac{4\sqrt{d}C^2 4(1+2\delta)\text{diam}(\Xi_n)}{s}\Big)^d \exp(-\frac{n\pi}{c_2(M,\delta)}s)$$

and this implies that

$$P(R_{n,\hat{k}}(S_n^{(j)}) > s) \leq 4\Big(\frac{4\sqrt{d}C^2 4(1+2\delta)\mathrm{diam}(\Xi_n)}{s}\Big)^d \exp(-\frac{n\pi}{c_2(M,\delta)}s)$$

Note that $ER_{n,\hat{k}} = ER_{n,\hat{k}}(S_n^{(j)})$. In general, for any random variable Z, $EZ \leq EI(Z>0)Z = \int_0^\infty P(Z>z)dz \leq u + \int_u^\infty P(Z>z)dz$ $(u>0)$. If we assume $u > (\frac{c_2(M,\delta)}{n\pi})$, we obtain the following upper bound:

$$\begin{aligned}
&ER_{n,\hat{k}} = ER_{n,\hat{k}}(S_n^{(j)}) \leq u + \int_u^\infty 4\Big(\frac{4\sqrt{d}C^2 4(1+2\delta)\mathrm{diam}(\Xi_n)}{s}\Big)^d \exp(-\frac{n\pi}{c_2(M,\delta)}s)ds \\
&\leq u + \Big(\frac{n\pi}{c_2(M,\delta)}\Big)^d 4(4\sqrt{d}C^2 4(1+2\delta)\mathrm{diam}(\Xi_n))^d \int_u^\infty \exp(-\frac{n\pi}{c_2(M,\delta)}s)ds \\
&= u + \Big(\frac{n\pi}{c_2(M,\delta)}\Big)^d 4(4\sqrt{d}C^2 4(1+2\delta)\mathrm{diam}(\Xi_n))^d \frac{c_2(M,\delta)}{n\pi}\exp(-\frac{n\pi}{c_2(M,\delta)}u)
\end{aligned}$$

If we let

$$u = \frac{c_2(M,\delta)}{n\pi}\log\Big\{\Big(\frac{n\pi}{c_2(M,\delta)}\Big)^d 4(4\sqrt{d}C^2 4(1+2\delta)\mathrm{diam}(\Xi_n))^d\Big\},$$

we have

$$\begin{aligned}
ER_{n,\hat{k}} &\leq \frac{c_2(M,\delta)}{n\pi}c_1(n\pi,d,\Xi_n,M,\delta) + \frac{c_2(M,\delta)}{n\pi} \\
&\leq 2\frac{c_2(M,\delta)}{n\pi}c_1(n\pi,d,\Xi_n,M,\delta)
\end{aligned}$$

(by $A4$, u is indeed larger than $(c_2(M,\delta)/n\pi)$). The last inequality also follows by $A4$ as $c_1(n\pi,d,\Xi_n,M,\delta)$ is assumed to be larger than 1.

The same bound also holds for $ET_{n,\tilde{k}}$ as $T_{n,\tilde{k}}$ has the same form as $R_{n,\hat{k}}$. This is seen by noting that

$$\begin{aligned}
T_{n,\tilde{k}} &= (1+\delta)E_{S_n}\int L(y,\psi_{\tilde{k}}(x|P^0_{n,S_n})) - L(y,\psi(x))dP^1_{n,S_n}(x,y) \\
&\quad - (1+2\delta)E_{S_n}\int L(y,\psi_{\tilde{k}}(x|P^0_{n,S_n})) - L(y,\psi(x))dP_0(x,y) \\
&= (1+\delta)E_{S_n}\int L(y,\psi_{\tilde{k}}(x|P^0_{n,S_n})) - L(y,\psi(x))dP^1_{n,S_n}(x,y) \\
&\quad - (1+\delta)E_{S_n}\int L(y,\psi_{\tilde{k}}(x|P^0_{n,S_n})) - L(y,\psi(x))dP_0(x,y) \\
&\quad - \delta E_{S_n}\int L(y,\psi_{\tilde{k}}(x|P^0_{n,S_n})) - L(y,\psi(x))dP_0(x,y)
\end{aligned}$$

Noting that $Z_{\tilde{k}}^{S_n^{(j)}}(x,y) = L(y,\psi_{\tilde{k}}(x|P^0_{n,S_n^{(j)}})) - L(y,\psi(x))$, and comparing the above expression with that in (3.20), we see that the upper bound obtained for $ER_{n,\hat{k}}$ also holds for $ET_{n,\tilde{k}}$.

Thus, we have

$$\begin{aligned}
0 \leq E\tilde{\theta}^{CV}_{n(1-\pi)}(\hat{k}) - \theta_{opt} \leq\ & (1+2\delta)\{E\tilde{\theta}^{CV}_{n(1-p)}(\tilde{k}) - \theta_{opt}\}+ \\
& \frac{4c_2(M,\delta)}{n\pi}c_1(n\pi,d,\Xi_n,M,\delta)+ \\
& (1+\delta)\gamma
\end{aligned}$$

□

3.6.4 Proof of Theorem 2

Proof. The K-fold cross-validation criterion is continuous as a function of H (the choice of Gaussian kernel is important for this purpose). Note that Ξ_n is closed and bounded, therefore Ξ_n is compact. As Ξ_n is compact and the cross-validation criterion is continuous, there exists a minimizer of the K-fold cross-validation criterion in Ξ_n. The cross-validation benchmark for K-fold cross-validation is proportional to the sum of conditional risks for K kernel regression estimators each based off of a training set of size $n(1-\pi)$. If we can show that one of these conditional risks is continuous as a function of k, it will follow that the cross-validation benchmark is continuous. Fixing, say, the jth split, $\int L(y, \psi_H(x|P^0_{n,S_n^{(j)}})dP_0(x,y)$, is continuous as a consequence of the dominated convergence theorem (see Theorem 6.27 of [66]), therefore the cross-validation benchmark is continuous. Again, because $\tilde{\theta}^{CV}_{n(1-\pi)}(H)$ is continuous and Ξ_n is compact, there also exists a minimizer of $\tilde{\theta}^{CV}_{n(1-\pi)}(H)$ in Ξ_n. Thus, part (a) of Theorem 2 holds.

$A1$ holds because the kernel regression estimator is bounded by M. $A2$ is satisfied because Ξ_n is bounded. $A4$ holds by assumption. This leaves us to show that $A3$ holds.

Let $k_{i,H} = K(H^{1/2}(X_i - x))$. Consider the partial derivative of $L(y, \psi_H(x|P_n)) - L(y, \psi(x))$, where $x = (x_1, \dots, x_p)'$ is in the support of X. We have for $u \neq v$,

$$\left|\frac{\partial L(y, \psi_H(x|P_n)) - L(y, \psi(x))}{\partial H_{uv}}\right|$$
$$= \left|4(\psi_H(x|P_n) - y)\frac{\sum_{i=1}^n (\psi_H(x|P_n) - Y_i)k_{i,H}(x)(X_{iu} - x_u)(X_{iv} - x_v)}{\sum_{i=1}^n k_{i,H}(x)}\right|$$
$$\leq 64M \sum_{i=1}^n \frac{k_{i,H}(x)MB^2}{\sum_{i=1}^n k_{i,H}(x)} = 64M^2B^2,$$

where, for the inequality, we have used the fact that $|(\psi_H(x|P_n) - y)| \leq 2M$, $|(X_{iu} - x_u)| \leq 2B$, and $|(X_{iv} - x_v)| \leq 2B$.

When $u = v$ we have the same expression as above with the 4 replaced by 2. Thus, the above derivative is bounded above by a constant and the constant is independent of the training set. As a function of H, $L(y, \psi_H(x|P_n)) - L(y, \psi(x))$ is continuously differentiable and has derivative bounded by $64M^2B^2$, therefore by [75] (Exercise 2.15) it is Lipshitz continuous with Lipshitz constant $C = 64\sqrt{p(p+1)/2}B^2M^2$. Thus, condition $A3$ is satisfied and part (b) is proven.

Finally, we prove part (c). Let $\tilde{H}_n$ be a bandwidth matrix that minimizes the cross-validation benchmark $\tilde{\theta}^{CV}_{n(1-\pi)}(H)$. By definition, for any other fixed bandwidth matrix, $\breve{H}_n \in \Xi_n$, the below inequality holds:

$$E\tilde{\theta}^{CV}_{n(1-\pi)}(\tilde{H}_n) - \theta_{opt} \leq E\tilde{\theta}^{CV}_{n(1-\pi)}(\breve{H}_n) - \theta_{opt}.$$

In addition, because the cross-validation technique being utilized is K-fold cross-validation and by the properties of conditional risk discussed in the introduction, it is the case that for $\breve{H}_n$ we have,

$$E\tilde{\theta}^{CV}_{n(1-\pi)}(\breve{H}_n) - \theta_{opt} = E_{O_1,\ldots,O_{n(1-\pi)},X}[(\psi_{n(1-\pi),\breve{H}_n}(X) - \psi(X))^2]$$

where the expectation is taken over a training set of size $n(1-\pi)$, $O_1, \ldots, O_{n(1-\pi)} \sim P_{0,X}$, and a newly observed covariate, $X \sim P_{0,X}$. The conditions of Theorem 2 will be met for sufficiently small δ and for sufficiently large n. In this case, we have the following upper bound for $E\tilde{\theta}^{CV}_{n(1-\pi)}(\hat{H}_n) - \theta_{opt}$:

$$\begin{aligned} E\tilde{\theta}^{CV}_{n(1-\pi)}(\hat{H}_n) - \theta_{opt} \leq (1+2\delta)(E\tilde{\theta}^{CV}_{n(1-\pi)}(\breve{H}_n) - \theta_{opt}) + \\ \frac{4c_2(M,\delta)}{n\pi} c_1(n\pi, d, \Xi_n, M, \delta). \end{aligned} \tag{3.30}$$

Given our choice of λ_n, we will show there exists a sequence of bandwidth matrices $\breve{H}_n \in \Xi_n$ that yields the desired result. Furthermore, λ_n must go to ∞ at a rate slow enough such that the second term on the right-hand side of the inequality in (3.30) is of smaller order than the first term.

Define

$$h_n(q) = V(\log(n)n)^{1/(q+2)},$$

where $V > 0$ is positive constant. Note that

$$\lambda_n = \sqrt{p}V(\log(n)n)^{1/3} = \sqrt{p} \max_{q \in \{1,\ldots,p\}} h_n(q)$$

Choose $\breve{H}_n = T'\Lambda T$, where Λ is a diagonal matrix with its m diagonal entries equal to $h_n(m)$. Let $||\breve{H}_n||_O$ be the operator norm or the largest eigenvalue of the matrix $\breve{H}_n$. As the largest eigenvalue of $\breve{H}_n$ is $h_n(m)$, $||\breve{H}_n||_O = h_n(m)$. We have $||\breve{H}_n||_F \leq \sqrt{p}||\breve{H}_n||_O = \sqrt{p}h_n(m) \leq \lambda_n$. Therefore, $\breve{H}_n \in \Xi_n$.

