My Final College Paper

A Thesis
Presented to
The Division of Mathematics and Natural Sciences
Reed College

In Partial Fulfillment
of the Requirements for the Degree
Bachelor of Arts

Emerson H. Webb

May 2018

Acknowledgements

I want to thank a few people.

Preface

This is an example of a thesis setup to use the reed thesis document class (for LaTeX) and the R bookdown package, in general.

Table of Contents

List of Tables

List of Figures

Abstract

The preface pretty much says it all.

Second paragraph of abstract starts here.

Dedication

You can have a dedication here if you wish.

Introduction

Welcome to the *R Markdown* thesis template. This template is based on (and in many places copied directly from) the Reed College LaTeX template, but hopefully it will provide a nicer interface for those that have never used TeX or LaTeX before. Using *R Markdown* will also allow you to easily keep track of your analyses in **R** chunks of code, with the resulting plots and output included as well. The hope is this *R Markdown* template gets you in the habit of doing reproducible research, which benefits you long-term as a researcher, but also will greatly help anyone that is trying to reproduce or build onto your results down the road.

Hopefully, you won't have much of a learning period to go through and you will reap the benefits of a nicely formatted thesis. The use of LaTeX in combination with *Markdown* is more consistent than the output of a word processor, much less prone to corruption or crashing, and the resulting file is smaller than a Word file. While you may have never had problems using Word in the past, your thesis is likely going to be about twice as large and complex as anything you've written before, taxing Word's capabilities. After working with *Markdown* and **R** together for a few weeks, we are confident this will be your reporting style of choice going forward.

Why use it?

R Markdown creates a simple and straightforward way to interface with the beauty of LaTeX. Packages have been written in **R** to work directly with LaTeX to produce nicely formatting tables and paragraphs. In addition to creating a user friendly interface to LaTeX, *R Markdown* also allows you to read in your data, to analyze it and to visualize it using **R** functions, and also to provide the documentation and commentary on the results of your project. Further, it allows for **R** results to be passed inline to the commentary of your results. You'll see more on this later.

Who should use it?

Anyone who needs to use data analysis, math, tables, a lot of figures, complex cross-references, or who just cares about the final appearance of their document should use *R Markdown*. Of particular use should be anyone in the sciences, but the user-friendly nature of *Markdown* and its ability to keep track of and easily include figures, automatically generate a table of contents, index, references, table of figures, etc. should make it of great benefit to nearly anyone writing a thesis project.

Chapter 1

Introduction to Trees, Random Forests, and the Bootstrap

This thesis is about understanding the variance in variable importance measures of random forests. Random Forests are a statistical learning algorithm developed by Leo Breiman and his collaborators in the early 2000's that leverages bagging and CART (Classification and Regression Trees) methodology to produce quite good predictions of an underlying classification or regression surface. The predictive capability of random forests is quite impressive and much random forest literature centers on exploring why random forests are effective. For this thesis we are interested in the use of random forests for statistical inference. More specifically we are interested in the use of random forest variable importance measures for statistical inference.

Machine learning approaches to regression can often produce models with good predictive accuracy compared to parametric modelling approaches. However, these machine learning algorithms often lack in the interpretability and inferential capabilities of more traditional statistical modelling. While predictive questions are indeed important in science and other applications, many scientific and statistical questions are concerned with description and inference. These questions could be about a causal mechanism, which hypothesis is correct concerning some phenomenon, and which are the important factors. The mechanisms that machine learning techniques use to form predictions on the data are often complicated and not easily amenable to mathematical analysis, but we believe one of Breiman's points in his 2001 paper "Statistical Modelling: Two Cultures" is that statisticians ought to work to develop inferential tools for machine learning techniques. Predictive accuracy is a good metric to measure how a model fits to the data, and without a method that can produce predictively accurate estimates of the underlying data mechanism, inferential conclusions seem less justifiable. Throughout this thesis we will be interested in both the regression and classification settings as random forests can handle response variables of both types. When the need arises to delineate between the setting we are working in, we will be explicit. Furthermore, we will be assuming that for the data at hand, there exists an underlying regression function $Y = f(X) + \varepsilon$ where $f(X)$ is the function and ε is generally assumed to be Gaussian error, but may have some other error structure.

1.1 CART

We begin our discussion of CART (Classification and Regression Trees) trees by considering the following problem. Suppose we have data generated from the piecewise function $Y = f(X) + \varepsilon$ where [at this point insert a very non-linear piecewise function.] with Gaussian error ε. One approach to fitting a model to this data would be to fit a linear regression model. However, the particular form of the data is not well suited for linear regression due to the non-linearity of the data. Another approach we might try to take is as follows. We might try to find a method that can fit a regression line to each piecewise componenet of the data. One such method is to fit CART trees. We fit a CART tree to the data and perform cost-complexity pruning to output the model shown in figure [figure not yet made]. In this example, we produced a regression surface which was able to take into account the piecewise, non-linear nature of the data and produce a reasonable estimate of the underlying data generating mechanism.

The CART methodology allows us to fit a model that can take into account non-linear regression or classification surfaces. The basic idea of CART is that if we can split the predictors into roughly homogenous partitions, or into partitions which minimize predictive error, then we can fit a simple model to predict the response of each partition. CART trees were first introduced by Breiman et al. in 1984 and are a flexible method, capable of handling classification and regression settings. CART is a product of computational statistics, it's development influenced by a data-centric algorithmic approach distinct from the models of classical statistics influenced by the contingency of small samples.

More formally, suppose we have a training data set $Z = \{Z_1, \ldots, Z_n\}$, where $Z_i = (X_i, Y_i)$ is an $p + 1$ dimensional vector in $\mathbb{R}^{p+1}$. Here we have X_i is a d-dimensional predictor variable and Y_i is the response. In particular, we can consider Z to be a $n \times (p + 1)$ array where the rows are observations and the columns are the response and predictor variables. CART works by partitioning the data through binary recursive splits via optimizing some loss function. CART trees are called trees because partitioning the data through binary recursive splits forms a tree-like structure. We adopt notation evocative of this tree structure. Any subset of the training data Z is called a node while the entire data set Z is called the root node. Nodes are of two kinds: they are either terminal (sometimes called leaves) or not terminal. Non-terminal nodes are nodes that are split on in the tree growing process, while terminal nodes are nodes which are not split on. Generally, a node is a terminal node if some stopping rule is reached. Note that terminal nodes form a partition of the training data Z.

