Statistical Applications in Genetics and Molecular Biology

Volume 10, *Issue* 1 2011 *Article* 32

Random Forests for Genetic Association Studies

Benjamin A. Goldstein, *Stanford University*
Eric C. Polley, *National Institutes of Health*
Farren B. S. Briggs, *University of California, Berkeley*

Recommended Citation:
Goldstein, Benjamin A.; Polley, Eric C.; and Briggs, Farren B. S. (2011) "Random Forests for Genetic Association Studies," *Statistical Applications in Genetics and Molecular Biology*: Vol. 10: Iss. 1, Article 32.
DOI: 10.2202/1544-6115.1691

Random Forests for Genetic Association Studies

Benjamin A. Goldstein, Eric C. Polley, and Farren B. S. Briggs

Abstract

The Random Forests (RF) algorithm has become a commonly used machine learning algorithm for genetic association studies. It is well suited for genetic applications since it is both computationally efficient and models genetic causal mechanisms well. With its growing ubiquity, there has been inconsistent and less than optimal use of RF in the literature. The purpose of this review is to breakdown the theoretical and statistical basis of RF so that practitioners are able to apply it in their work. An emphasis is placed on showing how the various components contribute to bias and variance, as well as discussing variable importance measures. Applications specific to genetic studies are highlighted. To provide context, RF is compared to other commonly used machine learning algorithms.

KEYWORDS: machine learning, SNP, genome wide association studies

Author Notes: Benjamin A. Goldstein, Quantitative Sciences Unit, Department of Medicine, Stanford University. Eric C. Polley, Biometric Research Branch, National Cancer Institute, National Institutes of Health. Farren B. S. Briggs, Genetic Epidemiology and Genomics Laboratory, University of California, Berkeley. The authors acknowledge Alan Hubbard, Lisa Barcellos and Adele Cutler for discussing and reviewing aspects of this work. BAG was funded in part by a National Institutes of Health NRSA Trainee appointment on grant T32 HG 00047 and the Russell M. Grossman Endowment. FBSB is a National Multiple Sclerosis Society Post-Doctoral Fellow (FG 1847A1/1)

1 Introduction

The Random Forests (RF) [Breiman 2001] algorithm is an increasingly popular machine learning algorithm within statistical genetics. While many different algorithms have been successfully applied to genetic data, RF contains a combination of characteristics that make it well suited for genetic applications. First, it is well adapted for both prediction (the traditional domain of machine learning) and variable importance (VI) (the typical interest of statistical geneticists). Second, RF is fairly robust to the setting of the tuning parameters, making it, as Breiman referred to, an "off-the-shelf" algorithm readily accessible to novice users. It is also one of the few algorithms capable of handling thousands of observations and hundreds of thousands of predictors. Furthermore, the final derived model based on a combination of trees, represents a non-parametric model relating the predictors to the outcome. Trees are capable of capturing interaction and complex relationships in the data and make an ideal "base" learner for complex genetic problems. Finally, unlike many algorithms, RF is relatively straightforward to understand due to its non-parametric features.

Given these characteristics, it is not surprising that the number of statistical genetics articles published, since the introduction of the RF algorithm in 2001, has steadily increased (see Figure 1). Such papers usually take one of four forms. The first are direct applications of the RF algorithm to uncover genes associated with disease. Such analyses either independently use RF to identify associations (e.g. Goldstein et al. [2010]) or use RF in conjunction with other modeling approaches (e.g. Briggs et al. [2010a]). In these, RF is often used as a classification algorithm (the outcome is disease or not diseased), however there is some growing work in using time-to-event outcomes Ishwaran et al. [2008]. The second class of papers use RF as a prediction tool to predict disease state (e.g. Sun et al. [2007]). The third class of papers are methodological, illustrating improvements of the RF algorithm tailored for genetic applications, primarily focusing on the calculation of VI (e.g. Meng et al. [2009]) or providing an approach for variable selection (e.g. Díaz-Uriarte and Alvarez de Andrés [2006]). The final class of papers are those that compare RF to other algorithms (e.g. Statnikov et al. [2008]).

Based on the breadth of work both studying and applying RF, it is easy to recognize that it has desirable properties that allow for successful application in genetic epidemiology studies. Unfortunately, there is often an inconsistent and less than optimal use of RF in the literature. In many ways it is a quintessential "black box" algorithm with many moving parts which spits out a series of "answers." However, underlying it is a build-up of simple theory which helps in understanding how best to optimize it. The purpose of this paper is not to justify the use of RF or provide a prescription of how best to use it. Sun [2010] provides a good introduction

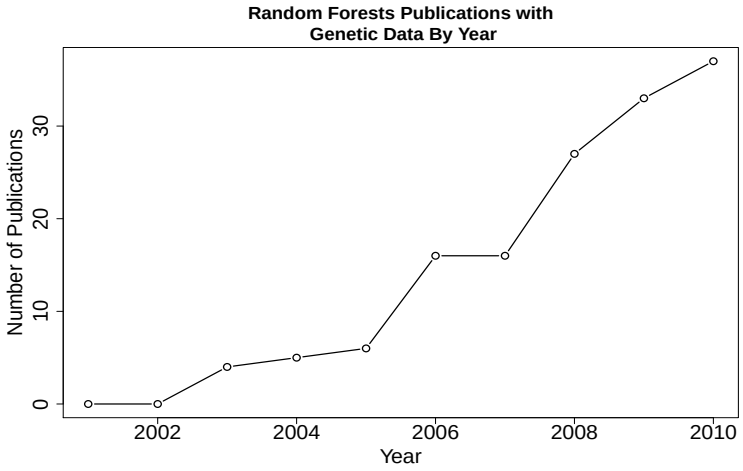

Figure 1: Articles listed in PubMed using the search terms (‘‘Random Forest’’ or ‘‘Random Forests’’) AND (Gene OR SNP). Over 125 articles since 2001 (articles from the Genetics Analysis Workshop omitted).

for the rationale behind using RF for genetic studies. Moreover, it is not our opinion that a “best” way to use RF (or any algorithm) exists since the best approach depends on the data problem at hand. Instead, the goal of this paper is to clearly lay out the theory behind RF to allow users to best determine how to optimize the algorithm for their specific data problem. Since the substantive theoretical work on RF occurred within the field of statistical learning and much of the applied work since has occurred within statistical genetics both literatures are represented.

This paper is written specifically to understand the use of RF for the type of classification problems encountered in large genetic association studies. While most of the discussion is generally applicable, some of it (particularly that of Section 3.1) will differ when the outcome is a continuous or survival outcome. Section 2 gives background on genetic association studies. Section 3 provides a discussion of classification and how its component break down into bias and variance. It also lays out the three main components of RF: trees, bagging, and randomization. Section 4 discusses various calculations of VI. The next section discusses how to use the RF algorithm focusing on tuning parameters and modifications that could be made to the input data relevant to genetic studies. The emphasis is on showing how these changes relate to changes in bias and variance. Other uses of RF and various implementations are briefly discussed in Section 5. Section 6 provides a comparison to some other algorithms to provide more context for RF. Some concluding thoughts are provided in section 7.