The kernel regression estimator $\psi_{n(1-\pi),\breve{H}_n}(X)$ is equivalent to the kernel regression estimator obtained by regressing the m-dimensional covariate vector TX_i on Y_i, using the bandwidth matrix $h_n(m)I_m$. Denote this equivalent estimator as $\phi_{n(1-\pi),h_n(m)}(TX)$. We have

$$\begin{aligned}
E\tilde{\theta}^{CV}_{n(1-\pi)}(\breve{H}) - \theta_{opt} &= E_{O_1,\ldots,O_{n(1-\pi)},X}[(\psi_{n(1-\pi),\breve{H}_n}(X) - \psi(X))^2] \\
&= E_{O_1,\ldots,O_{n(1-\pi)},X}[(\phi_{n(1-\pi),h_n(m)}(TX) - \phi(TX))^2].
\end{aligned}$$

Thus, by using Inequality (3.18) from Lemma 2:

$$E\tilde{\theta}^{CV}_{n(1-\pi)}(\breve{H}) - \theta_{opt} \leq A' \log(n(1-\pi))^{\frac{m}{m+2}}(n(1-\pi))^{\frac{-2}{m+2}}, \tag{3.31}$$

where $A' = (A_1V^{-1} + A_2V^{\frac{m}{m+2}})$. Note that in the application of Lemma 2, there exists σ^2 such that $Var(Y|X=x) \leq \sigma^2$ because Y is a.s. bounded.

Next consider the rate at which $diam(\Xi_n)$ grows. If $H \in \Xi_n$ then $||H|| \leq ||H||_F \leq \lambda_n$ ($||H||$ is simply the Euclidean norm of the unique elements in H). Thus, $diam(\Xi_n)$ grows at rate $O(\lambda_n)$. Therefore, the second term on the right-hand side of the inequality (3.30) is indeed of smaller order than the first.

By the properties of the conditional risk discussed in the introduction,

$$E\tilde{\theta}^{CV}_{n(1-\pi)}(\hat{H}_n) - \theta_{opt} = E\int(\psi_{n(1-\pi),\hat{H}_n}(x|P^0_{n,S^{(j)}_n}) - \psi(x))^2 dP_{0,X}(x). \tag{3.32}$$

The desired result then follows by (3.30).

□

CHAPTER 4

Metric Learning for Right Censored Survival Outcomes

4.1 Introduction

Due to the increasing availability of data from medical databases and cancer registries, there is increasing demand for regression methods in survival analysis able to take advantage of large, information-rich data sets. In such settings, it is likely that there is enough information in the data to detect complex interactions and nonlinear structure. Parametric models such as the accelerated failure time (AFT) model or semi-parametric models such as the Cox proportional-hazards (CoxPH) model may fail to capture complex interactions or non-linear structure. In data rich settings, a fully nonparametric model is often preferable. Nonparametric methods may also be used to check modeling assumptions. In this way, nonparametric methods serve to complement parametric methods.

Numerous nonparametric regression methods have been developed for survival data. Under independent right censoring, Beran [76] and Dabrowska [77, 78] developed estimators of the conditional distribution function and conditional quantile function and studied the properties of such estimators. Li and Doss [79] extended Beran's estimator. Li and Datta [80] and Li and Van Keilegom [81] developed new inferential procedures for these estimators. Nonparametric methods for the estimation of the conditional quantile function was studied by Dabrowska [82], Gannoun [83] and Heuchenne and Van Keilegom [84]. Methods for nonparametric estimation of the conditional mean were developed by Doksum and Yandell [85] and Zheng [86], Carbonez et al [87], and more recently, Kohler et al [88, 89].

Unfortunately, these nonparametric regression methods are subject to the “curse of dimensionality." When the number of covariates is larger than five or six, these fully nonparametric regression methods often provide highly unstable estimates. Parametric methods such as AFT models are more stable but can be biased if the parametric assumptions are violated. While semiparametric methods such as the Cox's proportional hazards model (CoxPh) or Aalen's additive regression model [90, 91] relax certain parametric assumptions, they still make strong assumptions about the form of the hazard function. In this article, we introduce a fully nonparametric regression method that maintains stability when the number of covariates is larger than five or ten.

Metric learning is an area of active research in the machine learning community [92, 93, 57, 9, 7]. Metric learning is used for regression as well as multi-class classification. It has been successfully utilized in such applications as facial recognition and image retrieval [94, 95]. Metric learning is a general framework for local nonparametric classification and regression. As the name suggests, a metric learning algorithm begins by defining a distance metric, known up to a finite number of parameters. Subjects that are “close" to one another according to this distance metric will have similar predicted outcomes. Metric learning uses the data to estimate the unknown parameters defining this distance metric. Often this optimal distance metric is determined by solving a constrained optimization problem that has been set up to balance the bias and variance of the resulting regression estimate.

In the case of regression, the optimal distance metric may be defined as the minimizer of a regularized sum of squared errors. In the setting of right censored survival data, the sum of squared errors is unobserved. We address this issue by applying the transformation of Koul et al [14] to the censored survival times. If the censoring times and survival times are independent, the relationship between the covariates and the outcome is preserved by this transformation procedure. To improve the efficiency of our method, an algorithm similar to that of Buckley and James [16] is used to iteratively impute censored survival times. These imputed survival times are then used to re-estimate the optimal distance metric.

In section 2, we introduce metric learning and, in the case of regression, we discuss the relationship between metric learning and kernel regression. In section 3, we discuss synthetic variables and use synthetic variables to extend metric learning to right censored survival data. In sections 4 and 5, we compare our metric learning algorithm to the semiparametric CoxPH model and the parametric AFT model in simulations and on a subset of early-stage non-small-cell lung cancer cases from the SEER (Surveillance, Epidemiology, and End Results) database. We conclude with a discussion of limitations and further directions for research.

4.2 Extending Metric Learning to Right-Censored Survival Data

4.2.1 Metric Learning

In the setting of regression we assume our data is independent and identically distributed (iid): (Y_i, X_i) for $i = 1, \ldots, n$ and $Y_i = f(X_i) + \epsilon_i$, where $Y_i \in \mathbb{R}$ and $X_i \in \mathbb{R}^p$. The error terms, ϵ_i, are assumed to be iid and independent of X_i with mean 0. Our objective is to estimate the conditional mean, $f(X_i)$. To this end, many nonparametric estimation methods rely on some form of local averaging. Intuitively, these methods work because subjects with similar values of X should have similar values of Y. A local estimator of $f(x)$ may take the form $\hat{f}(x) = \sum_{i=1}^{n} W_x(X_i)Y_i$, where the weights $W_x(X_i)$ are nonnegative and sum to 1. Certain kernel regression estimators and partition based estimators, such as regression trees, are of this form. Informally, $W_x(X_i)$ can be thought of as representing how close X_i and x are.

For kernel regression, $W_x(X_i) = k(X_i, x)/\sum_{i=1}^n k(X_i, x)$, where k is a kernel function. When $p = 1$, common choices of kernel include the Epinechnikov $k_h(X_i, x) = \frac{.75(1-|\frac{x-X_i}{h}|^2)}{h} I(|\frac{x-X_i}{h}| \leq 1)$ or Gaussian kernel $k_h(X_i, x) = \frac{1}{\sqrt{2\pi}h} e^{-\frac{1}{2}(\frac{x-X_i}{h})^2}$ where h, the bandwidth, controls the bias and variance of the resulting estimator of f. When p is larger than 1, k_h may take the form of a product kernel $k_h(X_i, x) = \Pi_{j=1}^p k_{h_j}(X_{ij}, x_j)$, where $X_i' = (X_{i1}, \ldots, X_{ip})$, $x' = (x_1, \ldots, x_p)$ and $h' = (h_1, \ldots, h_p)$ is a vector of bandwidth parameters controlling the bias-variance trade-off. When, for all $j = 1, \ldots, p$, k_{h_j} is a Gaussian kernel, $k(X_i, x)$ depends on X_i and x only through the weighted Euclidean distance: $d(X_i, x) = (X_i - x)'D(X_i - x)$, where D is a diagonal matrix with $D_{jj} = h_j^{-1}$ for $j = 1, \ldots, p$. Kernel regression methods often perform worse when the number of covariates is large because, in higher dimensions, data points may be far away from one another according to the usual Euclidean distance.

In the case of regression, metric learning can be seen as a generalization of kernel regression in which the data is used to estimate the optimal bandwidth matrix. In this case, the estimator is defined as above, $\hat{f}(x) = \sum_{i=1}^n W_x(X_i)Y_i$, where $W_x(X_i) = k_M(X_i, x)/\sum_{i=1}^n k_M(X_i, x)$ and k_M is multivariate kernel function depending on a semi-positive definite matrix, M, to be estimated using the data. In this paper, we use the Gaussian kernel: $k_M(X_i, x) = \exp(-(X_i - x)'M(X_i - x))$. If the multivariate Gaussian kernel with invertible bandwidth matrix, H, is used, the matrix M is equivalent to the inverse of the bandwidth matrix, H^{-1} [96], however, relaxing M so that it may be less than full rank allows greater control of model complexity and interpretability.

The optimal value of M is determined by minimizing the leave-one-out cross-validation criterion $s(M) = \frac{\sum_{i=1}^n (y_i - \hat{y}_i)^2}{n}$, where $\hat{y}_i = \sum_{i \neq j}^n Y_j k_{ij}/\sum_{i \neq j}^n k_{ij}$, where $k_{ij}(M) = k_{ij} = \exp((X_i - X_j)'M(X_i - X_j))$. The complexity of the model can be summarized by trace of M, $\text{Tr}(M)$ or Frobenius norm $||M||_2 = \sum_i \sum_j m_{ij}^2$ where m_{ij} element of M in the ith row and jth column. As $\text{Tr}(M)$ or $||M||_2$ become large the bias of the resulting estimator decreases and variance increases. However, M does more than control the variance and bias; in addition, it allows the estimator to take advantage of lower-dimensional structure.

We now present the details of the computational methods used to minimize the objective function. This algorithm is presented in [9]. One apparent issue is that the matrix M is constrained to be positive semidefinite. We solve the minimization problem using an iterative gradient descent algorithm with a projection step to ensure M is positive semidefinite. In our implementation, a grid of step-sizes, $\alpha = (\alpha_1, \ldots, \alpha_m)$, is pre-specified and the best step size is chosen at each iteration of the algorithm. Start out with an initial value $M^{(0)}$. Given $M^{(l)}$, $M^{(l+1)}$ is obtained by the following algorithm:

1. Given $M^{(l)}$, calculate the matrix $K^{(l)}$ with (u, v)th entry equal to $k_{uv}(M^{(l)})$ and calculate the predicted outcomes $\hat{y}_i^{(l)}$ using $K^{(l)}$.

2. The gradient $\nabla s(M)$ is evaluated at $M^{(l)}$. Letting $P = \sum_{i=1}^n P_i = \sum_{i=1}^n (\hat{y}_i - y_i) \sum_{j \neq i}^n (\hat{y}_i - y_j) \frac{k_{ij}}{\sum_{j \neq i}^n k_{ij}} (X_i - X_j)(X_i - X_j)'$, the gradient is equal to R where $R_{uv} = 4P_{uv}$ for $u \neq v$ and $R_{uv} = 2P_{uv}$.