We aim to grow a tree with roughly homogenous terminal nodes. As Breiman et al. (1984) note, there are several factors that we need to consider:

1. How to select splits of the data.

2. When to stop splitting the data.

3. How to assign classes or values to terminal nodes.

CART constructs trees by decreasing the nodal impurity of a node t. Nodal impurity is defined through some nodal impurity measure $i(s, t)$, usually the GINI index in the classification setting and the residual sum of squares in the regression setting. We present the regression setting and then the classification setting. If we have a continuous response Y, then we could try to predict Y by partitioning the data using the decision tree structure and predicting that points that fall within a particular partition will take on the average response on that partition. To choose the best binary split on the data, we need to search across the splitting variables and splitting points for the split which maximizes the reduction in the RSS between the parent node and daughter nodes. In particular, we want to find the split which maximizes $RSS_l(j, s) + RSS_r(j, s)$ where j is the proposed splitting variable and s is the proposed splitting point, and RSS_l is the RSS of the proposed left node and RSS_r is the RSS of the proposed right node. The assignment of response within a node is given by the average of the response. We present the following algorithm. The construction of

Algorithm 1 Construction of Regression tree

```
1: while minimum node size not reached do
2:     for each node $t$ do
3:         for $j = 1, \dots, p$ and $s = 1, \dots, n$ do
4:             Compute $RSS_l(j, s) + RSS_r(j, s)$.
5:         Pick the $(j, s)$ which maximizes $RSS_l(j, s) + RSS_r(j, s)$.
6:         Split the current node into left and right nodes according to $(j, s)$.
7:         Compute left and right averages, ave$(y_i|x_i \in t_l)$ and ave$(y_i|x_i \in t_r)$, respectively.
8:     Output the tree $T$.
```

classification trees is similar to the construction of regression trees except a different impurity measure must be used. For the node t containing N_t observations, let

$$\hat{p}_{tk} = \frac{1}{N_t} \sum_{x_i \in t} \mathbb{I}(y_i = k)$$

where $k = 0$ or $k = 1$. Then $\hat{p}_{tk}$ measures the proportion of observations of class k in node t. Using the two-class example, there are several impurity measures available in the classification setting. The most common measure is perhaps the Gini index defined by

$$2p(1 - p).$$

Other options include misclassification error

$$1 - \max(p, 1 - p),$$

and cross-entropy

$$-p \log(p) - (1 - p) \log(1 - p).$$

If t_L and t_R are left and right nodes proposed under the split, let $\hat{p}_{tL}$ and $\hat{p}_{tR}$ be proportion of observations falling into t_L and t_R, respectively. Denote the Gini index of the left node t_L by G_L and denote the Gini index of the right node t_R by G_R. Then our splitting criterion is to seek the spliting variable and splitting point which minimizes

$$\hat{p}_{tL} G_L + \hat{p}_{tR} G_R.$$

The case for misclassification rate and cross-entropy as the impurity measures is similar.

Some issues with CART trees include overfitting and variability. While there is a stopping criterion for growing the CART trees, often times naively growing a tree can result in overfitting to the data. One method of alleviating overfitting is to employ cost-complexity pruning, which searches for an optimal tree that balances fitting a good tree model with overfitting to the data. We do not go into details here, but the point the reader to Friedman et al. (2009). A larger issue with trees is their variability to small perturbations in the data. [Provide computational example.] Allowing the data to vary even a bit can result in a different tree structure upon refitting. As a statistical learning method, this means that CART trees are not robust and suffer from high variance, even when constructed using cost-complexity pruning. To get around this issue of variability the best solution is to employ bagging and use the random forest algorithm.

1.2 Bootstrap

We feel it is worth introducing some notation and theory for the bootstrap. In this line, for the most part we follow notation from Efron and Tibshirani (1993). As is well-known, the bootstrap is a resampling method first introduced by Efron in 1979 with links to earlier work of Tukey and Quenouille on the jackknife. It goes something like this. If we have a dataset D of size n, then it is generally infeasible to obtain more data from the underlying data generating mechanism. Assuming that the data set is not too heavily distributed in the tails and that the sample we have is representative of the population, we can independently resample n points from D to form the bootstrap resampled dataset D^*. We repeat this procedure B many times with B generally quite large. We can then proceed to use the many D^* we generated to obtain estimates, standard errors, and confidence intervals of whichever parameter is of interest. [Note: the bootstrap discussed here is called the non-parametric bootstrap. There is also the parametric bootstrap, but we do not go into that method here].

What is going on? If we obtain an i.i.d. random sample of size n from a probability distribution F, then we can form the empirical distribution function $\hat{F}$ which places a probability of equal mass $\frac{1}{n}$ of seeing any value of x_i for $i = 1, \ldots, n$. We do not want to go too deeply into bootstrap theory in this introduction, but we would like to mention the plug-in principle underlying applications of the bootstrap, as well as variance estimation.

In statistics, we are interested in obtaining information about parameters defined on a probability distribution F by dealing with statistics defined on the EDF $\hat{F}$. Often a parameter is a function of the probability distribution:

$$\theta = t(F)$$

where in this context t denotes the functional form of the parameter. Often the probability distribution F is unknown or too complex to deal with directly, so we try to form estimates. One estimate we could form of the parameter $\theta = t(F)$ is the plug-in estimate (or plug-in statistic, or functional statistic), which is defined to be

$$\hat{\theta} = t(\hat{F}).$$

It can be shown under suitable conditions, that the functional statistic $t(\hat{F})$ is the nonparametric maximum likelihood estimate of $t(F)$ (Efron and Tibshirani). Therefore there is good theoretical ground for proceeding with estimating parameters using the bootstrap.

The usefulness of the bootstrap extends beyond estimating functional statistics. Using the bootstrap, we can estimate the variance of the functional statistic using several methods. The most ubiquitous is perhaps the bootstrap estimate of the standard error. Say we have drawn B many bootstrap samples of a dataset Z to obtain Z_b^* for $b = 1, \dots, B$. We compute an estimate $\hat{\theta} = s(Z)$ of $\theta = t(F)$, where Efron and Tibshirani emphasize that $s(Z)$ is not necessarily the plug-in estimate $t(\hat{F})$. To emphasize that the estimates computed from Z_b^* are bootstrap replicates of $\hat{\theta}$, we write

$$\hat{\theta}_b^* = s(Z_b^*).$$

In some settings we write $\hat{\theta}^* = s(Z^*)$ when we do not care as much about the actual subscripting from the bootstrap and just want to emphasize the fact we are estimating $\hat{\theta}$ using bootstrap replicate.

Denote the standard error of the statistic $\hat{\theta}$ by $se_F(\hat{\theta})$. Following Efron and Tibshirani, the bootstrap estimate of the $se_F(\hat{\theta})$ is a plug-in estimate of the standard error of $\hat{\theta}$ replacing the distribution F in the subscript by the corresponding empirical distribution $\hat{F}$ and is given by $se_{\hat{F}}(\hat{\theta}^*)$. Efron and Tibshirani call $se_{\hat{F}}(\hat{\theta})$ the ideal bootstrap estimate of the standard error of $\hat{\theta}$. Beyond simple examples like the mean, the exact form of $se_{\hat{F}}$ is often too difficult to compute exactly, but the bootstrap can be used to compute a good estimate of $se_{\hat{F}}$. Namely, once we have drawn $Z_1^*, \dots, Z_B^*$ independent bootstrap samples from Z and compute the B many bootstrap replicates $\hat{\theta}_b^* = s(Z_b^*)$, then the estimate of the standard error $se_F(\hat{\theta})$ is given by

$$\widehat{se}_B = \left(\frac{1}{B-1} \sum_{b=1}^{B} (\hat{\theta}_b^* - \bar{\theta}^*) \right)^{1/2},$$

where $\bar{\theta}^* = \frac{1}{B}\sum_{b=1}^{B} \hat{\theta}_b^*$ is the sample mean of the bootstrap replicates. Note $\widehat{se}_B$ is the familiar formula for the standard deviation of a sample.