DOI: 10.2202/1544-6115.1691

2 Genetic Data

Typical applications of RF for genetic studies involves the use of case-control association studies. In these studies, individuals with and without a disease of interest are recruited. The outcome, Y, can be expressed as a binary variable making the task a classification problem. Since there is no easy means to control for confounding factors within the algorithm, it is important that such considerations are made in the study design to minimize confounding (e.g. matching cases and controls by age and gender). The primary form of confounding in genetic studies are gender and ethnicity. Generally, researchers have focused analyses to autosomal chromosomes thus avoiding potential gender bias that may exist when investigating the sex chromosomes. Confounding due to ethnicity (population stratification) is an important concern and it is essential to ensure that ethnic backgrounds between cases and controls are similar and that population outliers are removed beforehand. This can be accomplished using any of several freely available programs (e.g. EIGENSTRAT [Price et al. 2006]).

The predictor variables of interest are usually either units of DNA referred to as single nucleotide polymorphisms (SNPs) or gene expression measurements. In gene expression studies, each individual receives a continuous measurement indicating the degree of expression for up to tens of thousands genes, where the measure of interest is the relative differences in expression levels between cases and controls.

SNP studies are generally much larger. Where expression studies may have only 100 observations, SNP studies can have over 10,000 observations. Moreover, SNP studies may contain up to 2.5 million SNPs (or more), referred to as genome wide association (GWA) studies. SNPs are genomic structural variation that consists of a pair of bi-allelic nucleotides, where each allele can be represented as a 0-1 variable, where 1 represents the minor (less frequent) allele. The identification of SNPs for inclusion in GWA studies have largely been informed by the International HapMap Project and the 1000 Genomes Project which extensively describe and catalogue the common patterns of human DNA sequence variation [The International HapMap Consortium 2003; The 1000 Genomes Project Consortium 2010]. The standard dogma is that variation at these points, or points correlated with them (referred to as linkage disequilibrium [LD]) account for biochemical changes which lead towards variable phenotypes and disease.

GWA studies attempt to identify relatively common SNPs (minor allele frequency [MAF] $> 1-5\%$) in LD with causal susceptibility variants in a complex disease. GWA studies assume the common-disease-common-variant hypothesis; that common disease susceptibility is a result of the joint action of several common variants with relatively small to moderate effects, and that a significant propor-

tion of disease alleles is shared among unrelated affected individuals (Reich 2001). These studies are an attractive approach to investigating the genetic basis of complex diseases as they are hypothesis free and unconstrained by a priori assumptions regarding the disease's etiology. Successful application of GWA studies is dependent on: 1) sufficiently large study samples from clearly defined study populations capable of contributing relevant genetic information regarding the research question, 2) polymorphic SNPs that can be inexpensively and efficiently genotyped, and that capture extensive genetic variation across the whole genome, and 3) analytical approaches that are statistically robust, despite the dimensionality conflict (excessively large number of variables relative to number of observations), and can be employed to identify the genetic association in an unbiased fashion (conventionally, a single-point, one degree of freedom test of association, where significance is defined as $p < 5 \times 10^{-8}$) [McCarthy et al. 2008; Cantor et al. 2010].

A SNP that has a causal impact may have one of four potential effects outlined in Table 1. For data analysis, we can consider coding SNPs as an ordinal 0/1/2 variable where the value represents the number of minor alleles. Tree based algorithms are well suited for both expression and SNP analyses. As opposed to parametric methods which require specifying a causal model, trees allow for a general search of an optimal cut-point (see section 3.2), allowing the range of causal mechanisms to be easily uncovered by the tree structure. For example, a partition for a given SNP between 0 & 1/2 would represent a dominant effect.

Type	**Mechanism**	**Partition**
Additive	Each additional minor allele increases variation	0, 1, 2
Dominant	Presence of at least 1 minor allele increases variation	0, 1/2
Recessive	Two minor alleles needed for variation	0/1, 2
Heterosis	Heterozygote leads to variation	0/2, 1

Table 1: Different genetic effects

3 The Components of the Random Forests Algorithm

3.1 Bias - Variance Decomposition

One of the first steps in understanding a predictor is to see how its predictions contribute to bias and variance. We start with the setup, given an outcome y, input

DOI: 10.2202/1544-6115.1691

vector $\mathbf{x}$, and relationship $y = f(\mathbf{x}) + \varepsilon$, where $E[\varepsilon] = 0$ and $Var[\varepsilon] = \sigma_\varepsilon^2$. For a given training set T, the prediction is $\hat{f}(\mathbf{x}|T)$. The well known decomposition for prediction error (PE) under squared-error loss with a continuous outcome is:

$$E_T[y - \hat{f}(\mathbf{x}|T)]^2 = \underbrace{\sigma_\varepsilon^2}_{\text{Noise}} + \overbrace{[f(\mathbf{x}) - E_T\hat{f}(\mathbf{x}|T)]^2}^{\text{Bias}} + \underbrace{E_T[\hat{f}(\mathbf{x}|T) - E_T\hat{f}(\mathbf{x}|T)]^2}_{\text{Variance}} \quad (1)$$

Where the expectation is over random training sets. The first term is the variance of the outcome y and is referred to as the noise. This represents the irreducible error. The next two terms represent the reducible error. The first of these is the bias. We can think of the bias as the systematic difference between the prediction and the target. The final term is the variance. It is the measure of randomness of the prediction. It is important to note that the variance is independent of the true outcome y and the true function $f(\mathbf{x})$.

In classification with a 0-1 outcome we are trying to minimize $P(\hat{f}(\mathbf{x}) \neq y), y \in \{0,1\}$ This is usually done under miss-classification loss

$$l(y, \hat{f}(\mathbf{x})) = \begin{cases} 1 & \text{if } y \neq \hat{f}(\mathbf{x}), \\ 0 & \text{if } y = \hat{f}(\mathbf{x}). \end{cases} \quad (2)$$

In the mid 1990s multiple authors attempted to define a decomposition for 0-1 loss [Dietterich and Kong 1995; Kohavi and Wolpert 1996; Breiman 1996b; Tibshirani 1996]. Most of the effort centered around trying to find a decomposition that was additive in the components of noise, bias, and variance, as in Equation (1). Each author proposed a slightly different decomposition, depending on which properties they hoped to satisfy.

Unfortunately a thought exercise shows that bias and variance are not additive when the goal is classification. First, note that if a classifier predicts the correct class, $P(\hat{f}(\mathbf{x}|T) = y) \geq .5$, when the true class is class 1, it is unbiased at $\mathbf{x}$. If we have an unbiased classifier, we would desire for the classifier to also have low variance. However, if the classifier is poor, $P(\hat{f}(\mathbf{x}|T) = y) < .5$, we say it is biased. In this scenario we would actually desire the classifier to have high variance because we want to increase the chance that the classification "flips." In this sense, to minimize PE, we see that for an unbiased (good) classifier we want low variance, but for a biased (poor) classifier we want high variance.

Friedman [1997] recognized this interaction between bias and variance. After averaging over all training sets, he decomposes the relationship as,

$$P(\hat{f}(\mathbf{X}) \neq y) = |2f(\mathbf{X}) - 1|P(\hat{f}(\mathbf{X}) \neq f^*(\mathbf{X})) + P(f^*(\mathbf{X}) \neq y) \quad (3)$$

where $f^*(\mathbf{x})$ is the Bayes classifier, and $\mathbf{X}$ designates over all inputs, x. Friedman referred to $P(\hat{f}(\mathbf{X}) \neq f^*(\mathbf{X}))$ as a decision boundary error. Making the simplifying assumption that $P(\hat{f}(\mathbf{X}))$ is normal, he showed this boundary could be represented by:

$$P(\hat{f}(\mathbf{X}) \neq f^*(\mathbf{X})) = \Phi\left[sign(1/2 - f(\mathbf{X}))\frac{E\hat{f}(\mathbf{X}) - 1/2}{\sqrt{var(\hat{f}(\mathbf{X}))}}\right] \quad (4)$$

The "boundary bias" is then represented by $sign(1/2 - f(\mathbf{X})(E\hat{f}(\mathbf{X}) - 1/2)$. It is clear that when predicting to the correct class, "boundary bias" is negative, and PE decreases as $[E\hat{f}(\mathbf{X}) - 1/2]$ increases. Furthermore, it is evident that decreasing the variance of the predictor is only beneficial when the classifier is on the correct side of the boundary. In this way we see the strong multiplicative interaction between bias and variance.

