Pomona College

Senior Thesis in Mathematics

Prediction Error Estimation in Random Forests

Author:
Ian Krupkin

Advisor:
Dr. Jo Hardin

Submitted to Pomona College in Partial Fulfillment
of the Degree of Bachelor of Arts

May 12, 2023

Abstract

In this paper, error estimates of classification Random Forests are quantitatively assessed. Based on the initial theoretical framework built by Bates, Hastie, and Tibshirani [1], the true error rate and expected error rate are theoretically and empirically investigated in the context of a variety of error estimation methods common to Random Forests. We show that in the classification case, Random Forests' estimates of prediction error is closer on average to the true error rate instead of the average prediction error. This is opposite the findings of Bates, Hastie, and Tibshirani [1] which were given for logistic regression. We further show that this result holds across different error estimation strategies such as cross-validation, bootstrapping, and data splitting.

Contents

1 Introduction **1**

1.1 Literature Review 1

1.2 Goals of Paper 2

2 Modeling **3**

2.1 Bias-Variance Trade-Off 3

2.2 Finding the Optimal Model Complexity 5

2.2.1 The Validation Set 5

2.2.2 v-fold Cross Validation 6

3 Trees and Random Forests **7**

3.1 Building Random Forests 7

3.2 Out-of-Bag (OOB) Error 8

4 Setting and Notation **10**

5 Methods **13**

5.1 Data Generation and Settings 13

5.2 Strategies for Error Estimation 14

5.3 Empirical Estimation of Err_{XY} and Err 16

6 Results **20**

6.1 Distance to Target Errors 20

6.2 Distance Across Error Estimation Strategies 23

7 Discussion **27**

Chapter 1

Introduction

1.1 Literature Review

As evidenced by the 2017 State of Data Science and Machine Learning report by Kaggle, almost half of data scientists use Random Forests at work [2]. Random Forests [3] have become a popular classification tool in a variety of fields, especially because of their excellent performance in very complex data settings. The fact that out-of-bag (OOB) errors are theoretically and computationally simple improvements over a train-test split, lead to their ubiquity. When deploying a predictive model, it is important to understand its prediction accuracy on future test points, so both good point estimates and accurate confidence intervals for prediction error are essential. When Random Forests are implemented, the OOB error is a widely-used approach for point and interval estimate tasks, but in spite of OOB's seeming simplicity, its properties remain opaque. In the past, the OOB error has been affirmed to be an unbiased estimate of the true error rate [4, 5]. Nonetheless, it has been shown that for two-class classification problems the OOB error can overestimate the true prediction error [6, 7]. It was later argued that the use of stratified subsampling with sampling fractions that are proportional to response class sizes of the training data yielded almost unbiased error rates [8]. The present work is primarily concerned with OOB errors, but also addresses other common methods such as data splitting and cross validation, as well as their combination with OOB errors.

Despite the apparent straightforwardness of data splitting, cross validation, and bootstrapping, the formal properties of these modeling techniques are subtle. The question of “what are we estimating?” rightfully reappears often. For the linear model, fit by ordinary least squares, it was proven that the most popular estimates of prediction error (including data splitting, bootstrapping, cross-validation, etc.) are closer in expected value to the average prediction error of models fit on other unseen training sets drawn from the same population than the prediction error of the model at hand [1]. In other words, although the error of the model fit on the training data may seem like a reasonable estimand, it is not the closest error target. To our knowledge, there are no studies investigating this result in the case of Random Forests, as well as with various different model building workflows.

1.2 Goals of Paper

The main contribution of this thesis is two-fold: (i) the proximity of Random Forests' error estimates to the error targets presented by Bates et al. [1] is compared to the proximity of logistic regression to its error targets, through studies with different numbers of observations and predictors and (ii) the performance of a variety of error estimation strategies is explored.

The paper is structured as follows: In Chapter 2, we review the statistical modeling workflow and its considerations. In Chapter 3, we build up the random forest algorithm and the seemingly advantageous out-of-bag error. In Chapter 4, we set up notation and introduce the error targets: true error rate (Err_{XY}) and expected true error (Err). Subsequently, Chapter 5, introduces simulation-based studies. The descriptions include an outline of the simulated data, the considered settings, and several model building workflows that will be investigated. In Chapter 6, we present the results of the studies. The results are discussed in Chapter 7 alongside recommendations.

Chapter 2

Modeling

2.1 Bias-Variance Trade-Off

When creating statistical models, one intention is to predict the outcome of future data with low prediction error. Prediction errors can be decomposed into two components: error due to bias and error due to variance. There is a trade-off between a model's ability to minimize bias and variance. Understanding these two types of error can help us diagnose model results and avoid the mistake of over- or under-fitting the model to the observations. We define the error of a model due to bias as the average difference between the expected prediction of our model and the correct value which we are trying to predict. The error due to variance is taken as the variability of a model prediction for a given data point.

At its root, dealing with bias and variance is really about dealing with over and under-fitting. Bias is reduced and variance is increased as model complexity increases. Illustrated in Figure 2.1 [9], as additional parameters are introduced into a model, the complexity of the model rises and variance becomes our primary concern while bias steadily falls. The modeling process can shed light on an optimal model complexity. In this work we do not discuss the "double descent" curve that describes the improved performance that can be achieved by increasing model complexity beyond the point of interpolation. See Belkin et al. [10] for further discussion of this phenomenon in a wide spectrum of models and data sets.

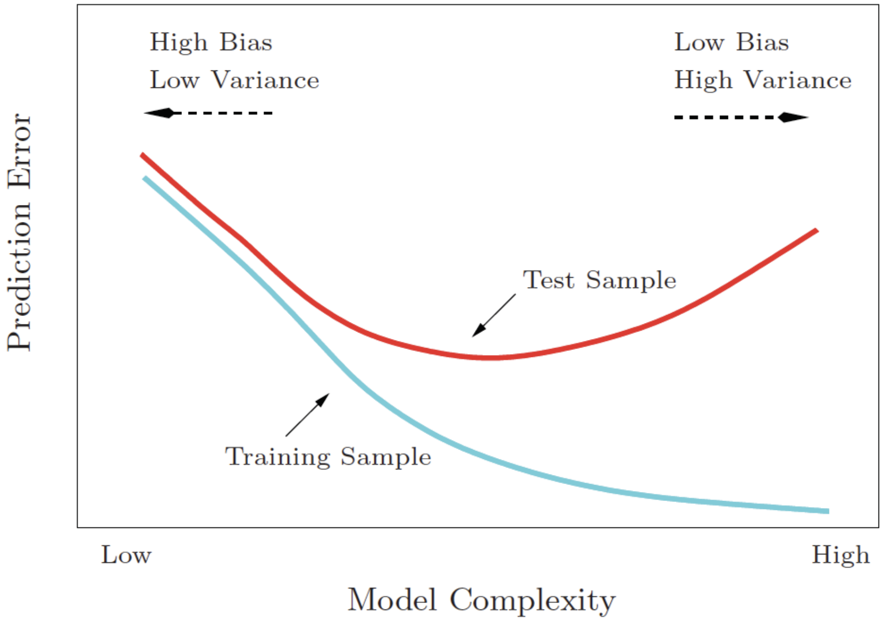

Figure 2.1: Test and training error as a function of model complexity. [9]