There are several other estimates of the standard error commonly used when exact forms of the standard error are unavailable or infeasible. The first of these is the jackknife. As a computational tool, the jackknife takes a different approach from the bootstrap. Say for the parameter θ, we have an estimate $\hat{\theta} = s(Z)$. Define the i-th "leave one out" sample $Z_{(i)} = (Z_1, \ldots, Z_{i-1}, Z_{i+1}, \ldots, Z_n)$. Corresponding to $Z_{(i)}$ we can form $\hat{\theta}_{(i)} = s(Z_{(i)})$, the ith jackknife replicate of the estimator $\hat{\theta}$. Define

$$\hat{\theta}_{(\cdot)} = \frac{1}{n}\sum_{i=1}^{n}\hat{\theta}_{(i)}$$

which is akin to the mean of the jackknife replications. From here we can obtain the jackknife estimate of the standard error, which is defined by

$$\widehat{se}_{\text{jack}} = \left(\frac{n-1}{n}\sum_{i=1}^{n}(\hat{\theta}_{(i)} - \hat{\theta}_{(\cdot)})^2\right)^{1/2}.$$

The idea of the jackknife is that absent new realizations of the data, a simple procedure we can undertake is to see the effect on the accuracy of the estimator when we remove each observation sequentially. In particular, we could consider the jackknife as forming n many datasets of size $n-1$ with the ith observation removed. The jackknife may be easier to compute than the bootstrap, but as the jackknife only uses limited information about the n observations and does not simulate additional full datasets like the bootstrap, overall it is less efficient than the bootstrap. The jackknife is related to the bootstrap in the sense that the jackknife is a linear approximation to the bootstrap. As Efron and Tibshirani explain, for linear statistics the jackknife is as efficient as the bootstrap. However, for highly non-linear statistics (of which it turns out random forests are), it turns out that jackknife estimation is very inefficient compared to the bootstrap. There are other methods of estimating the standard error such as the Infinitesimal Jackknife and Non-parametric delta method, but we will address those methods in later chapters.

The bootstrap is a flexible method and applications of the bootstrap extend beyond simply sampling pairs of observations from the the data $Z = \{Z_1 = (X_1, Y_1), Z_2 = (X_2, Y_2), \ldots, Z_n = (X_n, Y_n)\}$. In particular, there are model-based approaches to the bootstrap that are often useful. A particularly important example for this thesis is bootstrapping the least squares linear regression model. In least squares, we model the response Y observed at the vector X as being given by $\mathbb{E}(Y|X) = X^T\beta + \varepsilon$, where $X = (1, X_1, \ldots, X_p)^T$ and $\beta = (\beta_0, \beta_1, \ldots, \beta_p)$ and ε is an error term of length n. In practice, the vector of coefficients β is unknown, so we form an estimate $\hat{\beta}$ through applying least squares regression. After applying least squares. For a point X_j, the least squares estimate of Y_j is given by $\hat{Y}_j = X_j\hat{\beta}$. The raw residuals are defined by $e_j = Y_j - \hat{Y}_j$ with the estimate of the variance σ^2 is given by

$$s^2 = \frac{1}{n-2}\sum_{j=1}^{n}e_j^2$$

.

There are a couple approaches we could consider in applying the bootstrap to linear regression. The conceptually simpler method is to resample cases (X_i, Y_i) to generate bootstrap replications of Z, $Z^* = \{(X_1^*, Y_1^*), \ldots, (X_n^*, Y_n^*)\}$ where the asterisks denote

Algorithm 2 Linear Regression Cases Resampling

```
1: for $b = 1, \dots, B$ do
2:     Draw a bootstrap sample $Z_b^*$ of size $n$ from the cases $(X_i, Y_i)$ to obtain $Z_b^* = \{(X_1^*, Y_1^*), \dots, (X_n^*, Y_n^*)\}$.
3:     Fit least squares regression to $(X_1^*, Y_1^*), \dots, (X_n^*, Y_n^*)$ to obtain the estimates $\hat{\beta}_b^*$ and $\hat{s}_b^{*2}$.
```

that we have resampled from the original data. With B many such datasets Z^*, we can use the following algorithm adapted from Davison and Hinkley (1997). Another approach we could take is to resample the residuals of the linear model. To motivate this approach, note that the vector of raw residuals $(e_1, \dots, e_n)$ approximate the error term ε. That is, assuming the bivariate distribution of (X, Y) is such that the specification of the linear model is correct, then given the true value of β, the error terms would be given by $\varepsilon_i = Y_i - X_i\beta$. As β is generally unknown, the best we can do is to form estimates of the error terms $\hat{\varepsilon}_i = Y_i - \hat{Y}_i = Y_i - X_i\hat{\beta}$. We now have a vector of estimated error terms $\hat{\varepsilon} = (\hat{\varepsilon}_1, \dots, \hat{\varepsilon}_n)$ which we could consider as being an i.i.d. sample if the cases (X_i, Y_i) were independently sampled from the distribution of (X, Y). As Efron and Tibshirani note, we could consider obtaining any particular component of the vector $\hat{\varepsilon}$ as being from the EDF $\hat{F}$ of the true distribution of the error terms F, where we place a probability mass of $\frac{1}{n}$ to drawing any one of the $\hat{\varepsilon}_i$. Taking this conceptual approach, we could then draw bootstrap samples of the residuals $\hat{\varepsilon}_1, \dots, \hat{\varepsilon}_n$. Consider the X_i component of the cases (X_i, Y_i) fixed and the estimate $\hat{\beta}$ as fixed. Resample with replacement the vector $\hat{\varepsilon}$ to obtain the bootstrapped vector of residuals, $\hat{\varepsilon}^* = (\hat{\varepsilon}_1^*, \dots, \hat{\varepsilon}_n^*)$. Then for each $i = 1, \dots, n$ our bootstrapped response variable would be $Y_i^* = X_i\hat{\beta} + \hat{\varepsilon}_i^*$. Denoting $X_i^* = X_i$, we obtain the bootstrapped dataset $Z^* = \{(X_1^*, Y_1*), \dots, (X_n^*, Y_n^*)\}$. We then run a linear regression again on Z^* to obtain estimates $\hat{\beta}^*$ and $\hat{s}^{*2}$. We present what was just discussed in the following algorithm. Note that the two methods of resampling the linear regression model

Algorithm 3 Bootstrapping Residuals of Linear Regression Model

```
1: Fit a linear regression model to the data $Z = \{(X_1, Y_1), \dots, (X_n, Y_n)\}$ to obtain the estimate $\hat{\beta}$.
2: Compute the estimated residuals vector $\hat{\varepsilon} = (\hat{\varepsilon}_1, \dots, \hat{\varepsilon}_n)$, where $\hat{\varepsilon}_i = Y_i - X_i\hat{\beta}$.
3: for $b = 1, \dots, B$ do
4:     Sample with replacement $n$ times from $\hat{\varepsilon}$ to obtain the bootstrapped residuals $\hat{\varepsilon}^{*b} = (\hat{\varepsilon}_1^{*b}, \dots, \hat{\varepsilon}_n^{*b})$.
5:     for $i = 1, \dots, n$ do
6:         Set $X_i^{*b} = X_i$ and $Y_i^{*b} = X_i\hat{\beta} + \hat{\varepsilon}_i^{*b}$
7:     Obtain the bootstrapped dataset $Z^{*b} = \{(X_1^{*b}, Y_1^{*b}), \dots, (X_n^{*b}, Y_n^{*b})\}$.
8:     Fit a linear regression model on the bootstrap replicate of the data $Z^{*b}$ to obtain the estimates $\hat{\beta}^{*b}$ and $(\hat{s}^2)^{*b}$.
```

rely on different assumptions on the data. In particular, we are assuming that the error structure of the data does not rely on our predictors X_i. By fixing the X_i, we are assuming the distribution of the error terms F is invariant to the data we have observed. This is a fairly strong assumption. Furthermore, resampling the residuals works well only if the variance of the errors is homoskedastic. If the variance of the errors is heteroskedastic, then the modeling assumptions we made are invalid and a weighted linear regression approach or wild bootstrap approach may be necessary. The cases resampling is much less sensitive to the modeling assumptions we make as we do not generate the Y_i^*'s in the same way as with the residuals method. If either of these two methods are applicable, we could then proceed to form summary statistics such as the sample mean or sample standard deviation of the coefficients and the estimate of the variance of the model. The two methods we presented for bootstrapping the linear regression are easily adapted to other types of models. In particular, we could consider applying these methods to highly non-linear estimators, which is precisely what we do with bagged and random forests with respect to CART trees.