Gareth [2003] followed this up by suggesting a unified bias-variance decomposition, applicable to all symmetric loss functions. Specifically, he recognized that there is both bias, and the effect due to bias. Similarly there is variance and the effect due to variance. He showed that under squared-error loss these are equal, under other losses they are not.

In genetic studies, we are often more interested in variable importance than prediction (though prediction is growing in interest). It is then natural to ask why concern ourselves with these issues. The concern arises when it comes to tuning the algorithm. In the classification setting, the interest may not be in predicting the class outcome, but instead the underlying probability. Friedman [1997] showed that in the probability estimation setting the bias-variance again becomes additive (assuming squared-error loss). In an application to K-Nearest Neighbors (K-NN), he demonstrated that the implication of this is that different tuning parameters will be favored depending on the task at hand.

It is also possible to show that the prediction error is directly related to the quality of the variable importance measures. Figure 2 illustrates this point. The RF algorithm was trained on a data set containing 117 predictors. Of these predictors, 17 were associated with the binary outcome in a basic additive logistic model. The algorithm was run, and the top 17 variables were examined based on RF VI (see Section 4). The experiment was repeated, however, *noise* was injected by randomly "flipping" a certain percentage of the outcomes (a similar simulation was performed by Breiman in the original RF paper [Breiman 2001]), leading to an increase of PE,

DOI: 10.2202/1544-6115.1691

as measured by the Out-of-Bag (OOB) error-rate (see Section 5.1.1). As shown, as PE increases, the number of "true" associations among the top results decreases. This shows that appropriate minimization of PE is just as important for VI as it is for prediction.

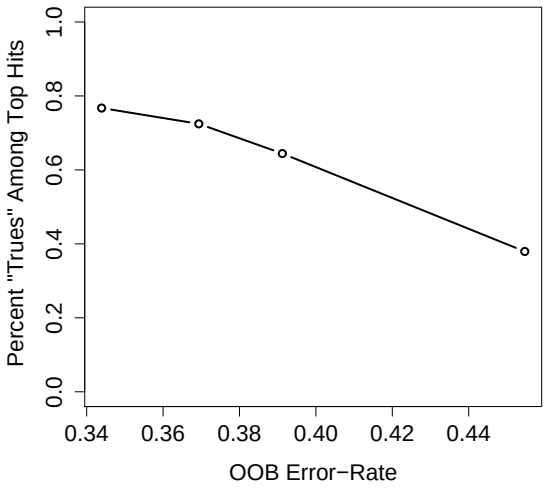

Figure 2: The relationship between quality of variable importance and prediction error. As PE increases the quality of VI rankings decrease.

3.2 CART

Underlying Random Forests is the Classification and Regression Tree (CART) algorithm [Breiman et al. 1984]. The CART algorithm recursively searches for a binary split that partitions the data in such a way that minimizes a splitting criterion. This is referred to as a "greedy" search. After a stopping criterion is met, the final splits partition the predictor space into hyper-rectangles. These regions are referred to as leaves or terminal nodes of the tree.

The variable at the top of the tree represents the "strongest" splitting variables (see Figure 3). Subsequent variables are conditional on those variables above it. Trees are an appealing base learner because they present a straightforward means to represent complex relationships, particularly those that are present in genetic data. The tree structure represents a conditional model making it suited for finding

interactions and higher order effects. Furthermore, since trees do not assume linearity in effects, instead performing binary splits, it is ideally suited for discovering *recessive* and *dominant* genetic effects. The main type of effect ill suited for trees are in *additive* effects since this requires consecutive splits of the same variable.

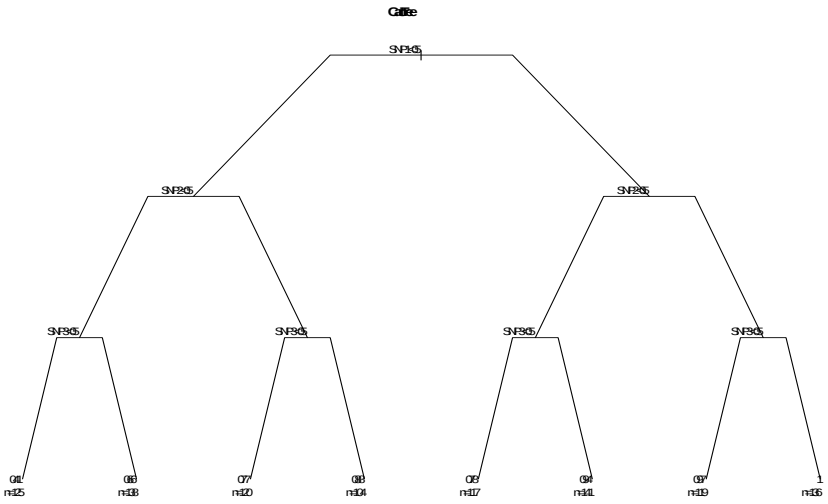

Figure 3: A CART tree representing the hierarchy of effects. An additive model was simulated where the effect of SNP1 $>$ SNP2 $>$ SNP3. This hierarchy is reflected in the ordering of the tree.

In the case of classification the splitting criterion, $Q_m(T)$, is typically the gini-index (though other convex losses can be used). For a node m, in region R_m, with N_m observations, we define

$$Q_m(T) \equiv \hat{p}_{mk} = \frac{1}{N_m} \sum_{i:x_i \in R_m} I(y_i = k)$$

where k is an outcome class. The gini index is then:

$$\begin{aligned} GI &= \sum_{k=1}^{K} \hat{p}_{mk}(1 - \hat{p}_{mk}) \\ &= 2p(1-p) \text{ when K = 2} \end{aligned} \tag{5}$$

This process continues until all of the leaves contain only members of one class. One nice feature of the gini criterion is that it is prefers such pure nodes. Other losses, notably misclassification loss, are not necessarily minimized by a pure node.

DOI: 10.2202/1544-6115.1691

Since a fully grown tree, T_0, will have high variance, (changes in the training data will lead to different tree structures), trees are typically pruned, by finding the sub-tree $T_\alpha \subseteq T_0$ which minimizes the criterion:

$$C_\alpha(T) = \sum_{m=1}^{|T|} N_m Q_m(T) + \alpha|T| \tag{6}$$

where $|T|$ is the number of terminal nodes in the tree and α is a tuning parameter chosen typically by cross-validation. However, in Random Forests, the trees are not pruned but kept at their maximal depth. This results in each tree having low bias, but high variance. This variance is alleviated by bagging (see next section).

To illustrate the the value of trees for genetic studies, a simulation was performed emulating a GWA study.[1] The number of true effects found among the top results by RF and marginal p-values (an allelic chi-square test) are compared in Figure 4. RF is more adapted to finding the non-linear dominant and recessive effects, while marginal testing is better at finding additive effects.