3. For each step-size α_r in α, calculate $L = M^{(l)} - \alpha_r \nabla s(M^{(l)})$, then project L onto the space of all positive semidefinite matrices, yielding a candidate $M_r^{(l+1)}$ for $M^{(l)}$. This projection is done first by calculating the eigendecomposition of $L = Q \Lambda Q^T$. Letting Λ^+ be the diagonal matrix with diagonal entries $max(\Lambda_{ii}, 0)$, the projection is $Q \Lambda^+ Q^T$.

4. $M^{(l+1)}$ is defined as the matrix $M_r^{(l+1)}$ that yields the lowest value of the objective function.

The iterative algorithm continues until the relative improvement in the objective function is less than a prespecified level ϵ: $|1 - s(M^{(l+1)})/s(M^{(l)})| < \epsilon$.

When n or p is large, metric learning can be computationally intensive. The calculation of the gradient can be computationally intensive because all outer products $(X_i - X_j)(X_i - X_j)'$ must be calculated for $i \neq j$. Calculation of $K^{(l)}$ also requires that all pairwise Mahalanobis distances be calculated. As in [9], we employ the heuristic of setting $\frac{k_{ij}}{\sum_{i \neq j} k_{ij}}$ equal to 0 if it is extremely small, as is often the case, for the sake of avoiding calculation of the outer product $(X_i - X_j)(X_i - X_j)'$ can be avoided.

4.2.2 Metric Learning for Right-Censored Outcomes

Assume survival times, censoring times, and covariates for all subject are iid: (T_i, C_i, X_i). In the presence of right censoring, we do not observe T_i for all individuals, rather, we observe $\tilde{T}_i = \min(T_i, C_i)$ along with $\delta_i = I(T_i < C_i)$ and the vector of covariates of X_i. We assume that T_i is independent of C_i. For the sake of reducing heteroscedasticity, we will analyze the log survival times: $Y_i = \log(T_i), \tilde{Y}_i = \log(\tilde{T}_i)$. We will assume that, on the log-scale, the regression model above holds so that $Y_i = f(X_i) + \epsilon_i$.

The metric learning algorithm described above is not directly applicable in the presence of right censoring. As noted earlier, the objective function $s(M) = \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{n}$cannot be minimized because y_i is not observed for those individuals whose survival times have been censored. To overcome this issue, we replace the right-censored survival times $\tilde{Y}_i$ by so-called synthetic variables that allow for the recovery of the true regression function, $f(X)$.

The synthetic variable approach was introduced in Koul [14] under the assumption of a linear regression model with fixed rather than random covariates, $x_i : Y_i = \beta' x_i + \epsilon_i$. According to Koul's approach, the right censored (log transformed) survival times, $\tilde{Y}_i$, are replaced by synthetic survival times

$Y_i^* = \tilde{Y}_i\delta_i/V(\tilde{Y}_i)$, where $V(t) = P(C_i > t)$. Under the assumption that the survival times T_i are independent of the censoring times we have $E[Y_i^*|X_i] = \beta' x_i$. Thus, regression of the synthetic survival times on the covariates x_i yields an unbiased estimate of β. Of course, $V(t)$ is unknown in most situations. In practice, V is replaced by its Kaplan-Meier estimate, $\hat{V}(t)$.

We propose addressing the issue of right censoring by first applying Koul's method to the right censored survival times. Then the metric learning algorithm described above is applied with the synthetic times replacing the censored times. Kohler [88] extended the synthetic variable approach to nonparametric regression. It is generally true of the above synthetic variables that $E[Y_i^*|X_i] = E[Y_i|X_i] = f(X_i)$. The assumption that $f(X_i)$ be linear is not required.

Kohler [88] prove consistency results for a simple class of local averaging estimators. They assume the covariates and survival time, (X_i, Y_i), are random and independent of C_i. Further extensions have also been developed under the assumption that Y_i is conditionally independent of C_i given X_i. In addition to this independence assumption, additional assumptions about the support of the survival times and support of censoring times must be made. Letting $U(t) = P(T_i > t)$, $T_V = \sup\{t : V(t) > 0\}$ and $T_U = \sup\{t : U(t) > 0\}$, they assume $T_U < \infty$, U is continuous, and $V(T_U) > 0$. Intuitively, this condition prevents the censoring process from obscuring the tails of the distribution of the survival times. For metric learning to perform well, we believe that similar assumptions must hold. It is important to note that the latter assumption usually fails when a type-1 censoring scheme is used.

If the above condition does not hold, it is possible to use our metric learning algorithm to estimate the conditional restricted mean rather than the conditional mean. The conditional $\tau-$restricted mean is defined as $E[\min(Y_i, \tau)|X_i]$, where $\tau > 0$ [97]. The procedure for estimating the conditional (unrestricted) mean can be adapted to estimating the conditional restricted mean by replacing the outcome Y_i with $\min(Y_i, \tau)$. In particular, our method can be adapted by treating censored observations such that $C_i > \tau$ as observations that have experienced an event.

4.2.3 Further Refinements

In practice, we have found that the metric learning algorithm for right censored data described above can perform poorly because of the heteroscedasticity induced by the synthetic variable transformation. Although the relationship between the synthetic variables and covariates preserves the original regression relationship the error structure is not preserved. Metric learning has particular difficulty with heteroscedasticity because it uses a single distance metric and this single distance metric uniformly controls the level of regularization across the entire covariate space. In the presence of heteroscedasticity, it would be ideal to apply heavier regularization over parts of the covariate space where the variability is higher.

To improve the performance of metric learning in the presence of right censoring, we apply a Buckley-James [16] type iterative algorithm which reduces the level of heteroscedasticity. Let $\check{Y}_i = E[Y_i|\tilde{Y}_i, X_i, \delta_i]$. Then we have $E[\check{Y}_i|X_i] = f(X_i)$ and $\check{Y}_i = f(X_i) + \epsilon_i^*$, where the ϵ_i^* are idendepent. It can then be shown that

$$\check{Y}_i = \delta_i \tilde{Y}_i + (1 - \delta_i)(f(X_i) + \int_{\tilde{e}_i}^{\infty} \epsilon d\{\frac{1 - W(\epsilon)}{W(\tilde{e}_i)}\}), \tag{4.1}$$

where W is the survival function of the residuals ϵ_i. If $f(X_i)$ and W were known, we could could calculate $\check{Y}_i$; if the $\check{Y}_i$ were known, we would be able to estimate $f(X_i)$. This suggests an iterative algorithm, whereby we obtain an estimate $\hat{f}(X_i)$ and then impute the $\check{Y}_i$. Once the $\check{Y}_i$ are imputed, we obtain a new estimate of $\hat{f}(X_i)$. To impute $\check{Y}_i$, we need an estimate of W. We use the Kaplan-Meier estimator. This iterative algorithm is continued until the estimated value of M has stabilized: $||M^{(l+1)}||_2/||M^{(l)}||_2 < \epsilon$.

4.3 Simulations

In this section, we investigate the performance of metric learning in a variety of simulation scenarios. It is of great interest whether metric learning is able detect lower dimensional structure and utilize it for improved predictive performance. In particular, if the true regression function $f(X_i)$ is a single index regression model, $g(X_i'\theta)$, we would hope that M would be approximately rank 1 so that the resulting local averages are taken roughly along the level curves of f. These simulations also demonstrate the extent to which metric learning performs well when the number of covariates is moderately large and how metric learning performs as the censoring rate varies. We compare metric learning to the CoxPH model in one scenario where the proportional-hazards assumption is satisfied to see how much efficiency is lost by assuming a fully nonparametric model. Then we make the same comparison in a scenario where the proportional hazards assumption fails and demonstrate the gains of metric learning over the CoxPH model when modeling assumptions are incorrect.

For all simulation scenarios, we vary the sample size, n, between 500 and 1000. The covariates were independent of one another with uniform distribution between -2 and 2, $Unif[-2, 2]$. The number of covariates is varied between $p = 5, 10$, and 20. The censoring times C_i are distributed according to the marginal distribution of T_i plus a constant, c, where c is chosen to achieve a desired censoring rate. Each simulation scenario was repeated 100 times. In both simulation scenarios, f depends on the first five covariates. Additional independent $Unif[-2, 2]$ covariates were added to test how metric learning performs when the dimensionality of the model grows. In applications, the user may not know which of a given set of covariates f depends upon.

In the first simulation scenario, we assume a log-linear model:

$$Y_i = .2X_1 + .4X_2 + .6X_3 + .8X_4 + X_5 + \epsilon_i$$

where $\epsilon_i \sim .5S$ and S has standard extreme value distribution. This error distribution leads to Weibull distributed survival times, therefore the proportional-hazards assumption is satisfied.

As a comparator to metric learning, a CoxPH model and a log-normal AFT model were fit to the data. Then the predictive performance of metric learning and the other two models was evaluated on a test set of the same size by calculating the root-mean squared error,

$$\widehat{RMSE}_{\hat{f}} = \sqrt{\frac{\sum_{i=1}^{n}(\hat{f}(X_i) - Y_i)^2}{n}}$$

As can be seen from Tables 4.1 and 4.2, metric learning is less accurate than the CoxPH model when the censoring rate low, however, it is clearly able to pick up a good portion of the signal. The performance of metric learning suffers most when the censoring rate is high, especially when the number of covariates is 20.

In the second simulation scenario the $Y_i = \sqrt{\sum_{m=1}^{5} X_m} + \epsilon_i$, where $\epsilon_i \sim N(0, 0.5)$, normal with mean 0 and standard deviation 0.5. Thus, f is of the form $g(X'\theta)$, with $g(t) = \sqrt{t}$ and $\theta = c(1, 1, 1, 1, 1)$ as described above. In this case, we expect the CoxPH model to perform worse than metric learning. First, the CoxPH model requires that the user specify the functional form of f. Even if the functional form for f is correctly specified, the error distribution of ϵ_i may be such that the proportional-hazards assumption is not met. Even if the form of f is specified correctly, normal errors will violate the proportional-hazards assumption. In this simulation, we fit a CoxPH model using only the main effects, as this is often the starting point when conducting an analysis using the CoxPH model.

n	p	Censoring Rate	Rounds of Imputation	Metric Learning	CoxPH
500	5	0.15	0	0.907	0.650
500	5	0.15	1	0.703	0.649
500	5	0.15	5	0.686	0.651
500	5	0.25	0	1.153	0.653
500	5	0.25	1	0.739	0.651
500	5	0.25	5	0.704	0.652
500	5	0.50	0	1.688	0.668
500	5	0.50	1	0.899	0.664
500	5	0.50	5	0.791	0.664
500	10	0.15	0	1.159	0.651
500	10	0.15	1	0.796	0.652
500	10	0.15	5	0.754	0.655
500	10	0.25	0	1.522	0.656
500	10	0.25	1	0.831	0.651
500	10	0.25	5	0.772	0.655
500	10	0.50	0	2.531	0.669
500	10	0.50	1	1.010	0.668
500	10	0.50	5	0.884	0.670
500	20	0.15	0	1.581	0.656
500	20	0.15	1	1.132	0.659
500	20	0.15	5	1.095	0.657
500	20	0.25	0	2.004	0.660
500	20	0.25	1	1.185	0.660
500	20	0.25	5	1.135	0.660
500	20	0.50	0	3.106	0.680
500	20	0.50	1	1.437	0.680
500	20	0.50	5	1.268	0.678

Table 4.1: Simulation results for linear Weibull model, $n = 500$

The results are displayed in Tables 4.3 and 4.4.