1.3 Random Forests

Among the limitations of CART discussed in the preceding sections, the biggest issues is perhaps the variability of the method and the issue of collinearity among predictors. While overfitting can be addressed using cost-complexity pruning, variability and collinearity are not fixed by pruning. Random forests deal with these two issues of CART by introducing a resampling and randomization mechanism. The natural order is to first discuss bagged forests before turning to random forests. One method of

Algorithm 4 Bagged Forest algorithm

```
1: for $b = 1, \dots, B$ do
2:     Draw a bootstrap sample $Z_b^*$ of size $N$ from the training data $Z$.
3:     Grow a CART tree $T_b$ on each bootstrap sample $Z_b^*$.
4: Output the bagged forest ensemble $\{T_b\}_{b=1}^{B}$.
```

improving CART trees is to bag them. Bagging, which stand for bootstrap aggregating, is a variation reduction technique particularly useful for improving the predictive power of weak learners. We are interested in bagging CART trees to reduce the variability of single trees under slight perturbations of the data. Generally bagged ensembles of learners produce robust predictions in comparison to running the learner once.

1.3.1 How Bagged Forests Work

In particular, bagged forests are an ensemble obtained by taking many bootstrap samples of the data and fitting a tree to each bootstrapped dataset. In this case, we grow the trees quite deep and do not employ cost-complexity pruning. The idea behind this decision is that we want to sufficiently explore the feature space using the tree ensemble, and since we are also taking an average of the trees, we are fine with overfitting at least a little bit. As the algorithm above for bagged forests indicates, the output is an ensemble of trees $\{T_b\}_{b=1}^B$. Given a test data point, we form a prediction by taking the average of predictions given by the tree ensemble:

$$\hat{f}_{bf}^B(x) = \frac{1}{B}\sum_{b=1}^{B} T_b(x),$$

where $\hat{f}_{bf}^B$ indicates we are taking the bagged forest estimate of the underlying regression or classification function using B many trees. Note that it is important to be consistent in the choice of splitting criterion throughout the bagging process.

It has been shown by Friedman and Hall (2000) and Chen and Hall (2003) that bagging is especially effective when used on highly non-linear models such as CART trees. Under ideal conditions, bagged estimates of non-linear estimators reduces both the bias and variance of the estimator. So bagging trees seem to be effective because of the reduction to the variance of the estimator, which produces a more robust prediction.

When we bag trees each observation in the data is not used within each individual tree. A bootstrap replicate Z_b of the data Z will likely exclude some of the observations within the data. The observations used within the bootstrap replicate Z_b is called the in-bag data while the observations not used within the bootstrap replicate Z_b is called the out-of-bag (OOB) data and is denoted by $\bar{Z}_b$. The OOB data allows us to approximate the test error of the ensemble as follows. For simplicity suppose we are in the regression setting (the classification setting is similar). Then the OOB estimate of the MSE of the bagged forest is given by

$$MSE_{\text{OOB}}(T; Z) = \frac{1}{B}\sum_{b=1}^{B} MSE(T_b; \bar{Z}_b).$$

As the number of trees in the ensemble grows, the OOB estimate of the MSE for the bagged forest converges to the LOOCV estimate of the MSE for forest (Friedman et al.).

A weakness of bagged forests is collinearity between trees (Friedman et al.). This collinearity between trees grown from the bootstrap sample can be addressed by adding a randomization mechanism in the tree growing process. When we grow a tree from a bootstrap sample, even with the randomness induced by resampling from the data, certain features may be explored at the expense of other just as interesting features. This is a particular issue with collinear predictors. If two predictors are collinear, then the bagged forest might consistently choose one predictor over another even if the predictor not chosen leads to splits that are just as informative. This is due in part to the greedy nature of the CART algorithm when searching for optimal splits over the feature space (Cite ESL or some similar textbook for this point). The algorithm does not take into account the second best or third best splits. Furthermore, it is not difficult to see that bagging will generally produce an ensemble of trees that are quite similar to one another, subject to some perturbations. These trees will be strongly correlated with other trees in the ensemble, so if there is a less explored part of the feature space, then the ensemble will struggle to produce good predictions over that part of the feature space.

1.3.2 The Random Forest Algorithm

Random Forests try to deal with this issue of correlated trees and collinearity among predictors by choosing at random only $m \leq p$ of the predictors to be considered as candidate splitting variables at each split in each tree in the ensemble. This randomness further reduces the variance of the bagged forest by decorrelating the trees in the ensemble. Furthermore, while individually the trees may perform worse than a single pruned tree, collectively the ensemble has a better chance of exploring the feature space fully. To form a prediction at a test point x, we have in the regression

Algorithm 5 Random Forest algorithm

```
1: for $b = 1, \dots, B$ do
2:     Draw a bootstrap sample $Z_b^*$ of size $N$ from the training data $Z$.
3:     Grow the CART tree $T_b$ on $Z_b^*$ with the following modification:
4:     while minimum node size $n_{\min}$ not reached across $T_b$ do
5:         Select $m \leq p$ candidate splitting variables at random.
6:         Pick the best splitting variable and splitting point among the $m$ variables
   selected at random.
7:         Split the node into two daughter nodes.
8: Output the random forest ensemble $\{T_b\}_{b=1}^B$.
```

setting

$$\hat{f}_{rf}^B(x) = \frac{1}{B} \sum_{b=1}^{B} T_b(x).$$

[Add in the classification setting].