3.3 Bagging

Brieman proposed the ensemble process *Bagging* (**B**ootstrap **Agg**regat**ing**) as a solution to the instability observed in classifiers such as CART trees [Breiman 1996a]. In ensemble methods such as bagging, the algorithm used (i.e. CART) is referred to as the "base learner." Bagging is a straightforward procedure where successive bootstrap samples of the data are selected, (X^b, Y^b), and a prediction, $\hat{f}(x^b)$, is derived from each of these samples. The final prediction, $\hat{f}_{bag}(\mathbf{x}|T)$ is determined by either averaging each of the predictions, $\frac{1}{B}\sum_{b=1}^{B}\hat{f}^b(\mathbf{x})$ (for a continuous outcome), or taking a majority vote, $argmax_k\hat{f}_{bag}(\mathbf{x})$ (for classification). To estimate the $P(f(\mathbf{x}) = k)$, the intuitive approach is to average the probability estimate of each of the base learners (in CART this would be the terminal nodes). However the better approach to estimate this quantity is to divide the number of bagged samples that vote for class k by the total number of bagged samples (see Hastie et al. [2009] pg.286 for discussion).

[1] For this simulation a 71 parameter logistic model was used to simulate an outcome. The 71 "causal SNPs" consisted of additive, dominant, recessive and interaction effects with varying minor allele frequencies and correlated variables. Real GWA data was used that had been pruned to an LD R^2 of 90% (p = 163,231) as noise variables. An in-house written version of RF was used. $6,000$ trees were grown with different `mtry` values. Top results were compared to those found in an allelic chi-square test.

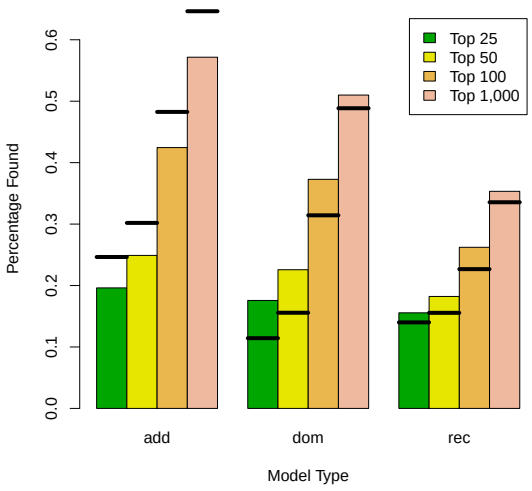

Figure 4: Comparison of type of effects found by RF and marginal allelic p-values. The black line in each graph indicates the percentage of the "true" effects found among the top results by looking at just the marginal p-values. RF VI is better at finding dominant and recessive (i.e. non-linear) effects while marginal p-values is more suited to finding additive effects.

The motivation behind bagging is to simulate having multiple training sets. If all training sets, T, were used, then there would be no variance in the final prediction. Bagging then works by reducing the variance of the final predictor. Bühlmann and Yu [2002] and Friedman and Hall [2007] each showed that bagging works via smoothing out first order and higher order variance terms. With respect to bias, since the distribution of $(X^b, Y^b) \sim (X, Y)$, the bias of $\hat{f}_{bag}(\mathbf{x})$ equals the bias of $\hat{f}(\mathbf{x})$, so there is no (asymptotic) increase in bias induced by bagging, though there can be in finite samples.

Another perspective on bagging is that manipulation of the input space is able to increase the search space for an optimal solution [Dietterich 2000a]. This process works only with unstable predictors [Breiman 1996c], defined as as a predictor where a small change in the data can lead to large changes in the prediction. Conversely, bagging stable predictors will lead to worse outcomes. Procedures like CART are unstable while procedures such as regression are stable.

In the case of classification, Breiman [1996a] argued that bagging is effective in the case of order-correct predictors which he defined as a classifier that

DOI: 10.2202/1544-6115.1691

places the greatest probability on the true class for a given input **x**, i.e. is unbiased at **x**. Similar to the previous discussion of the bias-variance relationship for classification, Breiman noted that for such order-correct classifiers bagging can be very useful, but for ones that are not, bagging can actually be harmful. This is because for good classifiers we want to decrease the variance (to reduce overall PE) but for bad classifiers we want to increase the variance (to reduce the overall PE).

3.3.1 Bagging Type

Bühlmann and Yu [2002] and Friedman and Hall [2007] also both showed that $m < n$ sampling without replacement, where $m = n/2$ is just as effective as bagging with replacement, and computationally more efficient. Bühlmann and Yu referred to this as *subagging* (**sub**sample **agg**regat**ing**).

Dietterich [2000a] notes that large datasets don't see the same benefits from bagging as do smaller ones because each bootstrap sample is more similar to each other than with a smaller data set. Subagging serves three benefits. First would be computational - since fewer observations are used in growing the trees. The second is through a reduction in tree correlation - the trees would be more different from each other. The third is in a reduction of tree size. This third component decreases the degrees of freedom of the final model.

3.3.2 Out-Of-Bag Error-rate

One of the appeals of bagging is that it presents a computationally efficient means to estimate the generalized error (GE), the PE of on an independent test set T'. The best way to estimate GE is on an independent validation set. In lieu of one, different analytic (e.g. AIC, BIC) and computational (e.g. Cross-Validation [CV]) approaches have been developed.

For bagged learners, analytic approaches are not feasible and CV is computationally very expensive. However in each iteration of bagging, approximately 37% of the sample is not part of the bootstrap sample.

$$\begin{aligned} P(\text{observation } i \in \text{bootstrap sample } b) &= 1 - \left(1 - \frac{1}{N}\right)^N \\ &\approx 1 - e^{-1} \\ &= 0.632 \end{aligned}$$

Breiman [1996d] showed that this Out-Of-Bag (OOB) sample can be used as a test set to get a measure of error, referred to as the out-of-bag error rate (OOB-ER). Over the entire bagging run, this error can be aggregated for each input vector **x**. This can provide a more stable estimate of GE than typical V-fold CV [Wolpert and Macready 1999]. If we define C^{-i} as the set of indices not in bootstrap sample b, and $|C^{-i}|$ as the number of such samples, the OOB estimate becomes:

$$\widehat{GE}_{OOB} = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{|C^{-i}|}\sum_{b\in C^{-i}} L(Y_i, \hat{f}^b(x_i)) \tag{7}$$

Since each bootstrap sample will have a sample size of about $.632N$, this estimate will behave similar to 2-fold CV.

In a two-class problem the expected OOB-ER under the null (i.e. no improved prediction) is equal to the probability of the minor class. For example if the classes are equal the expected OOB-ER is 50%, however if 90% of the sample is from one class the expected OOB-ER would be 10%. This can be shifted back to 50% via weighting within the tree growing process and often is in RF (see Section 5.1.5).

Of particular interest, is that the OOB-ER provides a convenient means of choosing tuning parameters in RF. As will be discussed, RF involves the choice of multiple tuning parameters, and these can be chosen by determining the settings that minimize the OOB-ER.

3.4 Randomization

A final method for improving ensemble learners is by injecting randomization into the base learner. Many different procedures have been explored for this [Dietterich 2000a]. For example, RF injects randomness into the tree growing process by only searching over a subset of variables when searching for the optimal split. Other randomized tree algorithms have been proposed using different forms of randomization (see Cutler [1999]).

Like bagging, Dietterich showed that this randomization is able to expand the search space and alleviate what he termed the "statistical" burden. In another study, it was shown that injecting randomization can be more effective than bagging for large datasets [Dietterich 2000b].