2.2 Finding the Optimal Model Complexity

The process of evaluating a model's performance is known as **model assessment**, whereas the process of selecting the proper level of flexibility for a model is known as **model selection**. Given a data set, the use of a particular statistical learning method and an associated appropriate level of model complexity is warranted if it results in a low test error. The test error can be easily calculated if a designated test set is available. Unfortunately, splitting the original data set into training and testing sets reduces power because the model is trained on a smaller sample. In contrast, the training error can be easily calculated by applying the statistical learning method to the observations used in its training. But as has been shown empirically, the training error rate often is quite different from the test error rate, and in particular the former can dramatically underestimate the latter [11].

Instead of creating a large designated test set to directly estimate the test error rate, a number of techniques can be used to estimate the **test** error rate using the available **training** data. The test estimate is then useful to tune parameters and find a suitable level of model complexity as well as to measure the performance of the fitted model on a new data set. Below the validation set and cross validation approaches are discussed. Later, the out-of-bag error is discussed as an alternative to test/train and cross validation.

2.2.1 The Validation Set

The first method to estimate the test error associated with fitting a particular statistical learning method on a set of training observations is the validation set approach. It involves randomly dividing the available set of observations into two parts, a training set and a validation set. The model is fit on the training set, and the fitted model is used to predict the responses for the observations in the validation set. Hence, the validation set error rate provides an independent estimate of the test error rate. This validation set approach is simple and straightforward to implement, yet has the drawback of using fewer observations to train the model. Therefore, the validation set error rate tends to overestimate the test error rate for the model fit on the entire data set [11]. Cross validation can be seen as a refinement of the validation set approach that addresses the overestimation issue.

2.2.2 v-fold Cross Validation

There are many cross validation methods, but the most ubiquitous is v-fold cross validation, which has computational advantages over Leave-One-Out cross validation (LOOCV). v-fold cross validation entails randomly dividing the set of observations into v groups, or folds, of approximately equal size. The first fold is treated as a validation set (or out-of-fold observations), and the method is fit on the remaining $v - 1$ folds (or in-fold observations). The error rate is then computed on the observations that are out-of-fold. This procedure is repeated v times; each time, a different group of observations is treated as a validation set. The process results in v estimates of the test error, which are averaged to arrive at the estimate of the test error rate [11].

Chapter 3

Trees and Random Forests

3.1 Building Random Forests

To understand Random Forests, we must first grasp the concept of a decision tree. Decision trees are a supervised learning model where prediction is done through recursive binary splitting based on an impurity metric. As implied by their name, Random Forests are an ensemble of decision trees, introduced by Breiman [3]. The power of Random Forests comes from the wisdom of crowds. A large number of relatively uncorrelated models operating as a committee will outperform any of the individual constituent models. While some trees may be wrong, many other trees will be right. Hence, as a group the trees are able to move in the correct direction. We will dive deeper into the process of creating these uncorrelated models to better understand the advantages of Random Forests.

To create many models, many "samples" are used. Tools such as boosting and bagging can assist us in this process. Assume that we have collected one sample on the population of interest. In order to create many models, the bootstrapping method creates many re-samples, of the original random sample, with replacement. Therefore, these bootstrapped samples represent proxy samples from the population. Using bootstrapping, we can create many samples, apply our statistical method to each one, and then average their predictions to obtain a final result [12]. The result of aggregating over multiple trees built on bagged re-samples are bagged decision trees.

Similar to bagged decision trees, Random Forests build numerous decision trees on bootstrapped training samples. However, with the motivation of infusing extra variability and then averaging over that variability, Random Forests include one additional step of variability. At each split in each tree of a random forest, a random subset of predictor variables is considered at every node in the tree. Each time a split in a decision tree is considered, a random sample of m predictors are chosen as candidates from the set of p predictors. This parameter, called *mtry*, is to be tuned. In other words, the random forest algorithm does not consider all of the available predictors at each split. In contrast, bagged decision trees consider all available predictor variables at every node.

Suppose there are two very strong predictors along with noise predictors. A collection of bagged trees will use the strong predictor at the top split almost exclusively, creating similar trees, and thus highly correlated predictions, even though they are built on different bootstrap samples. Random Forests solve this problem by forcing each tree to be even more different, due to the limited predictors considered at each node, and creating uncorrelated trees.

Bagging and other resampling techniques can be used to reduce the variance in model predictions. In Random Forests, the bias of the full forest is equivalent to the bias of a single decision tree (which itself has low bias and high variance) [9]. However, by creating many trees and then averaging them, the variance of the final forest can be greatly reduced over that of a single tree. In practice, the only limitation on the size of the forest is computing time as an infinite number of trees could be trained without ever increasing bias and with a continual (if asymptotically declining) decrease in the variance.

3.2 Out-of-Bag (OOB) Error

One of the advantages of the bagging algorithm of Random Forests is the ability to avoid the need for cross-validation or a validation set to get an unbiased estimate of the prediction error. Rather, it is estimated internally when building the tree. The process is as follows:

1. Build many trees based on bootstrapped samples.

2. On average, and for large sample sizes, the b^{th} tree does not use

$$\lim_{n\to\infty}(1-\frac{1}{n})^n = \frac{1}{e} \approx \frac{1}{3}$$

of the observations. These observations are referred to as OOB observations for that tree.

3. Predict the response for the i^{th} observation using the trees where that observation was OOB.

4. Average the $\approx \frac{B}{3}$ OOB predictions for the i^{th} observation in regression (or take a majority vote for classification) to obtain a single prediction for the i^{th} observation, where B is the number of trees used in the random forest.

5. Let the OOB prediction for the i^{th} observation be $\hat{y}_{(-i)}$. Therefore,

$$\begin{aligned} \text{OOB}_{\text{error}} &= \frac{1}{n}\sum_{i=1}^{n} \mathbf{1}(y_i \neq \hat{y}_{(-i)}) \quad \text{for classification} \\ \text{OOB}_{\text{error}} &= \frac{1}{n}\sum_{i=1}^{n} (y_i - \hat{y}_{(-i)})^2 \quad \text{for regression} \end{aligned}$$

The convenience of having the OOB error measure comes in the form of computing time. When fitting a random forest to a data set, the out-of-bag error is calculated simultaneously. The advantage of using the OOB error, while not producing biased error estimates, is investigated in this work.