1.3.3 Variable Importance Measures

One of the advertised outputs of random forests is the variable importance (VI) measure. There are two main variable importance measures in common use, with the choice of VI measure varying depending on the setting and splitting criterion chosen. The first choice is the Mean Decrease in Impurity (MDI) which is typically used in the classification setting where the GINI index or Shannon entropy is used as the splitting criterion. The second choice is Mean Decrease in Accuracy (MDA) which is typically used in the regression setting where RSS has been used as the splitting criterion. The idea of MDI is to find how much the nodal impurity $p(t)\Delta i(s,t)$ decreases for all nodes t in which the variable of interest X_j is used and to take that average over all trees in the ensemble. More important variables are those which are on average more often chose for splits and which also contribute most to reducing the nodal impurity of the trees. The idea of MDA is to measure for each variable X_j, on average how much the predictive accuracy of the forest as measured using RSS suffers when the X_j component is permuted across observations within the OOB dataset $\bar{Z}_b$ of each tree T_b in the ensemble. More important variables in the random forests are those for which the variable importance is large, as those are the variables for which the predictive

Algorithm 6 MDI Variable Importance

```
1: Grow a random forest $\{T_b\}_{b=1}^B$.
2: for $j = 1, \ldots, p$ do
3:     for $b = 1, \ldots, B$ do
4:         Compute the importance of $X_j$ in $T_b$ as $VI_b(X_j) = \sum_{t \in T_b} \mathbb{I}(j_t = j) p(t) \Delta i(s,t)$ to be the sum of the decrease in impurity over nodes where variable $X_j$ is used .
5:     Compute the importance of $X_j$ in the random forest to be $VI(X_j) = \frac{1}{B} \sum_{b=1}^B VI_b(X_j)$.
```

Algorithm 7 MDA Variable Importance

```
1: Grow a random forest $\{T_b\}_{b=1}^B$.
2: for $j = 1, \ldots, p$ do
3:     for $b = 1, \ldots, B$ do
4:         Permute the $X_j$ component of $Z_b$ to obtain the dataset $Z_b^j$, where $X_j$ has been permuted.
5:         Compute the importance of $X_j$ in $T_b$ to be $VI_b(X_j) = \frac{1}{|Z_b|}(RSS(T_b, Z_b) - RSS(T_b, Z_b^j))$.
6:     Compute the importance of $X_j$ in the random forest to be $VI(X_j) = \frac{1}{B} \sum_{b=1}^B VI_b(X_j)$.
```

accuracy of the random forest suffers the most. Note that as variable importance measures currently defined and used, the threshold for importance of a variable is something which the researcher has to decide.

1.3.4 Issues with Random Forests

While random forests are good out of the box predictors, there are situations where the random forest algorithm can fail to produce good predictions. If the underlying regression function the random forest is linear, if the predictors are highly correlated, or if the data cannot be bootstrapped, then the random forest will not perform well. Random forests are especially adept at handling highly non-linear functions, but can struggle with linear response compared to say linear regression.

If the predictors are highly correlated, then ithas been shown by Strobl et al. that the trees within the forest ensemble will be biased and the variable importance measures will not be reliable due to confounding between similar looking variables. While the randomization step in random forest algorithm can alleviate the correlation between trees in the ensemble, the collinearity between predictors can cause the predictive performance of the random forest to suffer. This is an issue with the underlying CART algorithm and Hothorn, Strobl, and their collaborators developed Conditional Inference trees and forests to deal with the issue of collinearity in the data (which will be explained in the next chapter). While using random forest variable importance for variable selection certainly seems like a good idea, the issue of collinear data makes the applicability of variable importance measures less immediate. Perhaps a proper use of random forest variable importance measures as a variable selection procedure would be to run a random forest only after some dimension reduction techniques have been applied. If the collinearity of the predictors has been dealt with, then heuristically the variable importance measures should be able to more accurately estimate the importance of predictors within the ensemble.

Finally, there are situations where the data cannot be bootstrapped. This could be due to a number of factors including that the data has a heavily-tailed distribution or if there are particularly extreme values. In this case, a different bootstrap scheme could perhaps remedy the situation where the naive bootstrap fails. One common important resampling scheme used within random forests is to use the m-out-of-n bootstrap (also referred to subsampling or subagging in the literature). The m-out-of-n bootstrap is a resampling scheme which resamples with or without replacement $m \leq n$ observations from the data to form datasets with m many data points to run the bootstrap computation. Bickel et al. have shown that in important cases, subsampling can succeed where the bootstrap fails. Of course, using less of the available data is less efficient, but in the context of random forests this loss of efficiency may not be an issue. In [Cite paper comparing CI trees to CART trees using IJ], simulation results showed that subsampling instead of using the standard nonparametric bootstrap can improve the performance of random forests. The authors suggested that subsampling further reduces the variance of the ensemble by producing trees that are even more decorrelated. Their heuristic is that there is a lower probability of duplicate data points being chosen using subsampling (this probability is zero if we are subsampling without replacement), furthermore there is a lower probability of highly correlated data points being chosen as one of the $m \leq n$ points. This certainly seems plausible, but we would like to see further simulation results or a technical result that explains why subsampling works well for forests. We would also like to note that most consistency results (Biau et al., Scornet et al., Ishwaran et al., and Wager et al.) make this subsampling assumption in their analysis of the random forest model, as subsampling constructions make the forest ensemble more amenable to mathematical analysis.

1.4 Focus of this Thesis

In the previous three sections we discussed the basics of trees, the bootstrap, and random forests. We would like to now discuss the direction we are going to take this thesis. We are interested in developing inferential tools using the random forest algorithm. The random forest often produces good predictions out of the box with little tuning required except growing a sufficient number of trees and choosing a good value of $m \leq p$ to try at each split in the tree growing process. The ease of fitting random forests is one of the advantages of the random forest compared to more complex methods like neural networks or support vector machines. Even if the random forest is easier to fit than neural nets or SVMs, the underlying random forest mechanism is quite complex. This complexity makes developing inferential tools with random forests difficult. In the next chapter we will discuss some approaches that have been developed recently by researchers. Of particular interest for us are the variable importance measures for random forests. There is the MDI variable importance and the MDA variable importance measure. While there are concerns about the bias in the MDI and MDA VI measures with random forests using CART tree's due to how CART tree's are constructed (Strobl et al., 2008), we would like to extend the inferential capabilities of random forests using CART as this is perhaps the most popular version of the algorithm in use.

The random forest VI can be viewed as a random variable with an unknown probability distribution. While determining the distribution of random forest VI's would be ideal towards developing inference for random forests, finding the distribution of random forest VI's is difficult and unclear in general. However, note that the bootstrap provides a computational method of approximating the sampling distribution of random forest VI measures. We could then proceed to obtaining estimates of the standard error and to form approximate interval estimates. We could grow a random forest once, obtain an estimate of the VI of each variable X_j and then run a bootstrap to obtain estimates of the standard error of each variable. Inference could then proceed using the VI measures along with the estimate of standard error. Note that in this scheme there is two levels of bootstrap at play. There is the bootstrap within each random forest and then the bootstrapping to obtain the standard error estimate. The estimate we produce of the VI of a variable X_j can be erratic. Running the random forest again can produce a new estimate of the VI of variable X_j. As Efron suggests in his 2014 article "Estimation and Accuracy after Model Selection," we could remedy the erratic, non-smooth nature of the VI estimate by bagging $VI(X_j)$. However, when we bag $VI(X_j)$ we would require a third-level of bootstrapping to produce an estimate of the standard error of the bagged estimate of $VI(X_j)$. This is computationally expensive, especially considering that the computational cost of the random forest depends on the number of observations, number of tree's grown, and number of variables present. However, applying Efron's infinitesimal jackknife (IJ) technique, which we will discuss in the next chapter, we can efficiently estimate the standard error of the bagged estimate of $VI(X_j)$ only using the two levels of bootstrapping. In this thesis we will be exploring the properties of IJ estimates of the standard error of the bagged estimate $VI(X_j)$ utilizing different bootstrap schemes. Utilizing the IJ, we can then produce interval estimates and confidence intervals of the variable importance of variables in the random forest. This allows us to characterize the uncertainty of $VI(X_j)$ when using variable importance measures for description, variable selection, and inference.