We see that the CoxPH model performs no better than ignoring the covariates and estimating the marginal mean of the Y_i. Meanwhile, metric learning performs quite well when the number of covariates when the censoring rate is below fifty percent. Again, heavy censoring and a large number unimportant covariates leads to serious degradation of performance.

n	p	Censoring Rate	Rounds of Imputation	Metric Learning	CoxPH
1000	5	0.15	0	0.804	0.648
1000	5	0.15	1	0.684	0.647
1000	5	0.15	5	0.674	0.649
1000	5	0.25	0	1.042	0.648
1000	5	0.25	1	0.709	0.647
1000	5	0.25	5	0.687	0.647
1000	5	0.50	0	1.445	0.655
1000	5	0.50	1	0.858	0.654
1000	5	0.50	5	0.763	0.653
1000	10	0.15	0	1.068	0.648
1000	10	0.15	1	0.721	0.649
1000	10	0.15	5	0.692	0.649
1000	10	0.25	0	1.343	0.650
1000	10	0.25	1	0.759	0.649
1000	10	0.25	5	0.706	0.649
1000	10	0.50	0	2.286	0.657
1000	10	0.50	1	0.913	0.656
1000	10	0.50	5	0.797	0.659
1000	20	0.15	0	1.496	0.652
1000	20	0.15	1	0.994	0.654
1000	20	0.15	5	0.952	0.651
1000	20	0.25	0	1.932	0.652
1000	20	0.25	1	1.059	0.651
1000	20	0.25	5	0.963	0.653
1000	20	0.50	0	3.249	0.661
1000	20	0.50	1	1.355	0.664
1000	20	0.50	5	1.129	0.661

Table 4.2: Simulation results for linear Weibull model, $n = 1,000$

4.4 Data Analysis

In this section we apply metric learning to two popular benchmark data sets. First, we discuss measures of predictive performance for right-censored survival data to assess various methods on these data sets. We will utilize two measures of predictive performance, each with different properties. First we utilize, Harrell's c-index [98] is a widely used measure of predictive performance for survival data. Second, we obtain an estimate of the RMSE by using the K-fold cross-validation

n	p	Censoring Rate	Rounds of Imputation	Metric Learning	CoxPH
500	5	0.15	0	1.747	1.619
500	5	0.15	1	0.622	1.617
500	5	0.15	5	0.600	1.614
500	5	0.25	0	2.204	1.618
500	5	0.25	1	0.660	1.619
500	5	0.25	5	0.640	1.619
500	5	0.50	0	2.892	1.624
500	5	0.50	1	0.877	1.622
500	5	0.50	5	0.799	1.626
500	10	0.15	0	2.225	1.627
500	10	0.15	1	0.698	1.621
500	10	0.15	5	0.644	1.626
500	10	0.25	0	2.932	1.628
500	10	0.25	1	0.727	1.625
500	10	0.25	5	0.676	1.626
500	10	0.50	0	4.723	1.637
500	10	0.50	1	0.914	1.639
500	10	0.50	5	0.817	1.638
500	20	0.15	0	3.061	1.649
500	20	0.15	1	1.183	1.644
500	20	0.15	5	1.206	1.644
500	20	0.25	0	3.679	1.651
500	20	0.25	1	1.254	1.648
500	20	0.25	5	1.249	1.648
500	20	0.50	0	5.466	1.664
500	20	0.50	1	1.480	1.669
500	20	0.50	5	1.443	1.664

Table 4.3: Simulation results for nonlinear model, n=500

method of [64], which is adapted to right-censored survival data.

Harrell's c-index is calculated by considering appropriately chosen pairs of observations and assessing the proportion of times an estimator correctly determines which of the pair of observations should experience the event of interest first. An estimator that is able to correctly rank observations in terms of event times would be perform well according to Harrell's c-index. However, accurate rankings are not equivalent to accurate predictions of survival times. Crucially, Harrell's c-index does not directly compare the predicted survival times with the observed survival

n	p	Censoring Rate	Rounds of Imputation	Metric Learning	CoxPH
1000	5	0.15	0	1.603	1.616
1000	5	0.15	1	0.603	1.612
1000	5	0.15	5	0.591	1.611
1000	5	0.25	0	1.753	1.612
1000	5	0.25	1	0.641	1.614
1000	5	0.25	5	0.629	1.614
1000	5	0.50	0	2.243	1.618
1000	5	0.50	1	0.818	1.617
1000	5	0.50	5	0.775	1.620
1000	10	0.15	0	2.052	1.617
1000	10	0.15	1	0.617	1.620
1000	10	0.15	5	0.592	1.616
1000	10	0.25	0	2.631	1.615
1000	10	0.25	1	0.659	1.622
1000	10	0.25	5	0.629	1.620
1000	10	0.50	0	4.187	1.628
1000	10	0.50	1	0.832	1.622
1000	10	0.50	5	0.764	1.622
1000	20	0.15	0	2.873	1.625
1000	20	0.15	1	0.931	1.625
1000	20	0.15	5	0.954	1.625
1000	20	0.25	0	3.764	1.627
1000	20	0.25	1	0.974	1.628
1000	20	0.25	5	0.993	1.630
1000	20	0.50	0	5.970	1.638
1000	20	0.50	1	1.213	1.643
1000	20	0.50	5	1.157	1.638

Table 4.4: Simulation results for nonlinear model, n=1,000

times. In contrast, the method of [64] uses an inverse-probability of censoring weighted estimate of the RMSE. The method of [64] may also be deemed more appropriate than Harrell's c-index as the conditional mean ,which we are estimating, minimizes the expected squared error. We use both of these methods to evaluate the predictive performance of metric learning, as well as the AFT model and CoxPH model, on data sets.

4.4.1 Serum Free Light Chain Data Set

We apply metric learning to the subset of the `flchain` data set from the **R** package, **survival** for which creatinine measurements were available. Subjects with reported survival time of 0 were also removed as the log-survival survival time of such observations would be $-\infty$. The event of interest was all-cause mortality. We chose to estimate the conditional mean restricted to 5 years. The sample size was 6521. 24% of the restricted survival times were censored. Our analysis will use serum free light chain tests, creatinine, age, and year at which the test was taken to predict survival time.

To calculate Harrell's c-index and the $RMSE$ we divided the data set into 5 groups. Each of the 5 divisions of the data set was used once as a test set and four times as a training set. This same procedure was followed to evaluate the predictive performance of the CoxPH model and a log-normal AFT model.

Estimates of the $RMSE$ are displayed in Table 4.5. The CoxPH model yields the best performance according to Harrell's c-index. This result is perhaps to be expected as the CoxPH's partial likelihood itself depends on the survival times through the ranks of the survival times. On the other hand, metric learning with 5 rounds of imputation yielded the lowest estimated RMSE.

Method	Harrell's c-Index	RMSE
CoxPH Model	0.21	0.83
Log Normal AFT Model	0.21	1.43
Metric Learning (No Imputation)	0.69	10.56
Metric Learning (Imputation)	0.23	0.82

Table 4.5: RMSE and Harrell's c-Index of methods for `flchain` data set.

4.4.2 Sample of Non-small-cell Lung Cancer Cases from the SEER Database

We now examine the performance of metric learning on a subset of subjects from the Surveilance, Epidemiology, and End Results (SEER) database. We considered a subset of patients diagnosed with non-small-cell lung cancer. Non-small-cell lung cancer is the most commonly diagnosed type of lung cancer. In particular, we considered malignant neoplasm of the main bronchus from the years 2004-2014. The sample size was 5889 with a censoring rate of 11%. Covariates included race, sex, age at diagnosis, year of birth, year of diagnosis, and stage (adjusted AJCC 6th edition).

As seen in Table 4.6, metric learning without imputation is competitive with the CoxPH and AFT model according to its estimated RMSE, although in this case the estimated RMSE for the CoxPH model and AFT model were slightly lower. The AFT model had the strongest performance as measured by both Harrell's c-index and RSME. Interestingly enough, metric learning performs worse with imputation than without imputation. In some cases when metric learning without imputation performs well, we have found that iterative imputation can lead to overfitting.

Method	Harrell's c-Index	RMSE
CoxPH Model	0.37	1.09
Log Normal AFT Model	0.37	1.09
Metric Learning (No Imputation)	0.37	1.10
Metric Learning (Imputation)	0.42	1.20

Table 4.6: RMSE and Harrell's c-Index of methods for subset of SEER data set.

4.4.3 Time Until Weaning for Breast-fed Newborns

Finally, we compared metric learning to the CoxPH model and AFT model on a data set from [99] in which the time until weaning from breast-fed newborns was recorded for a sample of 927 woman. The censoring rate is 4%. The data was obtained via the **R** package **KMsurv** on **CRAN**. See Section 1.14 of [99] and documentation for the data set `bfeed` in the **KMsurv** package for details concerning each covariate. We used all explanatory covariates in the data set.

Method	Harrell's Index	RMSE
CoxPH Model	0.21	0.86
Log Normal AFT Model	0.23	0.85
Metric Learning (No Imputation)	0.39	2.60
Metric Learning (Imputation)	0.23	0.78

Table 4.7: RMSE and Harrell's c-Index of methods for subset of data set on time until weaning for breast-fed newborns.

4.5 Discussion

In this article, we introduced an extension of metric learning to right censored survival data. The algorithm involves applying a synthetic variable transformation to the right-censored survival times. Metric learning is applied using these synthetic times, producing an initial estimator of f. Given this initial estimator, we impute the right-censored survival times. This process is carried out until the estimate of M has converged.

We have seen in simulations and on data sets that metric learning performs well when the sample size is large, the number of covariates is small to moderate in comparison, and when the censoring rate is low to moderate. Our simulations indicate that performance seriously degrades when the censoring rate is high. Computational speed is a primary drawback of metric learning. Each iteration of gradient descent requires calculation of the Mahalanobis distance between all pairs of observation. Each observation's contribution to gradient also requires the calculation n outer-products of p dimensional vectors. When either n or p is large, metric learning can become prohibitively slow. Further research into increasing computational time is required.

CHAPTER 5

Discussion

5.1 Further Extensions of Random Forests and Fuzzy Forests

5.1.1 Correcting the Bias in Fuzzy Forests VIMs

As demonstrated in the above simulations, fuzzy forests select features in such a way that correlated covariates are much less likely to be unfairly favored over independent covariates. In terms of feature selection, fuzzy forests perform comparably to conditional inference forests and do so within computationally feasible times. However, for reasons discussed in Chapter 2, within the selected group of covariates, correlated VIMs may be unduly large.

Once again, the target estimand of the random forest VIM is given by the following expression:

$$VIM(v) = E(f(X_i^{(1)}, \ldots, X_i^{(v)}, \ldots, X_i^{(p)}) - f(X_i^{(1)}, \ldots, \tilde{X}_i^{(v)}, \ldots, X_i^{(p)}))^2. \quad (5.1)$$

Noting that the estimate of the VIM relies on an estimate of f, if the estimator $\hat{f}$ is biased, the estimate for the VIM will be biased as well. We suspect that VIMs given by fuzzy forests are biased for two primary reasons. First, if important features are left out of the final set of selected features and these features are correlated with features that have been selected, the VIMs of the selected features will be biased due to confounding. Second, even if the correct features are selected, regression trees still yield biased estimates of f. Our simulations demonstrate this to be the case when the true regression model is linear and covariates are correlated.