Chapter 4

Setting and Notation

Before turning to our main method in the next section, we introduce our notation and review topics related to error targets. We consider the supervised learning setting where we have p features $X = (X_{i,1}, \ldots, X_{i,p}) \in \mathbb{R}^n \times \mathbb{R}^p$ and response $Y = (Y_1, \ldots, Y_n) \in \mathbb{R}^n$. We assume that the data points (X_i, Y_i) for $i = 1, \ldots, n$ are independent and identically distributed from some underlying distribution P on $\mathbb{R}^n$. We wish to understand the performance of our fitted model when generalized to unseen data points, which can be formalized by a loss function:

$$\ell(\hat{y}, y) : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}_{\geq 0}$$

such that $\ell(y, y) = 0$ for all y. The form of ℓ need not be specified and could be squared error loss, misclassification error, cross-entropy, etc. Now consider a model $f(\cdot)$ parameterized by θ. Let $\hat{f}(x, \hat{\theta})$ be the function that predicts y from $x \in \mathbb{R}^p$ using the model with parameters θ, which take values in the space Θ. Let $\mathcal{A}$ be a model-fitting algorithm that takes any number of data points and returns a parameter vector $\hat{\theta} \in \Theta$. Hence, $\hat{\theta} = \mathcal{A}(X, Y)$ is the fitted value of the parameter based on the observed data X and Y. Let $(X_{n+1}, Y_{n+1}) \sim P$ be another independent test point from the same distribution. Using the training data, we are interested in finding the function $\hat{f}(x, \theta)$ that minimizes the loss $\ell(\hat{f}(X_{n+1}, \theta), Y_{n+1})$. Note that $\ell(\hat{f}(X_{n+1}, \theta), Y_{n+1})$ is a random and unknown object, and our target is one of two quantities:

$$\text{True Error Rate: } \mathrm{Err}_{XY} = E[\ell(\hat{f}(X_{n+1}, \hat{\theta}), Y_{n+1})|(X, Y)] \tag{4.1}$$

$$\text{Expected Error Rate: } \mathrm{Err} = E[\ell(\hat{f}(X_{n+1}, \hat{\theta}), Y_{n+1})] = E[\mathrm{Err}_{XY}] \tag{4.2}$$

These are the two most natural quantities of interest to the analyst. Known as the *true error rate*, Err_{XY} is the expected test error of the model that was fit on our actual training set. Err is the expected average error of the fitting algorithm run on the same-sized data sets drawn from the underlying distribution P, and called the *expected error rate*. It is important to note that the random variable Err is a constant with respect to (X, Y), while Err_{XY} is a function of (X, Y) [1, 13].

In practice, we usually seek to grasp Err_{XY}, however, Err is sometimes used as a quantity of interest. The former quantity is of the most interest to a practitioner deploying a specific model, whereas the latter may be of interest to a researcher comparing different fitting algorithms. To illustrate this difference consider the following examples.

Suppose Statistician A is trying to estimate the average height of penguins, found in the wild, based on a sample of 100 researched penguins. Statistician A will use the original sample to build a model to estimate the average height of the next sample of penguins. When presenting the model, they will be interested in the true error rate (Err_{XY}) of the model because they will want to know how the specific model they built will perform on the next data set.

However, if Statistician B is trying to accomplish the same task of estimating the average height of penguins, based on a finite sample, but is unsure of the structure of the model to utilize, they will be interested in a slightly different error metric. Statistician B will run a variety of fitting algorithms to build numerous models based on the sample available and will need to compare the models. They will want to estimate the expected error rate (Err) to know the average error of the fitting algorithm run on same-sized data sets drawn from the underlying distribution. Statistician B is less interested in the performance of the model which was built using the sample at hand (which is Statistician A's target), but rather the performance of the process of arriving at the model.

While it may initially appear that the quantity Err_{XY} is easier to estimate, since it concerns the model at hand, it has been observed that the cross-validation estimate of error is farther from Err_{XY} than Err [14]. This issue, as it is called, is mainly attributed to data re-usage.

Let (X_∞, Y_∞) represent a data set of unlimited size enabling the best possible model $f(\cdot)$ to be chosen. Theoretically, Err_{XY} can be decomposed into four parts [15]:

$$\mathrm{Err} = E[\ell(f(X_\infty, \theta), Y_\infty)] \qquad \text{Best possible performance} \tag{4.3}$$

$$+ E[\ell(\hat{f}(X_\infty, \hat{\theta}), Y_\infty) - \ell(f(X_\infty, \theta), Y_\infty)] \qquad \text{Model selection cost} \tag{4.4}$$

$$+ E[\ell(\hat{f}(X_{n+1}, \hat{\theta}), Y_{n+1}) - \ell(\hat{f}(X_\infty, \hat{\theta}), Y_\infty)] \qquad \text{Parameter estimation cost} \tag{4.5}$$

$$+ E[\ell(\hat{f}(X, \hat{\theta}), Y) - \ell(\hat{f}(X_{n+1}, \hat{\theta}), Y_{n+1})] \qquad \text{Data re-use cost} \tag{4.6}$$

The most interesting component is the final term (4.6), caused by data re-usage, has a non-zero expectation when the same data points are used for both model selection and parameter estimation. If one uses a validation set approach, (4.6) will have an expectation of zero because each observation is only used once. However, as estimated empirically, when using a full data approach, this term (4.6) can be large and easily outweigh the advantages the full data has in model selection and parameter estimation [15]. Thus, the full data strategy will have lower model selection and parameter estimation costs than the validation set strategy due to the higher number of observations used to complete the model selection and parameter estimation processes.

The difference between the full data and validation set strategies in (4.4) & (4.5) is bounded and well understood as an effect of sample size [15]. Despite suffering in model selection and parameter estimation costs, the validation set strategy will have a lower data re-use cost than the full data strategy, and we know the data re-use cost term (4.6) could be very large. Therefore, we would like to investigate the situations when the data re-use cost outweighs the model selection and parameter estimation costs. This comes in the form of analyzing various model fitting approaches and when they provide more accurate estimates of error. Specifically the use of OOB errors compared to validation set and cross validation strategies is investigated.

Chapter 5

Methods

In this section, some simulation-based studies are described. Simulated data are used to study the behavior of modeling strategies in simple settings, in which all predictor variables are uncorrelated. The results provide insight into the mechanisms which lead to different targets in error estimates.

5.1 Data Generation and Settings

The bias of error estimates in different data settings with numeric predictor variables was systematically investigated by means of simulation studies in balanced binary two-class response variable data. The settings considered were:

- Different number of predictors, $p \in \{10, 100\}$.
- Different number of observations such that $n < p, n > p, n \gg p$.