1.5 Outline of Remaining Chapters

Chapter 2: Inference and variable importance for random forests. In this chapter we will discuss the various approaches to inference for random forests that have been developed by researchers. These include Ishwaran and Louppe's analysis of VI measures of RF. Ishwaran's analysis is of the variable importance scheme he devised.

Ishwaran's scheme is amenable to theoretical analysis and he produces asymptotic results for his scheme. Louppe analyzes the MDI variable importance in the context of a construction called totally randomized forests in the setting of finite sample spaces. Louppe investigates the effect of relevant and irrelevant variables to MDI VI in this setting and provides asymptotic results. These are two of themain papers investigating properties of VI.

The other schemes include Mentch and Hooker's prediction intervals which are produced by interpreting the random forest ensemble as a particular type of U-statistic and applying the U-statistics theory to produce prediction intervals. They then can produce hypothesis tests and confidence intervals in this context.

A second approach towards prediction intervals is Wager's application of the IJ to the random forest estimator to produce prediction intervals and confidence intervals. Wager has a couple of papers in this area which we will discuss. In this part we will also go into greater detail about the IJ, it's properties, and why it works.

We will also discuss the conditional inference trees and conditional variable importance developed by Strobl, Hothorn, and their colleagues to deal with the biased manner in which the trees in random forests are constructed. In this part we will also discuss the Infforest variable importance measure developed by Aurora.

Chapter 3: In this chapter we will go into great detail about the bootstrap and why it works. We will discuss the technical details of different methods of estimating the standard error of functional statistics, and we will go into the construction of different bootstrap confidence intervals along with properties of these confidence intervals. We will also discuss different bootstrap schemes and in particular subsampling, and why subsampling works well, especially in the random forest context.

Chapter 4: We will present our different approaches to bootstrapping variable importance measures and provide simulation results. The first approach is the cases bootstrap, the second approach is the bootstrapping residuals of the random forest approach. Within the cases approach, there are two ways we will try to constrain the variability of bootstrap replicates. Adopting the notation from general linear models, we could fit a pruned tree predicting $Y \sim X_1, \ldots, X_p$. Then we would bootstrap from the terminal nodes of this single tree to try to produce bootstrap replicates which capture the relationship between the variables. Another approach is for each predictor X_j, fit a single well pruned tree $X_j \sim X_1, \ldots, X_{j-1}, X_{j+1}, \ldots, X_p$. We would then draw bootstrap samples from the terminal nodes of this tree which respect the partition produced by these terminal nodes. The idea is to fix the relationship between every predictor not X_j and allow the X_j's to be exchanged across the bootstrap samples. We would produce simulation results of each approach and compare the statistical properties of these different constructions.

Chapter 5: If all goes well there might be a chapter 5 where we explore large sample properties and distribution of VI measures. This is likely going to be a difficult mathematical problem to solve in general, so a tangible course of action might be to pick a generating mechanism and easy VI measure from which we could figure out the VI distribution under some set of strong assumptions.

Chapter 2

Variable Importance and Inference for Random Forests

2.1 Introduction

In this chapter we focus on various approaches towards inference for random forests that have been developed. Generally, we could consider to be two approaches towards inference with random forests.

The first approach involves using variable importance measures of random forests to evaluate the relative importance of different variables in the construction of the forest. Ishwaran, Louppe, Owens, and Strobl et al. have been involved in work in this area. Louppe and Ishwaran focused on theoretical properties of variable importance measures while Owens, and Strobl and colleagues work is centered on developing less biased variable importance measures for hypothesis testing.

The second approach involves utilizing properties of the functional form of the random forest estimator to construct prediction intervals. Once predicton intervals have been constructed, depending on the context, confidence intervals and hypothesis tests can be produced. Mentch and Hooker's work involves interpreting the random forest ensemble as a particular type of U-statistics, and applying theoretical results about U-statistics to construct prediction intervals. Wager's work involves applying the infintisemal jackknife to the random forest estimator to produce prediction intervals. With both approaches there are asymptotic results that we will discuss in some detail. In addition, we will discuss some of the consistency results that have been proven for random forests. It is worth noting here that in mathematically analyzing the random forest algorithm, there are trade-offs between fidelity to the CART algorithm and amenability to mathematical tool. To our best knowledge, at this time there are no theoretical results showing that random forests constructed using CART are consistent. Generally, simplifications need to be made to the random forest algorithm to allow for asymptotic analysis. These simplifications usually involve using a different partitioning scheme than CART and also in working a sampling without replacement framework. We will note when such assumptions are being made when necessary. Otherwise, we assume a setting in which the random forest is constructed using CART and bootstrap sampling with replacement.

2.2 Some Theory for Variable Importance Measures

In this section we discuss Louppe and Ishwaran's work exploring theoretical properties of variable importance measures.

2.2.1 Ishwaran's Variable Importance Measure

Ishwaran's approach to variable importance measures of random forests, as developed in "Variable Importance in Binary Regression Trees and Forests," differs from the original variable importance measures for random forests, so we require some additional vocabulary.

Suppose T is a binary recursive tree and suppose that T has M many terminal nodes. For each point $\mathbf{x}$ in the feature space, T maps $\mathbf{x}$ to one of the M-many terminal nodes. In particular, if we let $\mathcal{X}$ denote the feature space, then T is a function $\mathcal{X} \to \{1, \ldots, M\}$ defined by the equation

$$T(\mathbf{x}) = \sum_{m=1}^{M} m B_m(\mathbf{x}),$$

where $B_m(\mathbf{x})$ is a $0-1$ basis function which partition the feature space $\mathcal{X}$.

Let $Z = \{(\mathbf{x}_i, Y_i) | i = 1, \ldots, n\}$ denote the training data, where $\mathbf{x}_i$ is a covariate in the feature space and Y_i is the response. We call T a binary regression tree if it is a binary recursive tree grown from Z using using binary recursive splits of the form $x_j \leq c$ and $x_j > c$ where split values c are chosen based on the observed $\mathbf{x}_i$ in the training data Z. The value a_m in the terminal node is the average response of the training observations falling in the mth node. That is,

$$a_m = \frac{\sum_{i=1}^{n} \mathbb{I}\{T(\mathbf{x}_i) = m\} Y_i}{\sum_{i=1}^{n} \mathbb{I}\{T(\mathbf{x}_i) = m\}}.$$

Note that $\mathbf{x_i}$ denotes a row of covariates for the training data Z, while x_j denotes the jth variable along the columns of the training data.

For a binary regression tree, the basis functions $B_m(\mathbf{x})$ are product splines of the following form:

$$B_m(\mathbf{x}) = \prod_{l=1}^{L_m} [x_{l(m)} - c_{l,m}]_{s_{l,m}},$$

where L_m denotes the number of splits used to construct $B_m(\mathbf{x})$. For each split l, there is a splitting variable $\mathbf{x}_{l(m)}$ which denotes the $l(m)$th coordinate of $\mathbf{x}$ and a splitting value $c_{l,m}$. The $s_{l,m}$ are binary ± 1 values, where for a given scalar x, $[x]_{+1} = \mathbb{I}(x > 0)$ and $[x]_{-1} = \mathbb{I}(x \leq 0)$. Note that the basis functions satisfy an orthogonality property, which gives $B_m(\mathbf{x})B_{m'}(\mathbf{x}) = 0$ if $m \neq m'$. Note also that given a tree T, the predictor associated with the tree can be written as a linear combination of basis functions:

$$\hat{\mu}(\mathbf{x}) = \sum_{m=1}^{M} a_m B_m(\mathbf{x}).$$

We are now prepared to define Ishwaran's variable importance measure.