To explore the extent to which regression trees are biased and to look for a possible solution, we carried out a simulation study with two variants of regression trees. The first variant of regression trees splits each node along the coordinate axes. In particular, the trees were fit using the `ctree` function of the **party** package. The second variant splits nodes along hyperplanes. A hyperplane in p dimensions is a set of the form $\{\sum_{k=1}^{p} a_k X_k = c\}$ for constants $a_k (k = 1, \ldots, p)$ and c. Consider a node, $\tau \subset \{1, \ldots, n\}$, representing a subset of n observations. The node τ is said to be split along a hyperplane if it is split into two nodes $\tau_l = \{i : \sum_{k=1}^{p} a_k X_i^{(k)} \leq c, i \in \tau\}$ and $\tau_r = \{i : \sum_{k=1}^{p} a_k X_i^{(k)} > c, i \in \tau\}$.

The strategy of splitting along hyperplanes was proposed early in the literature on CART, largely in the context of classification [100]. More recently, splitting along hyperplanes has been presented as a successful strategy for increasing predictive performance in [43]. Choosing which hyperplane to split a node along is more complex than simply splitting along coordinate axes.

In this section, we test an algorithm that splits nodes along hyperplanes in the spirit of the one presented in [43] and test its ability to estimate VIMs. Namely, the hyperplane is determined by fitting the LASSO to all observations in a given node. Ten fold cross-validation was used to determine the size of the L_1 penalty. Once the hyperplane was determined by the LASSO, nodes were split until a minimum split size was reached.

In this simulation, to eliminate potential effects due to sparse data at the tails of the distribution of covariates, the covariates were derived by first generating data uniformly distributed between 0 and 1. The first 4 covariates were linearly transformed and re-scaled to have compound symmetry covariance matrix with correlation 0.8. The additional covariates were independent and were re-scaled to have variance 1. The true model was linear $Y_i = X'\gamma$ with $\gamma_1 = \gamma_2 = \gamma_4 = \gamma_5 = 5$

and $\gamma_3 = \gamma_7 = 2$. The other elements of γ were equal to 0. The sample size was varied from 500, 1,000, to 2,000. In each simulation, $p - 7$ independent noise covariates were included. We display the VIMs for the first 8 covariates. The true VIMs are displayed in the Table 5.1.

The results of the simulations are split by sample size across Tables 5.2, 5.3, and 5.4. The type of regression tree used is given in the column labeled "methods", where "lmtree" denotes trees with splits along hyperplanes and "ctree" with splits along coordinate axes. In nearly all settings, trees with splits along hyperplanes provided better estimates of the VIMs than trees with splits along coordinate axes. The trees split along hyperplanes were much closer to estimating the VIMs of covariates with coefficients 2. This was particularly the case when the minimum sample size required to split a node was at 50. It is also important to note that the VIMs for the trees split along hyperplanes do not appear to be biased in favor of the correlated covariates.

These simulations suggest that cutting nodes along hyperplanes rather than coordinate axes leads to less biased estimates of VIMs. We plan on further investigating computationally efficient methods for growing trees with splits along hyperplanes. Along the same lines, we will explore the performance of regression trees with a linear model fit to each leaf. Using a single index regression model to determine the hyperplanes on which to split or fitting a single index regression model within leaves may also perform well. It will also be important to decide on a set of simulations that appropriately reflect advantages and disadvantages of these regression trees methods. For example, this simulation assumed a linear model, therefore, it should be expected that splitting nodes along hyperplanes will lead to more accurate estimates of VIMs.

Feature	True VIM
X_1	50
X_2	50
X_3	8
X_4	0
X_5	50
X_6	50
X_7	8
X_8	0

Table 5.1: True VIMs for simulation study

n	p	min_split	method	X1	X2	X3	X4	X5	X6	X7	X8
500	12	50	lmtree	50.4	50.5	10.2	0.0	49.9	50.1	10.2	0.0
500	12	50	ctree	87.8	100.2	2.8	0.1	37.9	40.9	0.1	0.0
500	12	100	lmtree	57.5	56.7	15.5	0.0	54.4	54.7	15.1	0.0
500	12	100	ctree	87.0	104.4	1.2	0.0	21.9	23.5	0.0	0.0
500	12	250	lmtree	75.7	73.5	24.9	0.0	66.7	66.7	24.2	0.0
500	12	250	ctree	91.6	83.8	0.0	0.0	1.0	0.2	0.0	0.0
500	20	50	lmtree	50.3	50.0	10.2	0.0	49.6	49.5	10.0	0.0
500	20	50	ctree	88.6	97.9	2.9	0.2	38.4	39.2	0.1	0.0
500	20	100	lmtree	57.4	56.2	15.5	0.0	54.9	55.1	15.0	0.0
500	20	100	ctree	96.3	94.6	1.1	0.0	24.1	23.7	0.0	0.0
500	20	250	lmtree	77.3	73.7	25.0	0.0	67.8	66.6	24.9	0.0
500	20	250	ctree	96.9	80.4	0.7	0.0	1.5	0.6	0.0	0.0
500	30	50	lmtree	50.4	50.2	10.4	0.0	49.8	50.0	10.2	0.0
500	30	50	ctree	94.9	95.0	2.1	0.2	39.0	39.6	0.0	0.0
500	30	100	lmtree	57.0	55.9	15.3	0.0	54.2	55.0	15.1	0.0
500	30	100	ctree	101.9	84.2	2.7	0.3	23.2	22.6	0.0	0.0
500	30	250	lmtree	76.7	73.8	25.0	0.0	66.0	65.4	24.0	0.0
500	30	250	ctree	100.3	71.6	1.3	0.0	0.5	0.8	0.0	0.0

Table 5.2: Regression tree VIMs with splits along coordinate axes versus regression tree VIMs with splits along hyperplanes, $n = 500$

n	p	min_split	method	X1	X2	X3	X4	X5	X6	X7	X8
1000	12	50	lmtree	48.8	48.8	9.1	0.0	48.3	48.6	9.3	0.0
1000	12	50	ctree	95.9	81.0	2.0	0.1	45.8	46.6	0.5	0.0
1000	12	100	lmtree	50.4	50.7	10.3	0.0	49.9	50.1	10.2	0.0
1000	12	100	ctree	91.6	88.9	0.9	0.0	38.7	37.8	0.0	0.0
1000	12	250	lmtree	58.8	58.1	15.7	0.0	55.1	54.9	15.4	0.0
1000	12	250	ctree	91.5	90.4	0.1	0.0	17.2	12.2	0.0	0.0
1000	20	50	lmtree	48.4	48.9	9.2	0.0	49.0	48.5	9.2	0.0
1000	20	50	ctree	90.0	82.3	2.3	0.2	45.3	45.2	0.4	0.0
1000	20	100	lmtree	50.6	50.7	10.2	0.0	50.0	50.4	10.0	0.0
1000	20	100	ctree	89.3	90.5	0.4	0.0	38.8	39.2	0.0	0.0
1000	20	250	lmtree	58.6	57.5	15.9	0.0	54.9	54.8	15.5	0.0
1000	20	250	ctree	91.7	88.9	0.3	0.0	14.5	12.8	0.0	0.0
1000	30	50	lmtree	48.6	48.9	9.2	0.0	48.5	48.5	9.3	0.0
1000	30	50	ctree	93.1	84.0	2.0	0.1	45.4	45.9	0.4	0.0
1000	30	100	lmtree	50.7	50.3	10.1	0.0	50.1	50.1	10.0	0.0
1000	30	100	ctree	90.7	88.2	0.5	0.0	37.5	39.5	0.0	0.0
1000	30	250	lmtree	58.7	57.3	15.6	0.0	55.0	55.2	15.2	0.0
1000	30	250	ctree	87.5	93.1	0.4	0.0	15.5	13.4	0.0	0.0

Table 5.3: Regression tree VIMs with splits along coordinate axes versus regression tree VIMs with splits along hyperplanes, $n = 1,000$

5.2 Future Work on Cross-Validation and Model Selection

5.2.1 Further Application of General Optimization Techniques to Minimizing the Cross-Validation Criterion

The idea of using general optimization techniques to minimize the cross-validation may have application outside of Nadaraya-Watson kernel regression estimator. First, it may be possible to extend the method to case of a local linear kernel regression estimator. Beyond kernel regression, a similar gradient descent algorithm could be applied to ridge regression estimators and certain estimators of generalized additive models have analytic forms. Optimization methods for simultaneous estimation of tuning parameters have been introduced in the context of minimizing the generalized cross-validation criterion [101, 102, 103, 104].

The ridge regression estimator is defined as $\hat{\beta}(\lambda) = \text{argmin}_\beta(\sum_{i=1}^{n}(Y_i - X_i'\beta) +$

n	p	min_split	method	X1	X2	X3	X4	X5	X6	X7	X8
2000	12	50	lmtree	48.4	48.6	8.8	0.0	48.1	48.3	8.8	0.0
2000	12	50	ctree	94.0	77.8	1.8	0.1	47.9	48.6	1.0	0.0
2000	12	100	lmtree	48.6	48.7	9.1	0.0	48.6	48.5	9.0	0.0
2000	12	100	ctree	94.3	78.9	0.9	0.0	44.5	44.2	0.1	0.0
2000	12	250	lmtree	51.5	51.3	10.8	0.0	50.5	50.1	10.7	0.0
2000	12	250	ctree	92.2	83.7	0.0	0.0	31.7	33.3	0.0	0.0
2000	20	50	lmtree	48.5	48.3	8.8	0.0	48.2	48.2	8.7	0.0
2000	20	50	ctree	98.0	76.7	1.7	0.1	49.0	48.2	1.1	0.0
2000	20	100	lmtree	49.0	48.5	9.1	0.0	48.7	48.5	9.1	0.0
2000	20	100	ctree	95.7	76.7	0.9	0.0	44.1	44.5	0.1	0.0
2000	20	250	lmtree	51.6	51.1	11.0	0.0	50.5	50.7	10.8	0.0
2000	20	250	ctree	91.2	86.6	0.0	0.0	33.2	33.1	0.0	0.0
2000	30	50	lmtree	48.6	48.7	8.9	0.0	48.1	48.2	8.7	0.0
2000	30	50	ctree	99.7	76.5	1.8	0.0	48.3	48.4	1.1	0.0
2000	30	100	lmtree	49.0	49.1	9.2	0.0	48.6	48.9	9.1	0.0
2000	30	100	ctree	100.9	73.8	0.7	0.0	43.7	45.1	0.1	0.0
2000	30	250	lmtree	51.4	51.2	10.9	0.0	50.3	50.8	10.8	0.0
2000	30	250	ctree	97.1	79.1	0.0	0.0	32.9	32.6	0.0	0.0

Table 5.4: Regression tree VIMs with splits along coordinate axes versus regression tree VIMs with splits along hyperplanes, $n = 2,000$

$\sum_{j=1}^{p} \lambda\beta_j^2)$[105]. The ridge regression estimator depends on a single tuning parameter λ which controls the bias-variance tradeoff. Intuitively, the ridge regression estimator could be improved by letting each covariate have its own penalty, λ_j. Unimportant covariates should receive higher penalties than important covariates. This ridge regression estimator is defined as $\hat{\beta}(\lambda) = \text{argmin}_{\beta}(\sum_{i=1}^{n}(Y_i - X_i'\beta) + \sum_{j=1}^{p} \lambda_j\beta_j^2)$. When p is moderately large, perhaps greater than 10, specifying a grid of potential values λ_j and minimizing the cross-validation criterion over this grid may be impractical. In this case, the use of a more sophisticated algorithm for optimizing the cross-validation criterion is necessary. As certain estimators of generalized additive models (GAMs) are closely related to ridge regression estimators [33], the same principle applies to GAMs.