As done when modeling real data, several Random Forests with different *mtry* values were constructed for each setting. The values for *mtry* ranged from $mtry = 1$ all the way up to $mtry = p$. Note that for $mtry = 1$ there is no selection of an optimal predictor variable for a split, while for $mtry = p$ the random forest method coincides with the bagging procedure which selects the best predictor variable from the entire set of predictors.

The number of trees chosen is a trade-off between accuracy and computational speed. More trees are necessary when using a large number of predictor variables. The OOB error stabilized at around 250 trees in convergence studies [16], and they concluded that 1000 trees might be sufficiently large for their genome-wide data set of more than 300,000 predictor variables. Also in high-dimensional settings, Random Forests with 500 trees and 1000 trees yielded very similar OOB errors [17]. In accordance with these findings the number of trees was set to 500 in all studies of this paper. Each setting was repeated 1000 times to obtain stable results.

Only numeric predictor variables are considered in the studies. Both predictors associated with the response and predictors not associated with the response were considered, with all predictors still independently distributed. The predictors not associated with the response followed a standard normal distribution. The distribution of predictors with association was different for each response class. The predictor values for observations from class 1 were always drawn from a standard normal distribution. The predictor values for observations from class 2 were drawn from a normal distribution with variance 1 and a mean different from zero. Figure 5.1 gives an overview of the distribution of predictors in the response classes. Let us consider the setting with $p = 10$ as an example. The first two predictors X_1 and X_2 are associated with the response, while the other predictors $X_3, \ldots, X_{10}$ are noise. Hence, $X_3, \ldots, X_{10}$ follow a standard normal distribution, while the distributions of X_1 and X_2 depend on the class to which the observations belong. If the observation comes from class 1, the distribution of X_1 and X_2 is $N(0, 1)$, and X_1 and X_2 are distributed $N(0.75, 1)$ for class 2. Randomly drawing the mean separately for X_1 and X_2 and for each repetition of the study makes sure that predictors with different effect strengths are considered.

It is important to note that the settings are simplistic because all predictors are uncorrelated. Although assuming no correlations between any of the predictors is not realistic, such settings are important to understand the mechanisms which lead to different targets in error estimation.

5.2 Strategies for Error Estimation

In simple terms, the modeling process consists of parameter estimation, followed by error estimation. An important point of consideration when completing the two estimation steps is the choice of which subset of observations will be used in each operation. Often, data for parameter estimation and data for error estimation are collected at the same time, thus resulting in

Number of Predictors	Predictors	Class 1 $N(\mu_1, 1)$	Class 2 $N(\mu_2, 1)$
$p = 10$	X_1	$\mu_1 = 0$	$\mu_2 \sim N(0.75, 1)$
	X_2	$\mu_1 = 0$	$\mu_2 \sim N(0.75, 1)$
	$X_3, \ldots, X_{10}$	$\mu_1 = 0$	$\mu_2 = 0$
$p = 100$	X_1	$\mu_1 = 0$	$\mu_2 \sim N(0.75, 1)$
	$\vdots$	$\vdots$	$\vdots$
	X_{10}	$\mu_1 = 0$	$\mu_2 \sim N(0.75, 1)$
	$X_{11}, \ldots, X_{100}$	$\mu_1 = 0$	$\mu_2 = 0$

Figure 5.1: Distribution of predictors in class 1 and class 2 of the simulated data set up as in Janitza and Hornung [8].

a single sample that needs to be apportioned to both parameter and error estimation. As described in Figure 2.1, finding the optimal model complexity requires an external test set. In an ideal world, to avoid “data snooping”, one needs one data set for model building, one for parameter estimation, and then after a model is accepted, another data set for error estimation. However, this is not always possible because of constraints, so one may need to do the best one can with the data available. Hence, when modeling it is important to outline the strategy that will be used to construct the model and then estimate its error.

Various strategies were chosen to separately target the parameter estimation and error estimation steps in the modeling process, and thus each strategy consists of two parts. The following strategies were considered:

- Logistic Regression CV Error (LGCV): The logistic model is built on the in-fold data set and the error of the model is estimated via 4-fold cross-validation.

- Full Data Set CV Error (FDCV): Parameters are set prior to model building with $mtry = \sqrt{p}$. The random forest is built on the in-fold data set and the error of the model is estimated via 4-fold cross-validation.

- Full Data Set OOB Error (FDO): Parameter and error estimation is done on the same data set. *mtry* is chosen by using the OOB error rate.

Hence, the random forest with the lowest OOB error rate is chosen and the OOB error is returned as the error estimate.

- Split Data Set OOB Error (SDO): The sample is divided into training and testing sets. Parameter estimation is done on the training set. The model is chosen by using the OOB error rate on the training set. The error of the random forest, built on the entire training set, is then estimated by predicting on the testing set with an accuracy measure as the error estimate.
- Split Data Set CV Error (SDCV): The sample is divided into training and testing sets. Parameter estimation is done on the training set, using 4-fold cross-validated error estimates to select *mtry*. The error of the random forest, built on the entire training set, is estimated by predicting on the testing set with an accuracy measure as the error estimate.
- Split Data Set Test Error (SDT): The sample is divided into training, validation, and testing sets. Parameter estimation is done on the training and validation sets. The Random Forests are fit on the training set, and the fitted forests are used to predict the responses for the observations in the validation set. The model with the highest accuracy is chosen. The error of this random forest, built solely on the training set, is estimated by predicting on the testing set and obtaining an accuracy measure, and thus uses observations that are not part of the set of observations that are considered for constructing the Random Forest.

The workflow for the Split Data Set Test Error (SDT) is illustrated in Figure 5.3.

5.3 Empirical Estimation of Err_{XY} and Err

The estimation of the theoretical quantities, Err_{XY} and Err, deepens the understanding of the difference between the two. As mentioned above, the *true error rate*, Err_{XY}, is the test error of the model that was fit on our actual training set. Hence, the estimation of this quantity is the error produced by the model on a new theoretically infinitely large test set. As illustrated in Figure 5.4, the sample is used to create the model, and then this model is

Strategy	Model Fitting	Parameter Estimation	Error Estimation
LGCV	In-Fold observations from full data set	N/A	Out-of-Fold observations from full data set
FDCV	In-Fold observations from full data set	N/A	Out-of-Fold observations from full data set
FDO	In-Bag observations from full data set	OOB observations from full data set	OOB observations from full data set
SDO	In-Bag observations from training data set	OOB observations from training data set	Test data set
SDCV	In-Fold observations from training data set	Out-Fold observations from training data set	Test data set
SDT	Training data set	Validation data set	Test data set

Figure 5.2: Data used in each step of the error estimation strategies

used to predict on a large test set, from the underlying population. The missclassification rate on the test set will be the *true error rate*.