Informally, the MDA variable importance of a variable x_j is the difference between MSE of the tree T when x_j is randomly permuted and MSE of the tree T when x_j is not permuted. As such a scheme of variable importance is difficult to analyze, Ishwaran proposes a surrogate measure. For the variable x_j, we drop $\mathbf{x}$ down the tree and follow the binary splits until either a terminal node is reached or a node with a split depending on x_j is reached. If a node with a split depending on x_j is reached, we then subsequently assign $\mathbf{x}$ randomly to either the left or right daughter node, whenever there is a split, until we reach a terminal node. The difference in MSE between noising up x_j and not noising up x_j to be the variable importance of x_j in the tree T. Denote the tree that results from noising up x_j by T_j.

Such a scheme relies on the following heuristic: if we chose an adequete splitting rule to construct our tree, then we expect that variables that are split earlier in the tree are more important, since prediction will suffer the most from noising up a variable higher up in the tree than a variable close to a terminal node. This is a behavior observed in CART trees and random forests based on CART: splits closer to the root node are more influential than splits close to terminal nodes, so the MDA or MDI variable importance of variables split on close to the root node are expected to be higher than otherwise.

2.2.2 Maximal Subtrees and Theoretical Results

Defining a structure on binary regression trees called subtrees, Ishwaran is able to write the predictor for the noised up tree as a deterministic component relying on terminal nodes for no parent nodes involve a split on x_j, and a random component involving terminal nodes for which there are parent nodes involving a split on x_j. The definition of the subtree is quite intuitive. We call $\tilde{T}_j$ a j-subtree of the tree T, if the root node of $\tilde{T}_j$ has daughters that depend on an x_j split. A j-subtree $\tilde{T}_j$ is a maximal j-subtree of the tree T, if there are no larger j-subtrees continaing $\tilde{T}_j$. For a given tree T and for each variable x_j, there is a set of K_j many distinct maximal j-subtrees, which are denoted by $\tilde{T}_{1,j}, \ldots, \tilde{T}_{K_j,j}$. Note each distinct $\tilde{T}_{k,j}$ maximal j-subtree contains a set of distinct terminal nodes $M_{k,j}$. Each $M_{k,j}$ is distinct for $k = 1, \ldots, K_j$, since we are working with maximal j-subtrees. Define

$$M_j = \bigcup_{k=1}^{K_j} M_{k,j}$$

to be the set of terminal nodes for which there is a parent node involving a split on x_j. Ishwaran proves the following lemma about the functional form of the predictor for the tree T_j.

Lemma 2.1. *Let $\hat{\mu}_j(\mathbf{x})$ denote the predictor for T_j. Then*

$$\hat{\mu}_j(\mathbf{x}) = \sum_{m \notin M_j} a_m B_m(\mathbf{x}) + \sum_{k=1}^{K_j} \tilde{a}_{k,j} \mathbb{I}\{T(\mathbf{x}) \in M_{k,j}\},$$

where $\tilde{a}_{k,j}$ is the random terminal value assigned by $\tilde{T}_{k,j}$ under the random left right path through $\tilde{T}_{k,j}$. We write $\tilde{P}_{k,j}$ to denote the distribution of $\tilde{a}_{k,j}$.

For a proof, the reader is referred to Ishwaran's paper. It is a bit surprising that the functional form of $\hat{mu}_j(\mathbf{x})$ can be separated into the two components. Given this lemma and the definition of j-subtrees, we can more formally define the variable importance of x_j in the tree T.

Let g be a loss function. Often we use the squared error to evaluate loss, which corresponds to MSE, but there is no strict requirement in the definition. Denote the test data by $(Y, \mathbf{x})$. Then the prediction error of the predictor $\hat{\mu}$ is given by $\mathbb{E}(g(Y, \hat{\mu}(\mathbf{x})))$. As a reminder, we assume that there is an underlying regression function

$$Y = \mu(\mathbf{x}) + \varepsilon,$$

where ε is independent error with zero mean and variance $\sigma^2 > 0$. Similarly, we can define the prediction error of the predictor $\hat{\mu}_j$ to be $\mathbb{E}(g(Y, \hat{\mu}_j(\mathbf{x})))$. Set $g(Y, \hat{\mu}(\mathbf{x})) = (Y - \hat{\mu}(\mathbf{x}))^2$ to be the L_2 loss, which corresponds to MSE. Define the variable importance of the variable x_j to be

$$\Delta_j = \mathbb{E}((Y - \hat{\mu}_j(\mathbf{x}))^2) - \mathbb{E}((Y - \hat{\mu}(\mathbf{x}))^2).$$

Application of the lemma and some manipulation allows us to write

$$\Delta_j = \mathbb{E}(R_j(\mathbf{x})^2) - 2\mathbb{E}\left(R_j(\mathbf{x})[\mu(\mathbf{x}) - \hat{\mu}(\mathbf{x})]\right),$$

where

$$R_j(\mathbf{x}) = \sum_{k=1}^{K_j} \sum_{m \in M_{k,j}} (\tilde{a}_{k,j} - a_m) B_m(\mathbf{x}).$$

To aid in his analysis, Ishwaran makes the assumption that the true regression function μ is of similar form to T. That is, assume

$$\mu(\mathbf{x}) = \sum_{m=1}^{M} a_{m,0} B_m(\mathbf{x}),$$

where $a_{m,0}$ are the true, but unknown, terminal values. Under this and some other large sample assumptions, Ishwaran finds that asymptotically, each maximal j-subtree will tend to contribute equally to the variable importance Δ_j. In effect, Ishwaran finds that nodes closer to the root of a maximal j-subtree will have a larger effect on Δ_j than nodes closer to the terminal nodes.

2.2.3 Extension to Forest Ensembles

Ishwaran's framework extends naturally to forest ensembles and his theoretical result regarding forest ensembles provides some information of the behavior of variable importance measures for random forests. First for some notation, recall that in the forest ensemble setting, we draw B many bootstrap resamples of the training data to obtain the bootstrap replicates $Z^b = \{(\mathbf{x}_i^b, Y_i^b) | i = 1, \ldots, n\}$ of the training data for $b = 1, \ldots, B$. We then construct a binary regression tree $T(\mathbf{x}; b)$ on each bootstrap replicate of the data and have the forest $\hat{\mu}_F$ as the average of predictions over the trees $T(\mathbf{x}; b)$:

$$\hat{\mu}_F(\mathbf{x}) = \frac{1}{B} \sum_{b=1}^{B} \hat{\mu}(\mathbf{x}; b),$$

where $\hat{\mu}(\mathbf{x}; b)$ denotes the predictor for the tree $T(\mathbf{x}; b)$. Given that each $\hat{\mu}(\mathbf{x}; b)$ is a linear combination of basis functions, we can write

$$\hat{\mu_F}(\mathbf{x}) = \frac{1}{B} \sum_{b=1}^{B} \sum_{m=1}^{M^b} a_m^b B_m(\mathbf{x}; b).$$