As noted in Hastie et al. [2009] the variance reduction induced by bagging is limited by the correlation between the trees, since the trees are not independent, only identically distributed. If we denote the variance of each tree's prediction as

DOI: 10.2202/1544-6115.1691

σ^2 and the correlation between the predictions as ρ the variance of the average is:

$$\rho\sigma^2 + \frac{1-\rho}{B}\sigma^2 \tag{8}$$

As the number of bootstrap iterations, B, increases, the second term goes to 0, and we are left with the correlation between trees, ultimately limiting the benefits of the bagging process. As dataset size increases, the correlation between bagged samples increases, decreasing the effect of bagging. Injecting randomization into the tree growing process serves to further de-correlate the trees, further reducing the variance.

3.5 The Random Forests Algorithm

At this point we can consider the RF algorithm, proposed by Leo Breiman in 2001 [Breiman 2001]. At its core, it is bagged CART trees, with injected randomization (see Figure 5).

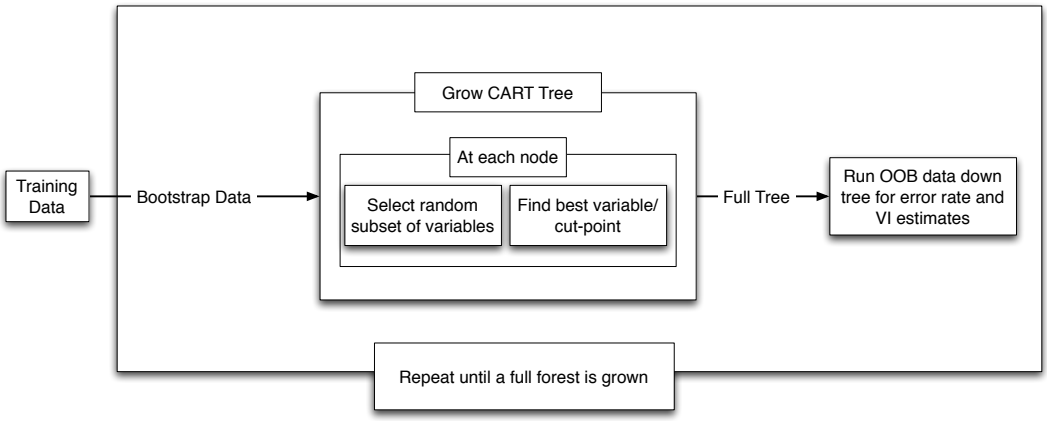

Figure 5: The RF algorithm begins by selecting a bootstrap sample of the data. A random subset of the variables is selected and searched over to find the optimal split. This is repeated at each node until an unpruned CART tree is formed. The data not part of the bootstrap sample is run down the tree to derive the error rate and measures of VI. This is repeated until a full forest is grown.

The first alteration is that instead of pruning the CART trees, they are grown to maximal depth. These fully grown trees will be fairly unbiased but will be highly variable (recall CART trees are pruned to reduce the variance). This variance is reduced via bagging and randomization.

In RF, the randomization comes in the tree growing process. Before each split, a subset, $m \leq p$, of the predictor variables are selected to search over. The choice of m, denoted `mtry`, is the primary tuning parameter. The smaller `mtry` the less correlation between trees and the greater the potential variance reduction via bagging is possible. However, smaller `mtry` will also lead to more biased trees, hence resulting again in the classic bias-variance trade-off.

RF reduces PE only through a reduction of variance, as the bias stays the same (or gets a bit a worse). Breiman shows that unlike other methods (notably Boosting), RF does not over-fit as the number of trees increases, i.e. the observed PE approaches the expected PE. However, it is possible, particularly with noisy data, for the model itself to be too rich (i.e. over-fit) and result in a poorer predictor.

4 Variable Importance

When applying RF for classification there are two primary forms of Variable Importance: permutation importance & gini importance.

4.1 Permutation Importance

The permutation importance (pVI) is the increase in misclassification for OOB person i after variable j has been permuted in tree k. If we consider the quantities:

- s_{ijk} = number of trees that split on variable j and misclassify observation i
- r_{ijk} = number of trees that do not split on variable j and misclassify observation i
- ps_{ijk} = number of trees that split on variable j and misclassify observation i when variable j is permuted
- pr_{ijk} = number of trees that do not split on variable j and misclassify observation i when variable j is permute

DOI: 10.2202/1544-6115.1691

We can represent it as:

$$\begin{aligned} pVI_{ijk} &= (ps_{ijk} + pr_{ijk}) - (s_{ijk} + r_{ijk}) \\ &= ps_{ijk} - s_{ijk} \text{ since } pr_{ijk} = r_{ijk} \end{aligned}$$

and we can calculate:

$$\begin{aligned} pVI_{ij} &= \frac{1}{ntree} \sum_{k=i}^{ntree} ps_{ijk} - s_{ijk} \\ pVI_{jk} &= \frac{1}{np} \sum_{i=i}^{np} ps_{ijk} - s_{ijk} \\ pVI_{j} &= \frac{1}{np \times ntree} \sum_{i=i}^{np} \sum_{k=i}^{ntree} ps_{ijk} - s_{ijk} \end{aligned} \tag{9}$$

where np and $ntree$ are the number of people and the number of trees respectively.

The three quantities in (9) represent respectively the importance of variable j for person i, the importance of variable j for tree k and the overall importance of variable j. Each representation will have different utility depending on the question of interest.

pVI has some nice properties. Since it is calculated off of the OOB sample, it can be viewed as the predictive quality of that variable. A variable with no importance would be expected to have $E(pVI) = 0$ since permutation should neither increase nor decrease misclassification. There is also a notion of a population level effect of the variable importance since the probability of being permuted to a different value is determined by the observed population. It is also applicable for any outcome or predictor type.

4.1.1 Correcting Permutation Importance

An important consideration with applying RF to genetic data is the large degree of LD (correlation) among SNPs. There are a couple of ways to formulate the problem in calculating VI induced by correlation. Being a "greedy" algorithm, RF searches over all variables. In calculations of VI, this creates a smoothing and shrinkage of all VI measures - in an analogous way to Ridge regression (see Section 6). This creates problems for correlated variables as the relative importance is diminished. Another formulation is that since VI is calculated from the number of trees for which a variable appears two SNPs that are in perfect LD will appear in trees about half as often as each individual one may appear by itself, effectively lowering the VI of each SNP. While this does not present a problem for prediction, it can skew the VI rankings.

Genuer et al. [2008] examined the impact of correlated variables, and found that as the number of variables correlated with a true causal one increased, the variability of the true causal one increased and its average importance decreased. Similar effects were noted by other authors, notably Strobl et al. [2007].

Meng et al. [2009] proposed a correction for this. Since pVI is calculated by dividing pVI_{ij} by the total number of trees in the forest, the authors suggested dividing by the total number of trees of which variable j is a member. This has the appeal that two perfectly correlated variables will no longer "take away" from each other. In practice, with large p this leads to highly unstable pVI measures and works best with less sparse solutions or smaller p where all variables have a chance to be brought into the model.

4.2 Gini Importance

The gini importance (gVI) is the second primary form of RF VI. Unlike pVI, gVI is only applicable in the case of classification. The gini index (GI) is the criterion used when growing the trees in RF for classification. Recalling Equation (5), for binary classification,

$$GI = 2p(1-p)$$

where p is the proportion in the second class. The split which minimizes GI is the preferred split. If we index the node for a given tree by n, we can then define:

$$\begin{aligned} gVI_{jkn} &= (GI_{parent} - GI_{\text{daughter left}} + GI_{\text{daughter right}})np_{kn} \\ gVI_{jk} &= \sum_{n_j \in Tree_k}^{N} gVI_{jkn} \\ &\text{(summing over the nodes containing variable } j \text{ in tree } k) \\ gVI_j &= \frac{1}{ntree}\sum_{k=1}^{ntree} gVI_{jk} \end{aligned} \tag{10}$$

gVI_{jk} directly measures the importance of variable j to tree k. The higher the value the better the variable was in splitting the data. In this sense it is very different from pVI. There is no notion of out of sample testing. Instead gVI_{jkn} can be thought of as a χ^2 test, conditional on what has already occurred in the tree (for the root node it is conditional on nothing).