Moreover, Err is the average error of the fitting algorithm run on the same-sized data sets drawn from the underlying distribution P, and called the *expected error rate*. When calculating the *expected error rate*, the average of the *true error rate*, but using a new model every time, is taken. As seen in Figure 5.4, the entire model fitting process is repeated to obtain each Err_{XY} from one sample.

In other words, the difference between the estimation of Err_{XY} and Err is that the former uses one model, while the latter averages over many models. As seen from Equation 4.1, Err_{XY} is conditional on the data, while Err is unconditional. Note that Err averages over everything that is random, including the randomness in the training set that produced the model.

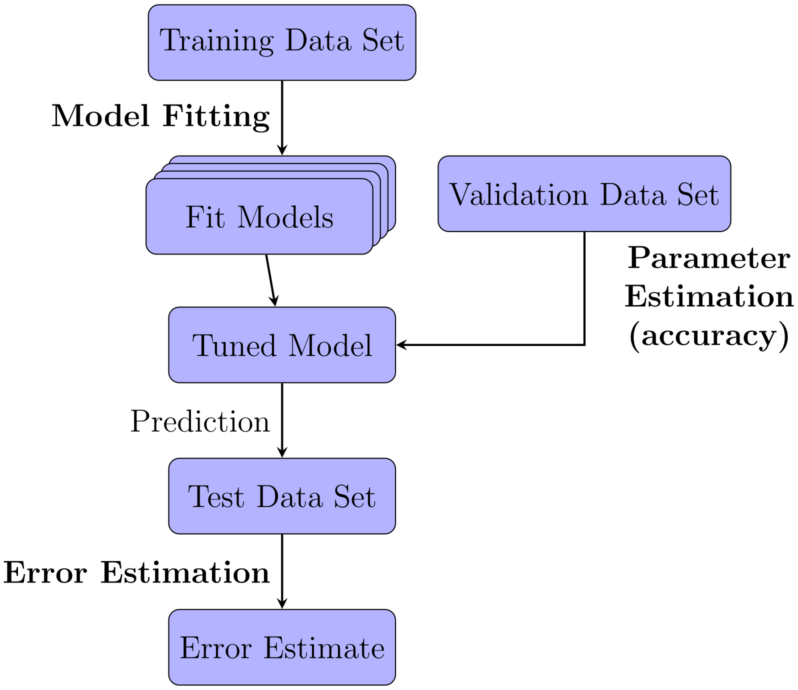

Figure 5.3: Workflow for SDT strategy

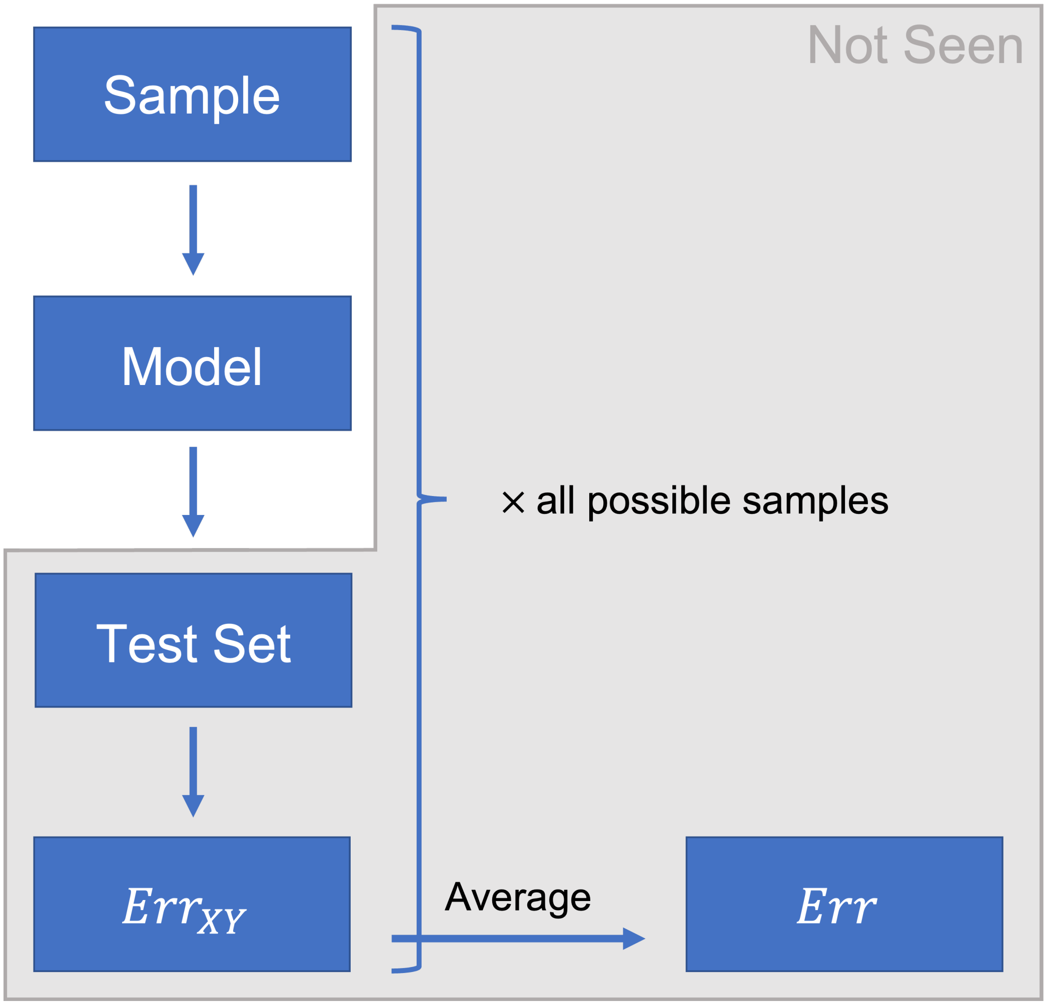

Figure 5.4: Empirical estimation of Err_{XY} and Err.

Chapter 6

Results

6.1 Distance to Target Errors

Recall, when creating statistical models, one wants estimates of error that are close to the truth and low. But that begs the question: close to what truth, Err_{XY} or Err? In an effort to compare the results of our simulations to those of Bates, Hastie, and Tibshirani [1], Figure 6.1 shows the distance of $\widehat{\mathrm{Err}}^{(\mathrm{LGCV})}$ from Err compared to its distance from Err_{XY}. Similar to Bates, Hastie, and Tibshirani, we can see that in the logistic regression model,

$$|\widehat{\mathrm{Err}}^{(\mathrm{LGCV})} - \mathrm{Err}_{XY}| > |\widehat{\mathrm{Err}}^{(\mathrm{LGCV})} - \mathrm{Err}|$$

The difference is more pronounced with $n < p$ and lessens as $n \to \infty$. Regardless, repeated simulations consistently show that $\widehat{\mathrm{Err}}^{(\mathrm{LGCV})}$ is on average closer to Err than Err_{XY}.