Again assume that the $\mu(\mathbf{x})$ has a similar structure to $\mu(\mathbf{x})$. That is, $\mu(\mathbf{x}) = \frac{1}{B} \sum_{b=1}^{B} \sum_{m=1}^{M^b} a_{m,0}^b B_m(\mathbf{x}; b)$, where $a_{m,0}$ are the true, but unknown, terminal values. We denote the noised up forest predictor for the variable x_j by $\hat{\mu}_{F_j}(\mathbf{x})$. As with the normal forest predictor, the noised up forest predictor is the average of the noised up predictors $\hat{\mu}_j$ over the bootstrap resamples:

$$\hat{\mu}_j(\mathbf{x}) = \sum_{b=1}^{B} \hat{\mu}_j(\mathbf{x}; b).$$

As forest predictors are simply the average of individual trees, we can extend the lemma to $\hat{\mu}_{F_j}$ with the use of b's in appropriate places denoting the usage of the bth bootstrap resample. That is, we can write

$$\hat{\mu}_{F_j}(\mathbf{x}) = \frac{1}{B} \sum_{b=1}^{B} \left(\sum_{m \notin M_j^b} a_m^b B_m(\mathbf{x}; b) + \sum_{k=1}^{K_j^b} \tilde{a}_{k,j}^b \mathbb{I}\{T(\mathbf{x}; b) \in M_{k,j}^b\} \right).$$

The variable importance of the variable x_j is defined to be

$$\Delta_{F_j} = \mathbb{E}((Y - \hat{\mu}_{F_j}(\mathbf{x}))^2) - \mathbb{E}((Y - \hat{\mu}(\mathbf{x}))^2).$$

Similar to the single tree case, the variable importance of the variable x_j in the forest ensemble can be written as

$$\Delta_{F_j} = \mathbb{E}(R_{F_j}(\mathbf{x})^2) - 2\mathbb{E}\left(R_{F_j}(\mathbf{x})[\mu(\mathbf{x}) - \hat{\mu}_F(\mathbf{x})] \right),$$

where

$$R_{F_j}(\mathbf{x}) = \frac{1}{B} \sum_{b=1}^{b} \sum_{k=1}^{K_j^b} \sum_{m \in M_{k,j}^b} (\tilde{a}_{k,j}^b - a_m^b) B_m(\mathbf{x}; b).$$

We are now ready to state a result about the asymptotic form of $\Delta_{f,j}$:

Theorem 2.1. *Let $R_{F_j,0}(\mathbf{x})$ be the function $R_{F_j}(\mathbf{x})$, in which each instance of a_m^b has been replaced with $a_{m,0}^b$. Assume that Y can be written in terms of a regression model*

$$Y = \mu(\mathbf{x}) + \varepsilon$$

and that

$$\mu(\mathbf{x}) = \frac{1}{B}\sum_{b=1}^{B}\sum_{m=1}^{M^b} a_{m,0}^b B_m(\mathbf{x}; b).$$

If $a_m^b \to a_{m,0}^b$ for each m and b, then

$$\Delta_{F_j} \to \mathbb{E}\left(R_{F_j,0}(\mathbf{x})^2\right) \leq \mathbb{E}\left(\frac{1}{B}\sum_{b=1}^{b}\sum_{k=1}^{K_j^b}\theta_0(k,j,b)\right),$$

where

$$\theta_0(k,j,b) = \sum_{m\in M_{k,j}^b} \pi_{m,b}\tilde{P}_{k,j,0}^b\left(\tilde{a}_{k,j,0}^b - a_{m,0}^b\right)^2$$

is the node mean squared error for the kth maximal v-subtree of $T(\mathbf{x};b)$and $\pi_{m,b} = \mathbb{P}(B_m(\mathbf{x};b) = 1|Z)$.

[Note: explain node mean squared error. Also show a proof.]

In the above theorem, we

As Ishwaran (2007) notes, the bound in the above theorem becomes tighter as the trees in the forest become more and more orthogonal to each other. The above theorem is applicable if the forest predictor is consistent. This suggests that forest ensemble methods which are consistent and produce trees that are at least approximately orthogonal to each other allow for variable importance to be characterized through node mean squared error of subtrees. Variable importance as defined above differs from the MDA variable importance for random forests, but the two follow a similar heuristic towards measuring variable importance via the loss of predictive accuracy when the variable x_j is pertrubed. Therefore this result suggests that, perhaps, so long as the random forest estimator is consistent and the trees constructed are at least approximately orthogonal, the MDA variable importance can be characterized via the mean squared error of some analogous structure to maximal subtrees. This is, of course, conjectural, but we are interested whether for simpler forest algorithms, results similar to the theorem can be proven. Asymptotic results for the MDA and also MDI variable importance measures are difficult to formulate and prove for several reasons. First, the $CART$ tree construction process combined with the bootstrapping and randomization procedure of the random forest is difficult to mathematically analyze. Second, it is still unknown whether random forests constructed via $CART$ are consistent. However, there are other random forest variants known to be consistent, so those variants may be a natural starting point to try to prove asymptotic results about MDA and MDI variable importances. Third, the definition of MDA variable importance involves a permutation step that is difficult to analyze mathematically. The above results are mathematically amenable due to the noising-up procedure adopted allowing for a quite elegant analysis of the variable importance. One possible approach to dealing with this issue may be to adopt the above definition of variable importance in different random forest settings and see if similar results to Ishwaran (2007) may be derived. Another approach, which is the one more or less taken by Ishwaran (2007), is to to find a mathematically tractible definition of variable importance which is approximately the MDA variable importance measure and proceed with analysis from there.

[Lead into Strobl and the problem of collinearity and bias(bias as in which variables are chosen) in random forests]

Chapter 3

The Bootstrap

Conclusion

If we don't want Conclusion to have a chapter number next to it, we can add the `{-}` attribute.

More info

And here's some other random info: the first paragraph after a chapter title or section head *shouldn't be* indented, because indents are to tell the reader that you're starting a new paragraph. Since that's obvious after a chapter or section title, proper typesetting doesn't add an indent there.

Appendix A

The First Appendix

This first appendix includes all of the R chunks of code that were hidden throughout the document (using the `include = FALSE` chunk tag) to help with readibility and/or setup.

In the main Rmd file

```r
# This chunk ensures that the thesisdown package is
# installed and loaded. This thesisdown package includes
# the template files for the thesis.
if(!require(devtools))
  install.packages("devtools", repos = "http://cran.rstudio.com")
if(!require(thesisdown))
  devtools::install_github("ismayc/thesisdown")
library(thesisdown)
```

In Chapter ??:

Appendix B

The Second Appendix, for Fun

References

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

Efron, B. (2014). Estimation and accuracy after model selection. *Journal of the American Statistical Association*, *109*(507), 991–1007. `http://doi.org/doi.org/10.1080/01621459.2013.823775`

Ishwaran, H. (2007). Variable importance in binary regression trees and forests. *Electron J Stat*, 519–537.

Ishwaran, H. (2015). The effect of splitting on random forests. *Mach. Learn.*, *99*(1), 75–118. `http://doi.org/10.1007/s10994-014-5451-2`

Louppe, G. (2014). *Understanding random forests from theory to practice* (PhD dissertation). University of Liège.

Mentch, L., & Hooker, G. (2016a). Formal hypothesis tests for additive structure in random forests. Retrieved from `http://arxiv.org/abs/1406.1845`

Mentch, L., & Hooker, G. (2016b). Quantifying uncertainty in random forests via confidence intervals and hypothesis tests. *Journal of Machine Learning Research*, *17*(26), 1–41. Retrieved from `http://jmlr.org/papers/v17/14-168.html`