Direct application of Theorem 1 to the ridge regression estimator or a closely related generalized additive model is not possible as the condition $sup_X|\phi_k(X|P_n)| \leq$

$M < \infty$ and condition A3 no longer hold. Adjusting the arguments of Theorem 1 so that it applies to a wider variety of estimators is another avenue of research I am interested in. Extending Theorem 1 to right-censored survival data is another possible direction in which to generalize Theorem 1.

Appendix A

Further Technical Results for Chapter 3

This supplemetary material contains the proof of Lemma 2 as well as the derivative of K-fold cross-validation criterion with respect to the bandwidth matrix H.

A.1 Proof of Lemma 2

Proof. For brevity, let $k_{i,H}(x) = K(H^{1/2}(X_i - x))$. Let

$$\tilde{\psi}_n(x) = \frac{\sum_{i=1}^n k_{i,h}(x)\psi(X_i)}{\sum_{i=1}^n k_{i,h}(x)}.$$

We have the following decomposition (obtained by adding and subtracting $\tilde{\psi}_n(x)$, expanding, and noting that the cross-product term is 0)

$$\begin{aligned}&E[(\psi_n(X) - \psi(X))^2|X_1, \ldots, X_n] \\ &= E[(\tilde{\psi}_n(x) - \psi(x))^2|X_1, \ldots, X_n] + (\tilde{\psi}_n(x) - \psi(x))^2.\end{aligned}$$

Letting μ be the distribution of the covariates, by the above decomposition, we have

$$
\begin{aligned}
&E[(\psi_n(X) - \psi(X))^2] \\
&= E_{\{X_1,\ldots,X_n\}}[\int E[(\psi_n(x) - \psi(x))^2 | X_1, \ldots, X_n]\mu(x)] \\
&= E_{\{X_1,\ldots,X_n\}}[\int E[(\tilde{\psi}_n(x) - \psi(x))^2 | X_1, \ldots, X_n]\mu(x)] \\
&+ E[\int (\tilde{\psi}_n(x) - \psi(x))^2 \mu(dx)] \\
&= \int E_{\{X_1,\ldots,X_n\}} \Big[E[(\tilde{\psi}_n(x) - \psi(x))^2 | X_1, \ldots, X_n] \Big] \mu(x) \qquad \text{(A.1)} \\
&+ E[\int (\tilde{\psi}_n(x) - \psi(x))^2 \mu(dx)] \qquad \text{(A.2)}
\end{aligned}
$$

Consider the second term (A.2): $E[\int(\tilde{\psi}_n(x) - \psi(x))^2\mu(dx)]$. We have

$$
\begin{aligned}
&(\tilde{\psi}_n(x) - \psi(x))^2 \\
&= \Big(\frac{\sum_{i=1}^n (\psi(X_i) - \psi(x)) k_{i,h}(x)}{\sum_{i=1}^n k_{i,h}(x)}\Big)^2 \\
&\leq \frac{\sum_{i=1}^n (\psi(X_i) - \psi(x))^2 k_{i,h}(x)}{\sum_{i=1}^n k_{i,h}(x)} \\
&\leq R^2 \frac{\sum_{i=1}^n ||X_i - x||^2 k_{i,h}(x)}{\sum_{i=1}^n k_{i,h}(x)},
\end{aligned}
$$

where the first inequality is due to Jensen's inequality and the second is due to the assumption that ψ is Lipshitz continuous with Lipshitz constant R. We now consider the class of truncated Gaussian kernels, $k_{i,h}^{T_n}(x) = \exp(-h||x - X_i||^2)I_{\{\sqrt{h}||x-X_i|| \leq T_n\}}$, where we will take $T_n = \sqrt{\log(n)}$. Note that $k_{i,h}^{T_n}(x) \leq k_{i,h}(x)$ and $k_{i,h}(x) - k_{i,h}^{T_n}(x) \leq \exp(-T_n^2)$.

We then have

$$
\begin{aligned}
&(\tilde{\psi}_n(x) - \psi(x))^2 \\
&\le R^2 \frac{\sum_{i=1}^n ||X_i - x||^2 k_{i,h}(x)}{\sum_{i=1}^n k_{i,h}(x)} \\
&= R^2 \frac{\sum_{i=1}^n ||X_i - x||^2 k_{i,h}(x)}{\sum_{i=1}^n k_{i,h}(x)} - R^2 \frac{\sum_{i=1}^n ||X_i - x||^2 k_{i,h}^{T_n}(x)}{\sum_{i=1}^n k_{i,h}^{T_n}(x)} \qquad (A.3) \\
&+ R^2 \frac{\sum_{i=1}^n ||X_i - x||^2 k_{i,h}^{T_n}(x)}{\sum_{i=1}^n k_{i,h}^{T_n}(x)}. \qquad (A.4)
\end{aligned}
$$

The second term in the above expression, in (A.4), can be bound as follows:

$$
R^2 \frac{\sum_{i=1}^n ||X_i - x||^2 k_{i,h}^{T_n}(x)}{\sum_{i=1}^n k_{i,h}^{T_n}(x)} \le R^2 \frac{T_n^2}{h}, \qquad (A.5)
$$

because $||X_i - x||^2 k_{i,h}^{T_n}(x) \le (T_n^2/h) k_{i,h}^{T_n}(x)$.

Now consider the term in (A.3). We have

$$
\begin{aligned}
&R^2 \frac{\sum_{i=1}^n ||X_i - x||^2 k_{i,h}(x)}{\sum_{i=1}^n k_{i,h}(x)} - R^2 \frac{\sum_{i=1}^n ||X_i - x||^2 k_{i,h}^{T_n}(x)}{\sum_{i=1}^n k_{i,h}^{T_n}(x)} \\
&\le \frac{R^2 \sum_{i=1}^n ||X_i - x||^2 (k_{i,h}(x) - k_{i,h}^{T_n}(x))}{\sum_{i=1}^n k_{i,h}(x)} \\
&\le R^2 B^2 \frac{\sum_{i=1}^n (k_{i,h}(x) - k_{i,h}^{T_n}(x))}{\sum_{i=1}^n k_{i,h}(x)} \\
&= R^2 B^2 \min\{1, \frac{\sum_{i=1}^n (k_{i,h}(x) - k_{i,h}^{T_n}(x))}{\sum_{i=1}^n k_{i,h}(x)}\} \\
&\le R^2 B^2 \min\{1, \frac{n \exp(-T_n^2)}{\sum_{i=1}^n k_{i,h}(x)}\} \\
&\le \frac{R^2 B^2}{b} \min\{1, \frac{1}{\sum_{i=1}^n I_{\{X_i \in S_{x, r/\sqrt{h}}\}}}\} \\
&\le \frac{R^2 B^2}{b} \frac{2}{1 + \sum_{i=1}^n I_{\{X_i \in S_{x, r/\sqrt{h}}\}}}.
\end{aligned}
$$

The last inequality can be seen by observing that $1/u \le 2/(1+u)$ for $u \ge 1$.

By the previous inequality and (A.5), we have

$$
\begin{aligned}
&E[\int(\tilde{\psi}_n(x)-\psi(x))^2\mu(dx)]\\
&=\int E[(\tilde{\psi}_n(x)-\psi(x))^2]\mu(dx)\\
&\leq R^2\frac{T_n^2}{h}+\frac{R^2B^2}{b}\int E[\frac{2}{1+\sum_{i=1}^n I_{\{X_i\in S_{x,r/\sqrt{h}}\}}}]\mu(dx)\\
&\leq R^2\frac{T_n^2}{h}+\frac{2R^2B^2}{b}\int\frac{1}{(n+1)\mu(S_{x,r/\sqrt{h}})}\mu(dx)\\
&\leq R^2\frac{T_n^2}{h}+\frac{2R^2B^2}{b}\frac{\tilde{c}h^{p/2}}{n+1}=R^2\frac{\log(n)}{h}+\frac{2R^2B^2}{b}\frac{\tilde{c}h^{p/2}}{n+1} &\text{(A.6)}\\
&\leq R^2\frac{\log(n)}{h}+\frac{2R^2B^2}{b}\frac{\tilde{c}h^{p/2}}{n}, &\text{(A.7)}
\end{aligned}
$$

The second and third inequalities follow by Lemma 4.1 of [49] and a result in the middle of page 76 (5.1) of [49], respectively. Here, $\tilde{c}$ is a constant depends on B as in the result on page 76 of [49].

Next we calculate an upper-bound for (A.1)

$$\int E_{\{X_1,\ldots,X_n\}}\Bigg[E[(\tilde{\psi}_n(x)-\psi(x))^2|X_1,\ldots,X_n]\Bigg]\mu(x)$$

Considering the inner most expectation, we have

$$
\begin{aligned}
&E[(\tilde{\psi}_n(x)-\psi(x))^2|X_1,\ldots,X_n]\\
&=E\Bigg[\Bigg(\frac{\sum_{i=1}^n(Y_i-\psi(X_i))k_{i,h}(x)}{\sum_{i=1}^n k_{i,h}(x)}\Bigg)^2|X_1,\ldots,X_n\Bigg]\\
&=\frac{\sum_{i=1}^n \mathrm{Var}(Y_i|X_i)k_{i,h}^2(x)}{(\sum_{i=1}^n k_{i,h}(x))^2}\\
&\leq\sigma^2\frac{\sum_{i=1}^n k_{i,h}^2(x)}{(\sum_{i=1}^n k_{i,h}(x))^2}\\
&=\sigma^2\sum_{i=1}^n W_{i,h}^2(x),
\end{aligned}
$$

where we define $W_{i,h}(x)=k_{i,h}(x)/(\sum_{i=1}^n k_{i,h}(x))$. Note that $0<W_{i,h}(x)<1$, thus, $\sum_{i=1}^n W_{i,h}^2(x) \le \sum_{i=1}^n W_{i,h}(x) = 1$.