In Figure 6.2 we see that this relationship has flipped for Random Forests. $\widehat{\mathrm{Err}}^{(\mathrm{FDCV})}$ is closer to Err_{XY} than Err. As a reminder, the difference between $\widehat{\mathrm{Err}}^{(\mathrm{LGCV})}$ and $\widehat{\mathrm{Err}}^{(\mathrm{FDCV})}$ is that the former is an error estimate for a logistic model while the latter a random forest. Both are cross-validated estimates on the in-fold data set with no parameter tuning. Thus, the relationship highlighted by Bates, Hastie, and Tibshirani [1] seems to be specific to generalized linear models as they investigated linear and logistic regression models.

The flip in relationship may be attributed to the difference in the way each model utilizes the data. In logistic regression, the coefficients are estimated via maximum likelihood estimation, thus possibly leading to over-fitting and

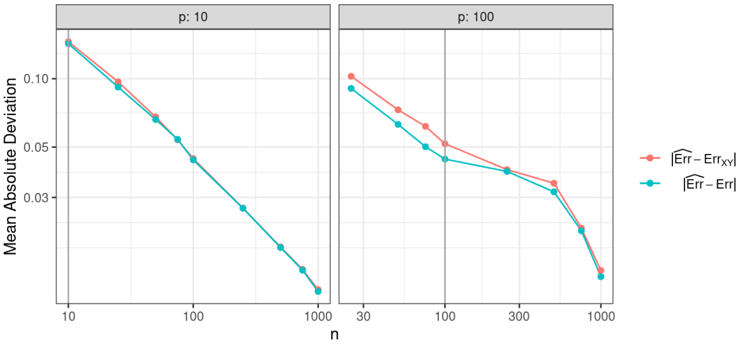

Figure 6.1: Simulation results comparing the error of cross-validated estimates of a logistic regression model when estimating Err to its error when estimating Err_{XY}: the mean absolute deviation between $\widehat{\text{Err}}^{(\text{LGCV})}$ and Err or Err_{XY}. The dark grey vertical line in each panel is where $n = p$. Notice that across all n, $\widehat{\text{Err}}^{(\text{LGCV})}$ is closer to Err than Err_{XY}.

biased estimates of error. On the other hand, Random Forests are infused with extra variability through resampling methods (as mentioned in Chapter 3), and therefore the model is built on different observations and different variables at each step. As a result, the logistic regression model may be less informative on the "next" sample, than a random forest. Hence, $\widehat{\text{Err}}$ is closer to Err_{XY} than Err for Random Forests because its resampling methods build the model on data more akin to wild data.

We further explore the difference between $\widehat{\text{Err}}$ for logistic regression and Random Forests in an experiment with $n = 50$ observations and $p = 10$ features; see Figure 6.3. In the right plot, there seems to exist a pairing between $\widehat{\text{Err}}^{(\text{FDCV})}$ and Err_{XY}, where high estimates of $\widehat{\text{Err}}^{(\text{FDCV})}$ are paired with high estimates of Err_{XY} and vice-versa (i.e. very few of the linking lines cross). In a logistic regression model, there does not seem to exist this pairing as seen in the left plot (i.e. most of the linking lines cross). In other words, $\widehat{\text{Err}}^{(\text{LGCV})}$ tends to be closer to Err than Err_{XY}, and $\widehat{\text{Err}}^{\text{FDCV}}$ tends to be closer to Err_{XY} than to Err.

Returning to the remaining strategies that all use Random Forests, Fig-

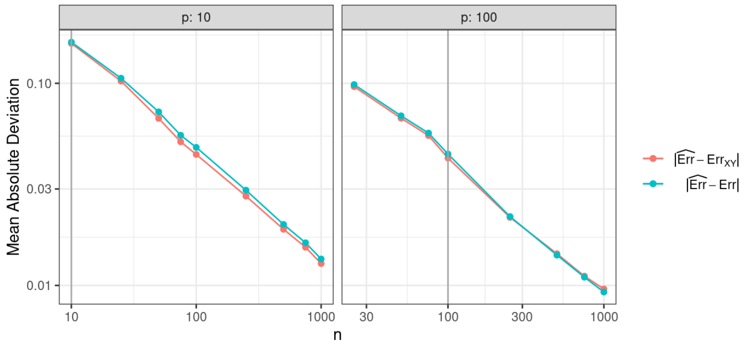

Figure 6.2: Simulation results comparing the error of CV estimates of a random forest model when estimating Err to its error when estimating Err$_{XY}$: the mean absolute deviation between $\widehat{\text{Err}}^{(\text{FDCV})}$ and Err or Err$_{XY}$. The dark grey vertical line in each panel is where $n = p$. Notice that across all n, $\widehat{\text{Err}}^{(\text{FDCV})}$ is closer to Err$_{XY}$ than Err.

ures 6.4 - 6.7 show that across the error estimation strategies, $\widehat{\text{Err}}$ is closer to Err$_{XY}$ than Err. Despite this relationship, the differences in mean absolute deviations, from Err$_{XY}$ and Err, tend to be quite small.

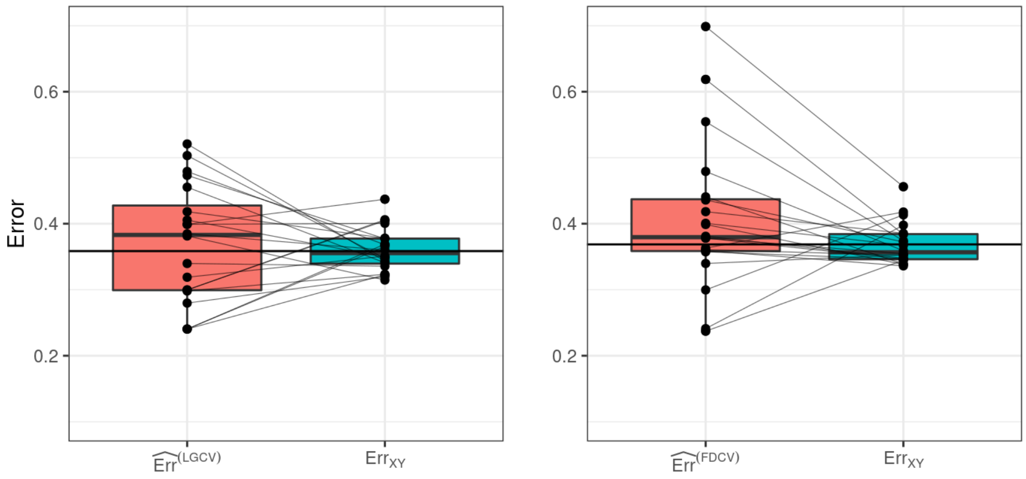

Figure 6.3: Random sample of 20 iterations of $\widehat{\mathrm{Err}}$ linked with the corresponding Err_{XY} for logistic regression (left plot) as compared to Random Forests (right plot). Solid black horizontal line is Err.