Another property is that $gVI_j \geq 0$ with equality if variable j does not appear in any tree. Like pVI it will have trouble with correlated variables but can also be corrected by weighting. Since gVI is calculated based on the in-sample data it does

DOI: 10.2202/1544-6115.1691

not have a population level interpretation as pVI. Instead gVI only considers the relationship between the variable and the model.

pVI is the more commonly used form of VI. However, some intuition shows that gVI can be a preferential VI measure when the predictive quality of the trees is low (i.e. OOB-ER $\approx 50\%$). Since pVI is calculated based on the increase of misclassification after permuting variable j, if the baseline misclassification rate is already relatively high, there is little chance for permutation to make prediction worse. This will lead to an uniformly low pVI. Conversely, since gVI is calculated relative to the grown tree it does not suffer this problem. It is easy to show this via simulation. However, since variables have to be in the tree, there will always be variables with high gVI and it is questionable how "important" a variable is that doesn't improve prediction. Moreover, it is much more challenging to consider distributional properties for gVI.

4.3 Other Variable Importance Measures

One of the appeals of RF is that advanced users can tailor the algorithm to calculate different VI measures depending on the question of interest. Lee et al. [2008] developed a VI measure relevant for looking at linkage data. Bureau et al. [2005] created a modification of pVI to look at the joint effects for pairs of SNPs. Jiang et al. [2009] proposed a sliding window VI measure for examining interactions.

Many of the modification to VI involve correcting for the "bias" incurred by the presence of correlated predictors. Strobl et al. [2008] suggested using a conditional permutation scheme to calculate VI. The variables correlated with the variable of interest are empirically determined, and then the partitions in the individual tree are utilized to permute the variable of interest within blocks. While effective when p is small, this has drawback of creating a VI measure that is more computational and not uniform across trees. Wang et al. [2010] developed a different VI measure that also takes into account conditional effects. The authors proposed a permutation based test to associate a p-value with their VI measure. While useful, many of these measures are ad hoc and as with pVI and gVI it is important to determine whether statistical properties exist since relying on permutation tests can be inefficient.

4.4 Determining Important Variables

The RF VI measures are fairly successfully in rank ordering associated predictors. However, in genetic studies the goal is often to determine which variables are worthy of future follow-up. Ideally, statistical properties for the VI measures would

exist to determine when the observed value differed from an expected value. Some work has been performed in this area but to this point no formal approach has been adopted.

Another challenge is that RF will incorporate most predictors into the final collection of trees. Other algorithms, notably LASSO (see Section 6), will incorporate fewer variables, referred to as a *sparse* model, making it easier to drop seemingly unimportant or redundant variables. In genetics applications, where many SNPs or genes are not associated with the disease and are simply noise, this lack of sparsity makes variable selection more challenging.

Due to the lack of sparsity and defined statistical properties, it is not possible to divide the SNPs or genes into disjoint sets of "associated" and "not-associated." This has led to a range of ad-hoc procedures. Díaz-Uriarte and Alvarez de Andrés [2006] suggested removing the bottom 10% and re-running until prediction decreased. Rodin et al. [2009] devised a method for selecting variables based on specification of optimal model size. Goldstein et al. [2010] examined the scree plots of the VI measures and used the "elbow" as the cut-off. However, these are unsatisfactory solutions and ideally some objective cut-off could be determined.

5 Applying Random Forests

5.1 Tuning Parameters

Running RF involves the choice of three primary tuning parameters: `mtry`, the number of trees (`ntree`), and tree size (`nsplit`, `maxnodes`). In the case of classification, class weights also can be varied. While RF is relatively robust to the settings of tuning parameters (fine changes will generally give similar results), each tuning parameters will contribute differently to the bias and variance of predictions and impact the quality of the final solution.

5.1.1 Using the OOB-ER

The optimal values of most of these tuning parameters are dataset dependent. By understanding how these tuning parameters contribute to bias and variance, it is possible to a priori speculate as to the optimal value, however they ultimately need to be empirically determined. The OOB-ER provides an unbiased estimate of the generalized error. Minimizing this error allows one to select the optimal tuning parameters to generate the best predictive model. However, when one begins augmenting the dataset (e.g. removing unimportant variables) the OOB-ER is no longer

DOI: 10.2202/1544-6115.1691

an unbiased estimate of the generalized error, though its minimization can still be used for tuning parameter selection [Svetnik et al. 2004].

Theoretical work has shown that the prediction error can be tied directly to the strength of association of the set of predictors [Goldstein et al. 2011] and our own internal testing has shown that as the OOB-ER improves, the quality of the VI rankings improve (see Figure 2). With this in mind, VI should be interpreted in conjunction with the OOB-ER. If the OOB-ER, is close to the null value it is likely that none of the predictors are associated with the outcome, regardless of the VI scores.

5.1.2 `mtry`

In each step of the tree growing process a different subset of variables is selected to search over in order to find the optimal split. When the size of this subset, `mtry`, is small, the trees will have lower correlation leading to a greater potential for variance reduction (see Equation (8)). As `mtry` increases, the variance reducing effect of the randomization decreases. When `mtry` is equal to the number of predictors, p, RF reduces to bagging.

`mtry` is the primary tuning parameter, and has its greatest impact on the complexity of the final model. Larger values of `mtry` lead to fewer variables brought into the tree, resulting in *sparse* solution (see Figure 6). Since CART is a greedy algorithm, the higher the `mtry`, the faster it will converge on the optimal splits, creating smaller and more efficient trees. When data are noisy, this is desirable since unrelated predictors will be ignored. However, if most data is related to the outcome this can potentially ignore important but weak predictors. In this way `mtry` can be thought of as controlling the degrees of freedom (df) of the model, with the higher `mtry` the fewer df used.

In genetic studies, we expect that the vast majority of the input variables are simply noise. Therefore, if `mtry` is too small the chance of selecting an important variable to search over at a given node will be small. This will add additional noise into the trees counteracting the variance reducing potential of decorrellated trees.