An upper bound for $\sum_{i=1}^n W_{i,h}^2(x)$ is found in a manner analogous to that of (A.3):

$$\begin{aligned}
&\sum_{i=1}^n W_{i,h}^2(x) \\
&= \min\{1, \frac{\sum_{i=1}^n k_{i,h}^2(x)}{(\sum_{i=1}^n k_{i,h}(x))^2}\} \\
&\le \min\{1, \frac{\sum_{i=1}^n k_{i,h}(x)}{(\sum_{i=1}^n k_{i,h}(x))^2}\} \\
&= \min\{1, \frac{1}{(\sum_{i=1}^n k_{i,h}(x))}\} \\
&\le \frac{1}{b}\min\{1, \frac{1}{\sum_{i=1}^n I_{\{X_i \in S_{x,r/\sqrt{h}}\}}}\} \\
&\le \frac{2}{b}\frac{1}{1+\sum_{i=1}^n I_{\{X_i \in S_{x,r/\sqrt{h}}\}}}.
\end{aligned}$$

Thus, we have an upper bound for (A.1)

$$\begin{aligned}
&\int E_{\{X_1,\dots,X_n\}}\bigg[E[(\tilde{\psi}_n(x)-\psi(x))^2|X_1,\dots,X_n]\bigg]\mu(x) \\
&\le \int E_{\{X_1,\dots,X_n\}}\bigg[\frac{2}{b}\frac{\sigma^2}{1+\sum_{i=1}^n I_{\{X_i \in S_{x,r/\sqrt{h}}\}}}\bigg]\mu(dx) \\
&\le \int \frac{2\sigma^2}{b(n+1)\mu(S_{x,r/\sqrt{h}})}\mu(dx) \\
&\le \frac{2\sigma^2\tilde{c}h^{p/2}}{nb}. \qquad\qquad (A.8)
\end{aligned}$$

Combining (A.1), (A.2), (A.7) and (A.8), we have

$$\int E[(\psi_n(x)-\psi(x))^2]\mu(dx) \le \frac{R^2\log(n)}{h} + \frac{2R^2B^2\tilde{c}h^{p/2}}{nb} + \frac{2\sigma^2\tilde{c}h^{p/2}}{nb} \qquad (A.9)$$

Setting the derivative of the right hand side of (A.9) to 0 and solving for h, we find that the minimizer of the above expression is

$$h_n^* = \left(\frac{A_1 \log(n) n}{A_2 \frac{p}{2}}\right)^{\frac{2}{p+2}},$$

where $A_1 = R^2$ and $A_2 = \frac{2\tilde{c}}{b}(R^2B^2 + \sigma^2)$. If we let the bandwidth increase at the above rate, by plugging h_n^* back into (A.9), we obtain the following upper-bound on the rate of convergence of the kernel regression estimator

$$\int E[(\psi_n(x) - \psi(x))^2]\mu(dx) \leq A \log(n)^{\frac{p}{p+2}} n^{\frac{-2}{p+2}},$$

where

$$A = A_1^{\frac{p}{p+2}} A_2^{\frac{2}{p+2}} \left((p/2)^{\frac{2}{p+2}} + (p/2)^{\frac{p}{p+2}}\right).$$

□

A.2 Derivation of Gradient for Kernel Regression

This section gives a derivation of the gradient of $L(y, \psi_H(x|P_n)) - L(y, \psi)$. First let the covariates for the ith observation be $X_i' = (X_{i1}, \dots, X_{ip})$ and let $x = (x_1, \dots, x_p)$. First of all we have

$$\frac{\partial L(y, \psi_H(x|P_n)) - L(y, \psi)}{\partial H_{uv}} = \frac{\partial (\psi_H(x|P_n) - y)^2}{\partial H_{uv}}.$$

We have

$$\frac{\partial (\psi_H(x|P_n) - y)^2}{\partial H_{uv}} = 2(\psi_H(x|P_n) - y)\frac{\partial \psi_H(x|P_n)}{\partial H_{uv}}.$$

And so we calculate $\frac{\partial \psi_H(x|P_n)}{\partial H_{uv}}$.

Again, for brevity, let $k_{i,H}(x) = K(H^{1/2}(X_i - x))$. We use the quotient rule:

$$\frac{\partial \psi_H(x|P_n)}{\partial H_{uv}} = \frac{(\sum_{i=1}^n \frac{\partial k_{i,H}(x)}{\partial H_{uv}} Y_i)(\sum_{i=1}^n k_{i,H}(x)) - (\sum_{i=1}^n k_{i,H}(x) Y_i)(\sum_{i=1}^n \frac{\partial k_{i,H}(x)}{\partial H_{uv}})}{(\sum_{i=1}^n k_{i,H}(x))^2}.$$

Thus, we have

$$\frac{\partial \psi_H(x|P_n)}{\partial H_{uv}} = \frac{(\sum_{i=1}^n Y_i \frac{\partial k_{i,H}(x)}{\partial H_{uv}}) - (\sum_{i=1}^n \frac{\partial k_{i,H}(x)}{\partial H_{uv}}) \psi_H(x|P_n)}{(\sum_{i=1}^n k_{i,H}(x))}$$
$$= \frac{\sum_{i=1}^n (Y_i - \psi_H(x|P_n)) \frac{\partial k_{i,H}(x)}{\partial H_{uv}}}{\sum_{i=1}^n k_{i,H}(x)}.$$

Then we calculate $\frac{\partial k_{i,H}(x)}{\partial H_{uv}}$. We first have

$$\frac{\partial k_{i,H}(x)}{\partial H_{uv}} = -\exp(-(X_i - x)' H (X_i - x)) \frac{\partial (X_i - x)' H (X_i - x)}{\partial H_{uv}}.$$

This then equals

$$\frac{\partial k_{i,H}(x)}{\partial H_{uv}} = -k_{i,H}(x) \frac{\partial (X_i - x)' H (X_i - x)}{\partial H_{uv}}, \text{where}$$

$$\frac{\partial (X_i - x)' H (X_i - x)}{\partial H_{uv}} = 2(X_{iu} - X_{ju})(X_{iv} - X_{jv}) \text{ for } u \neq v$$

and

$$(X_{iu} - X_{ju})^2 \text{ for } u = v.$$

Thus we have

$$\frac{\partial L(y, \psi_H(x|P_n)) - L(y, \psi(x))}{\partial H_{uv}}$$
$$= 4(\psi_H(x|P_n) - y) \frac{\sum_{i=1}^n (\psi_H(x|P_n) - Y_i) k_{i,H}(x) (X_{iu} - x_u)(X_{iv} - x_v)}{\sum_{i=1}^n k_{i,H}(x)}$$

for $u \neq v$ and when $u = v$ the above 4 is replaced 2.

References

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

[2] Strobl C, Boulesteix AL, Kneib T, Augustin T, Zeileis A. Conditional variable importance for random forests. *BMC Bioinformatics* 2008; **9**(1):307.

[3] Nicodemus KK, Malley JD. Predictor correlation impacts machine learning algorithms: Implications for genomic studies. *Bioinformatics* 2009; **25**(15):1884–1890.

[4] Nicodemus KK, Malley JD, Strobl C, Ziegler A. The behaviour of random forest permutation-based variable importance measures under predictor correlation. *BMC Bioinformatics* 2010; **11**(1):110.

[5] Conn D, Ngun T, Li G, Ramirez CM. Fuzzy forests: Extending random forest feature selection for correlated, high-dimensional data. *Journal of Statistical Software* 2018; (In Press).

[6] Guillaumin M, Verbeek J, Schmid C. Is that you? metric learning approaches for face identification. *Computer Vision, 2009 IEEE 12th International Conference on*, IEEE, 2009; 498–505.

[7] Weinberger KQ, Saul LK. Distance metric learning for large margin nearest neighbor classification. *The Journal of Machine Learning Research* 2009; **10**:207–244.

[8] Bellet A, Habrard A, Sebban M. A survey on metric learning for feature vectors and structured data 2013. ArXiv preprint arXiv:1306.6709.

[9] Weinberger KQ, Tesauro G. Metric learning for kernel regression. *International Conference on Artificial Intelligence and Statistics*, 2007; 612–619.

[10] Hardle W, Hall P, Ichimura H. Optimal smoothing in single-index models. *Annals of Statistics* 1993; **21**(1):157–178.

[11] Ichimura H. Semiparametric least squares (sls) and weighted sls estimation of single-index models. *Journal of Econometrics* 1993; **58**(1):71–120.

[12] Hristache M, Juditsky A, Polzehl J, Spokoiny V, *et al.*. Structure adaptive approach for dimension reduction. *Annals of Statistics* 2001; **29**(6):1537–1566.

[13] Polzehl J, Sperlich S. A note on structural adaptive dimension reduction. *Journal of Statistical Computation and Simulation* 2009; **79**(6):805–818.

[14] Koul H, Susarla V, Van Ryzin J. Regression analysis with randomly right-censored data. *Annals of Statistics* 1981; **9**(6):1276–1288.

[15] Sue L. Linear models, random censoring and synthetic data. *Biometrika* 1987; **74**(2):301–309.

[16] Buckley J, James I. Linear regression with censored data. *Biometrika* 1979; **66**(3):429–436.

[17] Tibshirani R. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society B* 1996; **58**(1):267–288.

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

[19] Hastie T, Tibshirani R, Wainwright M. *Statistical Learning with Sparsity: the Lasso and Generalizations*. Chapman and Hall/CRC, 2015.

[20] Fan J, Li R. Variable selection via nonconcave penalized likelihood and its oracle properties. *Journal of the American Statistical Association* 2001; **96**(456):1348–1360.

[21] Breheny P, Huang J. Coordinate descent algorithms for nonconvex penalized regression, with applications to biological feature selection. *Annals of Applied Statistics* 2011; **5**(1):232.

[22] Raskutti G, Wainwright MJ, Yu B. Restricted eigenvalue properties for correlated Gaussian designs. *The Journal of Machine Learning Research* 2010; **11**:2241–2259.

[23] Dua S, Acharya UR, Dua P. *Machine Learning in Healthcare Informatics*. Springer-Verlag, 2014.

[24] Strobl C, Boulesteix AL, Zeileis A, Hothorn T. Bias in random forest variable importance measures: Illustrations, sources and a solution. *BMC Bioinformatics* 2007; **8**(1):25.

[25] Díaz-Uriarte R, De Andres SA. Gene selection and classification of microarray data using random forest. *BMC Bioinformatics* 2006; **7**(1):3.

[26] Langfelder P, Horvath S. WGCNA: an R package for weighted correlation network analysis. *BMC Bioinformatics* 2008; **9**(1):559.

[27] Zhang B, Horvath S. A general framework for weighted gene co-expression network analysis. *Statistical Applications in Genetics and Molecular Biology* 2005; **4**(1).

[28] Liaw A, Wiener M. Classification and regression by randomforest. *R news* 2002; **2**(3):18–22.

[29] Hothorn T, Bühlmann P, Dudoit S, Molinaro A, Van Der Laan MJ. Survival ensembles. *Biostatistics* 2006; **7**(3):355–373.

[30] Robert C. *The Bayesian choice: from decision-theoretic foundations to computational implementation.* Springer, 2007.

[31] Harrell FE. *Regression Modeling Strategies.* Springer, 2015.

[32] Hastie T, Tibshirani R, Friedman J, Hastie T, Friedman J, Tibshirani R. *The Elements of Statistical Learning.* Springer, 2009.

[33] Ruppert D, Wand MP, Carroll RJ. *Semiparametric Regression.* 12, Cambridge University Press, 2003.

[34] Wang Y. *Smoothing splines: methods and applications.* CRC Press, 2011.

[35] Cook RD, Weisberg S. *Applied regression including computing and graphics*, vol. 488. John Wiley & Sons, 2009.

[36] Team RC. *R: A Language and Environment for Statistical Computing.* Vienna, Austria 2015. URL `https://www.R-project.org/`.

[37] Mumford JA, Horvath S, Oldham MC, Langfelder P, Geschwind DH, Poldrack RA. Detecting network modules in fMRI time series: a weighted network analysis approach. *Neuroimage* 2010; **52**(4):1465–1476.