6.2 Distance Across Error Estimation Strategies

Section 6.1 detailed our investigation of the distance of the estimates of error to Err_{XY} and Err. We showed that in Random Forests, the error estimates are closer to Err_{XY} than to Err. Here we assess a follow-up question: how close is $\widehat{\mathrm{Err}}$ to Err_{XY}? Figure 6.8 compares the strategies according to the expected value of $|\widehat{\mathrm{Err}} - \mathrm{Err}_{XY}|$.

In the case of $p = 10$ features, the strategies that utilize the in-fold data set to train the model (LGCV, FDCV, and FDO) seem to outperform the split data approaches. The models seem to not over-fit and therefore the error estimates do suffer a drop in performance as a result of train/test splits. It is important to mention that FDO is the only strategy of the three that tunes parameters. Moreover, FDO seems to act poorly when n is small.

In the case of $p = 100$ features, FDCV seems to be the best candidate

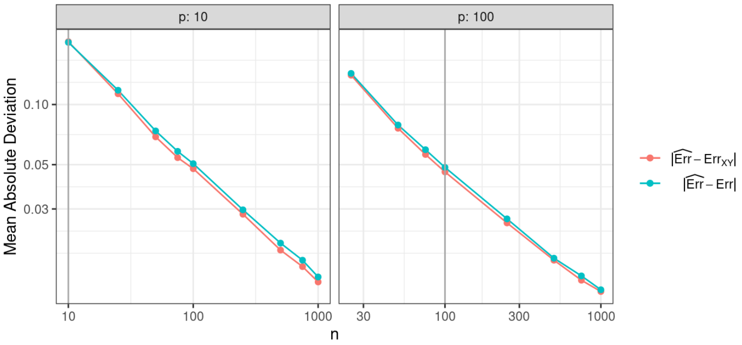

Figure 6.4: Simulation results comparing the error of OOB estimates of a random forest model when estimating Err to its error when estimating Err_{XY}: the mean absolute deviation between $\widehat{\mathrm{Err}}^{(\mathrm{FDO})}$ and Err or Err_{XY}. The dark grey vertical line in each panel is where $n = p$.

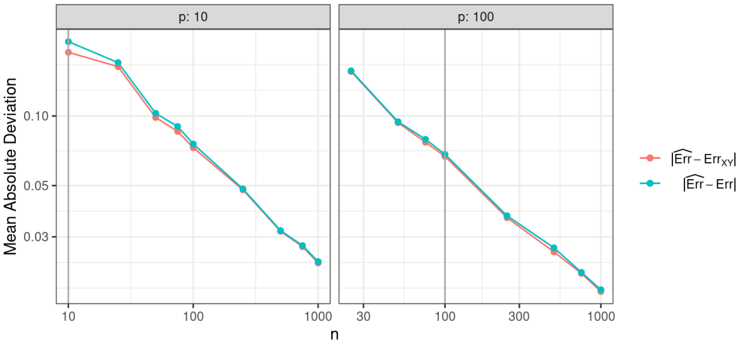

Figure 6.5: Simulation results comparing the error of OOB estimates of a random forest model when estimating Err to its error when estimating Err_{XY}: the mean absolute deviation between $\widehat{\mathrm{Err}}^{(\mathrm{SDO})}$ and Err or Err_{XY}. The dark grey vertical line in each panel is where $n = p$.

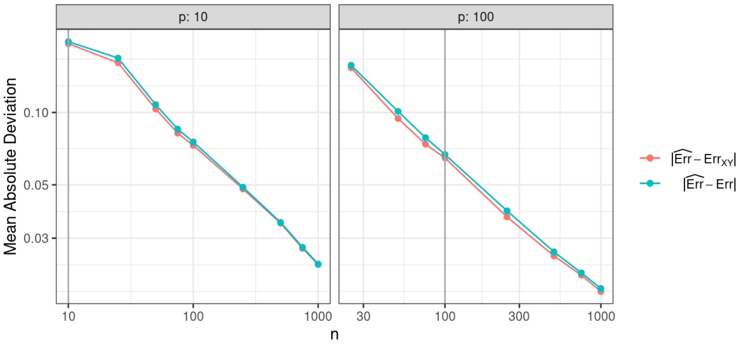

Figure 6.6: Simulation results comparing the error of CV estimates of a random forest model when estimating Err to its error when estimating Err_{XY}: the mean absolute deviation between $\widehat{\mathrm{Err}}^{(\mathrm{SDCV})}$ and Err or Err_{XY}. The dark grey vertical line in each panel is where $n = p$.

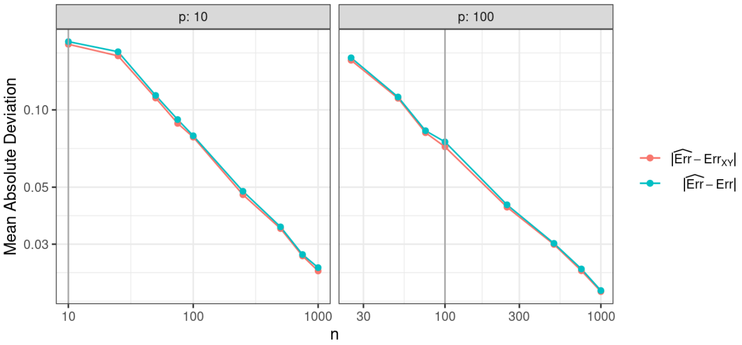

Figure 6.7: Simulation results comparing the error of validation set estimates of a random forest model when estimating Err to its error when estimating Err_{XY}: the mean absolute deviation between $\widehat{\mathrm{Err}}^{(\mathrm{SDT})}$ and Err or Err_{XY}. The dark grey vertical line in each panel is where $n = p$.

when $n < p$ as well as $n > p$. As was the case when $p = 10$ features, FDO performs weakly with small n, but is akin to FDCV when $n = p$. In contrast to $p = 10$ features, FDCV and FDO perform better than LGCV when $n > p$. Once again, it is important to note that, out of the strategies that build the model on the entire data set, FDO tunes the model's parameters, compared to FDCV which does not.