The default recommended `mtry` is often $\sqrt{p}$. Díaz-Uriarte and Alvarez de Andrés [2006] examined `mtry` values ranging from $\sqrt{p}$ to $13\sqrt{p}$ and generally found that the larger the `mtry` the more reliable the VI measure. Genuer et al. [2008] noted that `mtry` is more important for VI calculation than for prediction and that with sparse data, `mtry`$= p$ leads to greatest stability. Goldstein et al. [2010], working with a large SNP dataset ($p > 330,000$) found that `mtry` values much larger than $\sqrt{p}$ were needed. In simulation work, the OOB error rate and VI measures

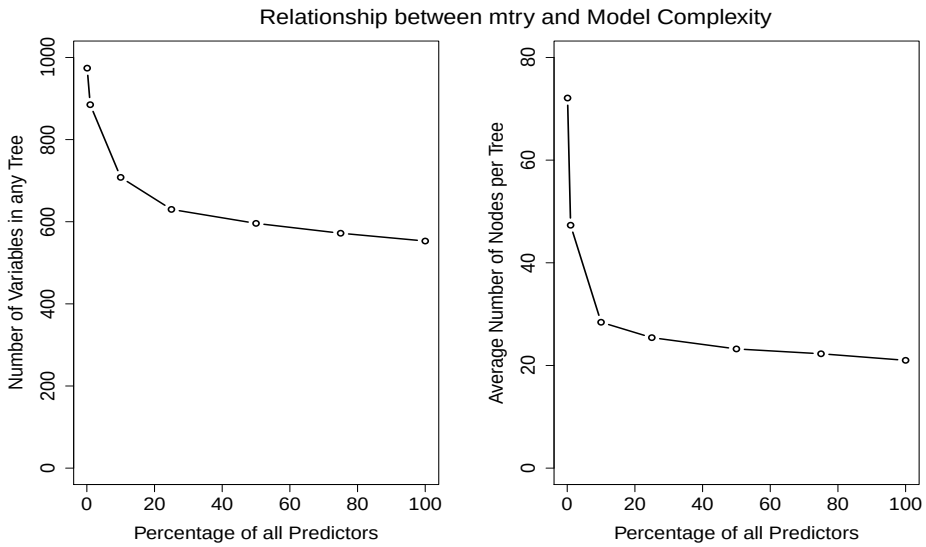

Figure 6: A RF with 100 trees and varying `mtrys` was grown. The data contains 1,000 predictors and 100 observations. Ten of the variables are associated on a logistic additive scale. As `mtry` increases the model complexity decreases as smaller trees are grown and fewer variables are brought into the final model.

were fairly similar with `mtry` values of $.1p$ and p, indicating that the setting is fairly robust to a sensible choice.

Ultimately, a default `mtry` value cannot be prescribed, as the optimal choice will depend on the data problem. Generally it is advisable to perform a coarse search of different values to see how results are impacted.

5.1.3 Number or Trees

Another important consideration is how many trees to grow, `ntree`. Unlike with `mtry` and the other tuning parameters, there is clear "best" `ntree`, with the larger the value the better. The main limitation to increasing the number of trees is the extra computation required and the diminishing returns with larger values.

Stronger predictors will lead to quicker convergence. Glaser et al. [2007] using a much smaller data set (20 SNPs), grew forests with up to 5000 trees, and found that after 400 trees, the OOB-ER was stable. Díaz-Uriarte and Alvarez de Andrés examined forest sizes ranging from 1000 to 40,000 and found no differences in error-rates. While for prediction purposes, fewer trees are necessary and the OOB

DOI: 10.2202/1544-6115.1691

error rate will generally converge rapidly, Genuer et al., noted that for variable importance more trees will generally lead to refinement and stability in variable importance.

If the only concern is prediction, the OOB error-rate can be used as the only guide to determine `ntree`. For VI this is not the case as larger forests will often be necessary. Path plots (Figure 7) provide one means of examining whether VI measures have converged, by showing the cumulative VI score as the forest grows. Using the same data from Figure 6, forests with up to 5,000 trees were grown with `mtry` = 100 (10%p). The VI measures stabilized after about 2,000 trees.

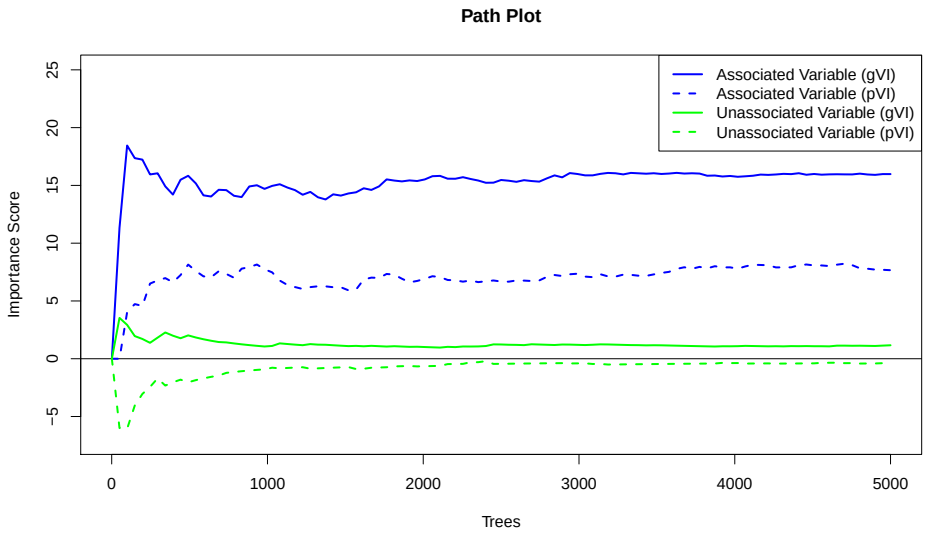

Figure 7: Path plots for two variables (one associated with the outcome and one not). The associated variable is simulated on the log additive scale. For the associated variable, after about 1000 trees gVI has stabilized, while pVI stabilizes at about 2000 trees. The unassociated variable stabilizes sooner.

5.1.4 Controlling Tree Size

Controlling tree size is not an often discussed tuning parameter but can be useful for genetic data. The tree size can be controlled explicitly by limiting the number of splits (`maxnodes`) resulting in what are referred to as *stumps* (as in boosting, see Section 6). The size can also be controlled adaptively by telling the tree growing algorithm when it should stop splitting the node leaves of the tree (`nsplit`). In RF

trees are grown to maximal depth (i.e. when nodes are pure of class) however one can choose to stop the splitting based on the number of people in the node.

Theoretically, limiting tree size is unnecessary. From a bias-variance perspective, the purpose of pruning is to increase the stability (i.e. lower the variance) at the cost of increased bias. The increased variance of unpruned trees is due to their tendency to over-fit the data. However, the bagging process specifically aims to reduce variance (with no cost to bias) and avoid over-fitting making pruning superfluous.

Practically, though, limiting tree size can prove beneficial. For one it will speed of computation. Moreover, particularly, with datasets with many noisy predictors, it will limit the number of superfluous splits. This will result in smaller trees that (hopefully) contain only the important predictors.

In practice, these tuning parameters have not received enough attention. The primary evaluation of it was by Segal [2004], who looked at `nsplit`. As with `mtry` the conclusion was that there was often an optimal `nsplit` though growing trees to maximal depth did not lead to over-fitting.

5.1.5 Class Weights

In the case of classification, the classes can be differentially weighted. By default most implementations of RF weight the classes so they are balanced. This ensures, in the case of grossly uneven classes (i.e. 90% vs 10%), that a degenerative solution is not found (i.e. one that predicts all observations to the larger class with a OOB-ER of 10%). However, if the interest is in determining which genes are important for cases, the class weights could be altered towards the diseased class. To our knowledge this issue has not been examined and is worthy of further exploration.

5.2 Modifying the Data

The final solution can also be affected by modifying the data that is input. It is important to note, that once the data is modified the OOB-ER no longer represents an unbiased estimate of GE [Svetnik et al. 2004].

5.2.1 Correlation

There are two approaches for dealing with this correlated data: computationally and in the set-up of the data. Computational approaches are discussed in Section 4.1.1. A second approach, applicable with SNP data, is to pre-process the data based on LD structure. This was the approach taken in Goldstein et al. [2010]. Different

DOI: 10.2202/1544-6115.1691

levels of LD pruning were used. It was found that pruning to an LD level of 90% (R^2) did not lead to a degradation PE and allowed new results to be detected, while improving computational speed.