[38] Horvath S. *Weighted Network Analysis: Applications in Genomics and Systems Biology.* Springer-Verlag, 2011.

[39] Diaz-Uriarte R. GeneSrF and varSelRF: a web-based tool and R package for gene selection and classification using random forest. *BMC Bioinformatics* 2007; **8**(1):328.

[40] Jiang H, Deng Y, Chen HS, Tao L, Sha Q, Chen J, Tsai CJ, Zhang S. Joint analysis of two microarray gene-expression data sets to select lung adenocarcinoma marker genes. *BMC Bioinformatics* 2004; **5**(1):81.

[41] van der Laan MJ. Statistical inference for variable importance. *The International Journal of Biostatistics* 2006; **2**(1).

[42] Gregorutti B, Michel B, Saint-Pierre P. Correlation and variable importance in random forests. *Statistics and Computing* 2017; **27**(3):659–678.

[43] Zhu R, Zeng D, Kosorok MR. Reinforcement learning trees. *Journal of the American Statistical Association* 2015; **110**(512):1770–1784.

[44] Hunt PW, Brenchley J, Sinclair E, McCune JM, Roland M, Shafer KP, Hsue P, Emu B, Krone M, Lampiris H, *et al.*. Relationship between T cell activation and CD4+ T cell count in HIV-seropositive individuals with undetectable plasma HIV RNA levels in the absence of therapy. *Journal of Infectious Diseases* 2008; **197**(1):126–133.

[45] Ramirez C, Sinclair E, Epling L, Lee S, Jain V, Hsue P, Hatano H, Conn D, Hecht FM, Martin JN, *et al.*. Immunologic profiles distinguish aviremic HIV-infected adults. *AIDS* 2016; (In Press).

[46] Klatt NR, Bosinger SE, Peck M, Richert-Spuhler LE, Heigele A, Gile JP, Patel N, Taaffe J, Julg B, Camerini D, *et al.*. Limited HIV infection of central memory and stem cell memory CD4+ T cells is associated with lack of progression in viremic individuals. *PLoS Pathog* 2014; **10**(8):e1004 345.

[47] Nadaraya EA. On estimating regression. *Theory of Probability & Its Applications* 1964; **9**(1):141–142.

[48] Watson GS. Smooth regression analysis. *Sankhyā: The Indian Journal of Statistics, Series A* 1964; **26**(4):359–372.

[49] Györfi L, Kohler M, Krzyzak A, Walk H. *A Distribution-Free Theory of Nonparametric Regression*. Springer Science & Business Media, 2006.

[50] Härdle W, Müller M, Sperlich S, Werwatz A. *Nonparametric and Semiparametric Models*. Springer Science & Business Media, 2012.

[51] Wong WH. On the consistency of cross-validation in kernel nonparametric regression. *Annals of Statistics* 1983; **11**(4):1136–1141.

[52] Hall P. Asymptotic properties of integrated square error and cross-validation for kernel estimation of a regression function. *Probability Theory and Related Fields* 1984; **67**(2):175–196.

[53] Härdle W, Marron JS. Optimal bandwidth selection in nonparametric regression function estimation. *Annals of Statistics* 1985; **13**(4):1465–1481.

[54] Härdle W, Kelly G. Nonparametric kernel regression estimation-optimal choice of bandwidth. *Statistics: A Journal of Theoretical and Applied Statistics* 1987; **18**(1):21–35.

[55] Walk H. On cross-validation in kernel and partitioning regression estimation. *Statistics & Probability Letters* 2002; **59**(2):113–123.

[56] Dudoit S, van der Laan MJ. Asymptotics of cross-validated risk estimation in estimator selection and performance assessment. *Statistical Methodology* 2005; **2**(2):131–154.

[57] Weinberger KQ, Blitzer J, Saul LK. Distance metric learning for large margin nearest neighbor classification. *Advances in Neural Information Processing Systems*, 2005; 1473–1480.

[58] Kedem D, Tyree S, Sha F, Lanckriet GR, Weinberger KQ. Non-linear metric learning. *Advances in Neural Information Processing Systems*, 2012; 2573–2581.

[59] Ichimura H, Lee LF. Semiparametric least squares estimation of multiple index models: Single equation estimation. *Nonparametric and Semiparametric Methods in Econometrics and Statistics: Proceedings of the Fifth International Symposium in Economic Theory And Econometrics. Cambridge*, 1991; 3–49.

[60] Li KC. Sliced inverse regression for dimension reduction. *Journal of the American Statistical Association* 1991; **86**(414):316–327.

[61] Xia Y, Tong H, Li W, Zhu LX. An adaptive estimation of dimension reduction space. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 2002; **64**(3):363–410.

[62] Xia Y. Asymptotic distributions for two estimators of the single-index model. *Econometric Theory* 2006; **22**(6):1112–1137.

[63] Horowitz JL. *Semiparametric and Nonparametric Methods in Econometrics*, vol. 12. Springer, 2009.

[64] Keles S, Van Der Laan M, Dudoit S. Asymptotically optimal model selection method with right censored outcomes. *Bernoulli* 2004; **10**(6):1011–1037.

[65] Marron JS, Härdle W. Random approximations to some measures of accuracy in nonparametric curve estimation. *Journal of Multivariate Analysis* 1986; **20**(1):91–113.

[66] Klenke A. *Probability Theory: A Comprehensive Course*. Springer Science & Business Media, 2013.

[67] Boyd S, Vandenberghe L. *Convex Optimization*. Cambridge University Press, 2004.

[68] Kohavi R. A study of cross-validation and bootstrap for accuracy estimation and model selection. *Ijcai*, vol. 14, Stanford, CA, 1995; 1137–1145.

[69] James G, Witten D, Hastie T, Tibshirani R. *An Introduction to Statistical Learning*, vol. 112. Springer, 2013.

[70] Arlot S, Celisse A. A survey of cross-validation procedures for model selection. *Statistics Surveys* 2010; **4**:40–79.

[71] Antos A, Györfi L, Kohler M. Lower bounds on the rate of convergence of nonparametric regression estimates. *Journal of Statistical Planning and Inference* 2000; **83**(1):91–100.

[72] Burman P. A comparative study of ordinary cross-validation, v-fold cross-validation and the repeated learning-testing methods. *Biometrika* 1989; **76**(3):503–514.

[73] van der Vaart AW. *Asymptotic Statistics*, vol. 3. Cambridge University Press, 2000.

[74] Shalev-Shwartz S, Ben-David S. *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press, 2014.

[75] Duistermaat JJ, Kolk JA. *Multidimensional Real Analysis I: Differentiation*, vol. 86. Cambridge University Press, 2004.

[76] Beran R. Nonparametric regression with randomly censored survival data. *Technical Report*, University of California, Berkeley 1981.

[77] Dabrowska DM. Non-parametric regression with censored survival time data. *Scandinavian Journal of Statistics* 1987; **14**(3):181–197.

[78] Dabrowska DM. Uniform consistency of the kernel conditional Kaplan-Meier estimate. *Annals of Statistics* 1989; **17**(3):1157–1167.

[79] Li G, Doss H. An approach to nonparametric regression for life history data using local linear fitting. *Annals of Statistics* 1995; **23**(3):787–823.

[80] Li G, Datta S. A bootstrap approach to nonparametric regression for right censored data. *Annals of the Institute of Statistical Mathematics* 2001; **53**(4):708–729.

[81] Li G, Van Keilegom I. Likelihood ratio confidence bands in non–parametric regression with censored data. *Scandinavian Journal of Statistics* 2002; **29**(3):547–562.

[82] Dabrowska DM. Nonparametric quantile regression with censored data. *Sankhyā: The Indian Journal of Statistics, Series A* 1992; **54**(2):252–259.

[83] Gannoun A, Saracco J, Yuan A, Bonney GE. Non-parametric quantile regression with censored data. *Scandinavian Journal of Statistics* 2005; **32**(4):527–550.

[84] Heuchenne C, Van Keilegom I. Location estimation in nonparametric regression with censored data. *Journal of Multivariate Analysis* 2007; **98**(8):1558–1582.

[85] Doksum KA, Yandell BS. Properties of regression estimates based on censored survival data. *Festschrift for EL Lehmann*. Wadsworth, 1982; 140–156.

[86] Zheng Z. Strong consistency of nonparametric regression estimates with censored data. *Journal of Mathematical Research and Exposition*, vol. 8, 1988; 307–313.

[87] Carbonez A, Györfi L, Van der Meulen E. Partitioning-estimates of a regression function under random censoring. *Statistics & Risk Modeling* 1995; **13**(1):21–38.

[88] Kohler M, Máthé K, Pintér M. Prediction from randomly right censored data. *Journal of Multivariate Analysis* 2002; **80**(1):73–100.

[89] Kohler M. Nonparametric regression with additional measurement errors in the dependent variable. *Journal of Statistical Planning and Inference* 2006; **136**(10):3339–3361.

[90] Aalen O. A model for nonparametric regression analysis of counting processes. *Mathematical Statistics and Probability Theory*. Springer, 1980; 1–25.

[91] Aalen OO. Further results on the non-parametric linear regression model in survival analysis. *Statistics in Medicine* 1993; **12**(17):1569–1588.

[92] Xing EP, Jordan MI, Russell S, Ng AY. Distance metric learning with application to clustering with side-information. *Advances in Neural Information Processing Systems*, 2002; 505–512.

[93] Goldberger J, Hinton GE, Roweis ST, Salakhutdinov R. Neighbourhood components analysis. *Advances in Neural Information Processing Systems*, 2004; 513–520.

[94] Nguyen HV, Bai L. Cosine similarity metric learning for face verification. *Computer Vision–ACCV 2010*. Springer, 2011; 709–720.

[95] Hoi SC, Liu W, Lyu MR, Ma WY. Learning distance metrics with contextual constraints for image retrieval. *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, vol. 2, IEEE, 2006; 2072–2078.

[96] Wand MP, Jones MC. *Kernel Smoothing*. CRC Press, 1994.

[97] Royston P, Parmar MK. The use of restricted mean survival time to estimate the treatment effect in randomized clinical trials when the proportional hazards assumption is in doubt. *Statistics in Medicine* 2011; **30**(19):2409–2421.

[98] Harrell FE, Lee KL, Mark DB. Multivariable prognostic models: issues in developing models, evaluating assumptions and adequacy, and measuring and reducing errors. *Statistics in Medicine* 1996; **15**(4):361–387.

[99] Klein JP, Moeschberger ML. *Survival Analysis: Techniques for Censored and Truncated data*. Springer Science & Business Media, 2005.

[100] Breiman L, Friedman J, Stone CJ, Olshen RA. *Classification and Regression Trees*. CRC Press, 1984.

[101] Craven P, Wahba G. Smoothing noisy data with spline functions. *Numerische Mathematik* 1978; **31**(4):377–403.

[102] Wood SN. Modelling and smoothing parameter estimation with multiple quadratic penalties. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 2000; **62**(2):413–428.

[103] Wood SN. Stable and efficient multiple smoothing parameter estimation for generalized additive models. *Journal of the American Statistical Association* 2004; **99**(467):673–686.

[104] Wood SN. Fast stable direct fitting and smoothness selection for generalized additive models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 2008; **70**(3):495–518.

[105] Hoerl AE, Kennard RW. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics* 1970; **12**(1):55–67.