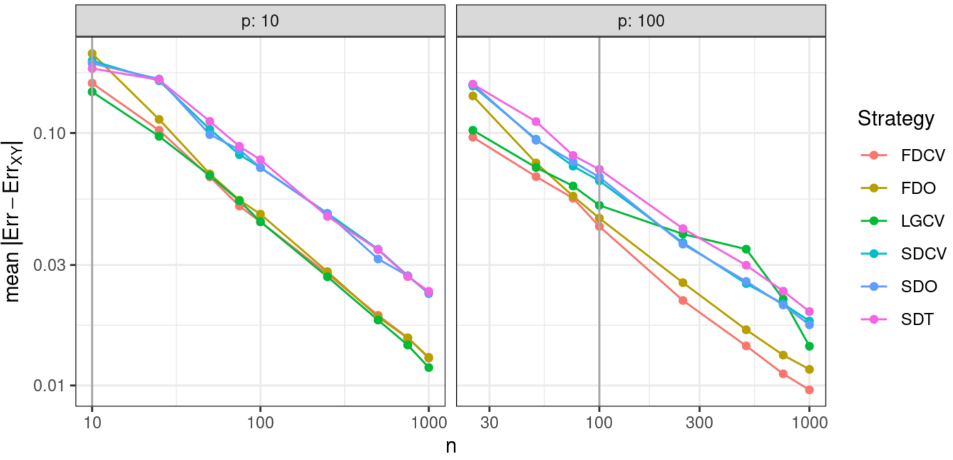

Figure 6.8: Mean absolute deviation between $\widehat{\mathrm{Err}}$ and Err_{XY} across error estimation strategies. The dark grey vertical line in each panel is where $n = p$.

Chapter 7

Discussion

This investigation had two main components. First, we discussed the difference in error targets presented by Bates et al. [1]. In their work, they found that in the special case of the generalized linear model using unregularized OLS for model-fitting, common estimates of prediction error — cross-validation, bootstrap, data splitting, and covariance penalties — should be viewed as estimates of the average prediction error, averaged across other hypothetical data sets from the same distribution. My primary result is that in the classification case, Random Forests' estimates of prediction error can be taken as an estimate of the true error rate instead of as an estimate of the average prediction error. In my simulations this result held across error estimation strategies such as cross-validation, bootstrapping, and data splitting (See Figures 6.2 - 6.7). The result was present regardless of n, p. Nonetheless, we wish to be clear that the estimates of prediction error were a good approximation of both the true error rate and expected error rate in the data splitting cases.

A fundamental open question is to understand the size of the gap of estimates of prediction error with the true error rate and expected error rate. The present work focused on which target the estimate is closer to. Moreover, it is necessary to understand under what conditions the gap is large, making it necessary to modify the method of error estimation depending on the target. Roughly speaking, we expect the gap to be small when n/p is large. In my experiments, the gap was always smaller than 1%. As n increases, however, the difference decreases. Other future directions are the investigation of this relationship in correlated data and imbalanced data.

Second, we discussed the performance of a variety of error estimation strategies. The models built on the entire sample were closer to the true error rate compared to those built on a training set and error estimates obtained from a testing set. Therefore, the data strategies that do not use a holdout set seem more appealing choice for model building, regardless if parameter tuning is to be performed or not. Empirically, the strategies that use resampling techniques as opposed to a holdout set are favorable.

Bibliography

[1] Stephen Bates, Trevor Hastie, and Robert Tibshirani. Cross-validation: what does it estimate and how well does it do it?, 2022.

[2] Kaggle. The state of data science amp; machine learning, 2017. URL `https://ailab-ua.github.io/courses/resources/the_state_of_data_science_machine_learning_-_kaggle_2017_survey.pdf`.

[3] L Breiman. Random forests. *Machine Learning*, 45:5–32, 10 2001. doi: 10.1023/A:1010950718922.

[4] Gai-Ying Zhang, Chun-Xia Zhang, and Jiang-She Zhang. Out-of-bag estimation of the optimal hyperparameter in subbag ensemble method. *Communications in Statistics - Simulation and Computation*, 39(10): 1877–1892, 2010. doi: 10.1080/03610918.2010.521277. URL `https://doi.org/10.1080/03610918.2010.521277`.

[5] Benjamin A Goldstein, Eric C Polley, and Farren B. S. Briggs. Random forests for genetic association studies. *Statistical Applications in Genetics and Molecular Biology*, 10(1), 2011. doi: doi:10.2202/1544-6115.1691. URL `https://doi.org/10.2202/1544-6115.1691`.

[6] Tom Bylander. Estimating generalization error on two-class datasets using out-of-bag estimates. *Machine Learning*, 48(1-3):287–297, 07 2002. URL `https://doi.org/10.1023/A:1013964023376`. Copyright - Kluwer Academic Publishers 2002.

[7] Matthew Mitchell. Bias of the random forest out-of-bag (oob) error for certain input parameters. *Open Journal of Statistics*, 01:205–211, 01 2011. URL `https://doi.org/10.4236/ojs.2011.13024`.

[8] Silke Janitza and Roman Hornung. On the overestimation of random forest's out-of-bag error. *PLOS ONE*, 13(8):1–31, 08 2018. doi: 10.1371/journal.pone.0201904. URL https://doi.org/10.1371/journal.pone.0201904.

[9] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. Springer Series in Statistics. Springer New York Inc., New York, NY, USA, 2001.

[10] Mikhail Belkin, Daniel Hsu, Siyuan Ma, and Soumik Mandal. Reconciling modern machine-learning practice and the classical bias–variance trade-off. *Proceedings of the National Academy of Sciences*, 116(32): 15849–15854, jul 2019. doi: 10.1073/pnas.1903070116. URL https://doi.org/10.1073%2Fpnas.1903070116.

[11] Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. *An Introduction to Statistical Learning: with Applications in R*. Springer, 2013. URL https://faculty.marshall.usc.edu/gareth-james/ISL/.

[12] B. Efron and R.J. Tibshirani. *An Introduction to the Bootstrap*. CRC Press, 1994. ISBN 9781000064988. URL https://doi.org/10.1201/9780429246593.

[13] Samyak Rajanala, Stephen Bates, Trevor Hastie, and Robert Tibshirani. Confidence intervals for the generalisation error of random forests, 2022.

[14] Waleed A. Yousef. A leisurely look at versions and variants of the cross validation estimator, 2019. URL https://arxiv.org/abs/1907.13413.

[15] Julian J. Faraway. Does data splitting improve prediction? *Statistics and Computing*, 26(1–2):49–60, Oct 2014. ISSN 1573-1375. doi: 10.1007/s11222-014-9522-9. URL http://dx.doi.org/10.1007/s11222-014-9522-9.

[16] Benjamin Goldstein, Alan Hubbard, Adele Cutler, and Lisa Barcellos. An application of random forests to a genome-wide association dataset: Methodological considerations new findings. *BMC genetics*, 11:49, 06 2010. URL https://doi.org/10.1186/1471-2156-11-49.

[17] Robin Genuer, Jean-Michel Poggi, and Christine Tuleau. Random forests: some methodological insights, 2008. URL `https://arxiv.org/abs/0811.3619`.