5.2.2 Removing Unimportant Variables and Important Ones

As mentioned, the sparsity of the final model is a function of both `mtry` and `ntree`. Such a sparse model will result in VI values equal to 0. It can be beneficial to remove these sparse results as they likely represent noise and will make finding the optimal solution more challenging. Díaz-Uriarte and Alvarez de Andrés outlined a strategy of sequentially removing genes by dropping the bottom 20% or 50% performing successive runs until there was a noticeable increase in PE.

Not only can we consider removing unimportant variables, we can also consider removing overly strong results. Since the final VI value is dependent on where a variable lies in the tree, a strong marker or set of strong markers, could over shadow weaker yet important effects by pushing them down the tree. By removing highly influential variables, other variables have the opportunity to rise to the top of the tree and have a stronger VI score. Goldstein et al. [2010] showed that after removing Chr 6p, a region known to be highly associated with the disease being studied, new SNPs were found that would not have been detected otherwise.

5.3 Interaction Detection

As GWAs have detected most "easy" to find effects a greater emphasis has been placed on the detection of interaction or epistatic effects. Many techniques and computational tools have been developed, each with different strengths and weaknesses [Cordell 2009]. Due to the conditional nature of trees they are well suited for modeling non-linear effects such as interactions. If a variable serves as a split on one side of a tree but not on the other, that is an indication of an interaction between that variable and one above it (see Figure 8).

Multiple studies have compared RF to other parametric and computational methods, showing the variable importance scores to be more powerful and stable than other current approaches ([García-Margariños et al. 2009; Lunetta et al. 2004; McKinney et al. 2006; Nicodemus et al. 2007]. In addition to gene x gene interaction RF has also been used to successfully detect gene x environment interactions [Maenner et al. 2009].

One of the primary limitations is that while RF can identify SNPs that are potentially epistatic, the typical output does not give the user any indication of which SNPs may be interacting and which they may be interacting with. While

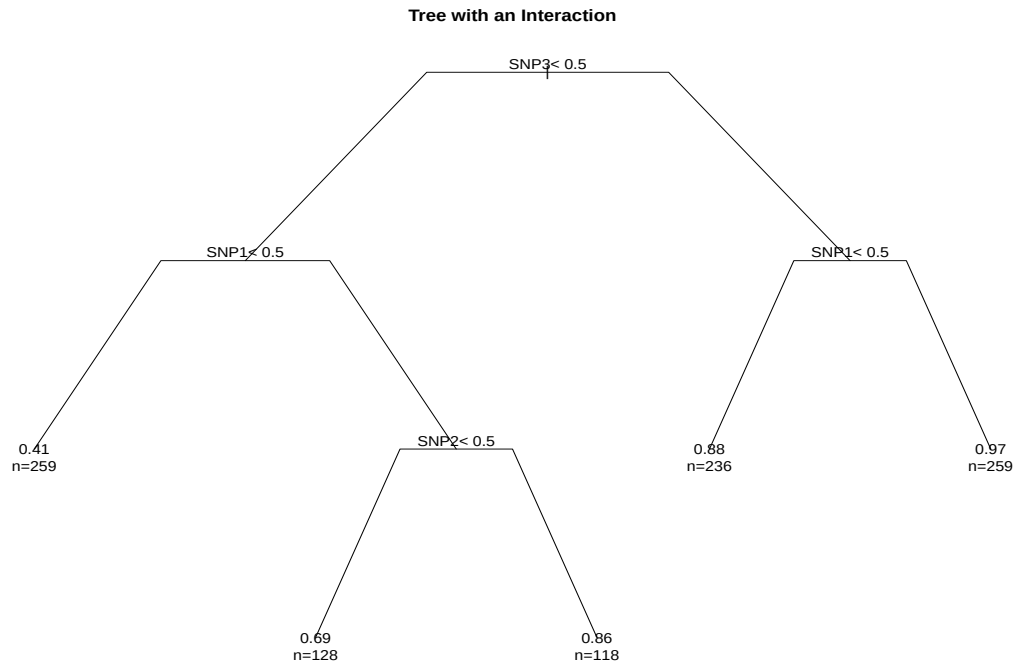

Figure 8: Illustration of a tree with an interaction between SNP1 and SNP2. Both SNP3 and SNP1 have independent main effects.

there is potentially room for the development of VI measures that explicitly examine interactions, little work has been performed to this point. In one of the few known methods, Jiang et al. [2009] proposed a sliding window method to scan for up to three-way interactions.

5.4 Two-Stage Analysis

While RF can successfully be used by itself, one of its greatest utilities comes in combination with other modeling approaches. Many authors have performed multistage analyses using RF as a first stage screening step and then followed up with a finer analytical method (usually logistic regression) [Briggs et al. 2010ab; De Lobel et al. 2010; Schwarz et al. 2007; Meng et al. 2007].

While this can be a very powerful tool for SNP discovery such two-stage approaches make inference more challenging. As discussed by Svetnik et al. [2004]

DOI: 10.2202/1544-6115.1691

such techniques can be subject to selection bias and inference will subsequently be biased. Ideally there would be one dataset on which to run RF and select SNPs and a second dataset on which to draw inference. If a second dataset does not exist a multiple testing correction based on the original number of predictors needs to be performed [Tuglus and van der Laan 2009].

5.5 Risk Prediction

RF was originally developed as a prediction tool. Traditionally, genetic studies have not been concerned with risk prediction, however as more disease susceptible loci have been identified for many disease, interest has shifted to predicting genetic risk [Janssens et al. 2011]. At this time much of that work has centered on summing marginal effects to derive an overall risk score (e.g. De Jager et al. [2009]; The International Schizophrenia Consortium [2009]). While initially valuable such methods do not adequately take into account the joint effect of multiple SNPs. As more disease studies shift from a paradigm of SNP detection to risk prediction, RF provides a valuable tool for such analyses.

Using RF for prediction does not change the previous discussion relating to choosing optimal tuning parameters. In many ways the task is easier, as one no longer needs to be concerned with correlated variables and the intricacies of VI measures. As with all bagging algorithms the final model generates a predicted probability for a class outcome based on the number of trees that vote for a particular class.

5.6 Other Uses of Random Forests

The derived collection of trees provides a significant amount of information about the complex relationships between predictors and observations. This information can be exploited for many additional uses including clustering, imputing missing data, detecting which observations are related (*proximities*), detecting outliers and graphing. Most of these are detailed on the main website for Random Forests [Breiman and Cutler 2010]. Cutler and Stevens [2006] provide an overview of some of these uses for genomic applications.

While many of these methods are implemented in most versions of RF, few of these have seen application. Some of these methodologies (e.g. imputing [Schwarz et al. 2009] and outlier detection) are probably better accomplished via other methods that take better advantage of genetic structure. However, other analyses (e.g. clustering and proximities) do have the potential to provide insight into the structure of genetic data. However, the primary aspect limiting their use

is the relative weakness of genetic data. Many of these analyses exploit very subtle relationships in the tree structure. Since genetic data is often weakly predictive [Clayton 2009], there is often not enough information content to accurately define these relationships. However these are methods worthy of further exploration.

5.7 Computational Considerations & Implementations of Random Forests

Computationally, RF is ideal for large genetic datasets. Trees are fast to grow, with the primary computational burden derived from pruning (which is not performed in RF). The algorithm scales very well with large data and is more than capable of handling GWA data. The number of observations often has a greater impact on computation, as more observations will lead to larger trees. This can be mitigated via *subbagging* or limiting tree size. More predictors increase computation only to the extent that `mtry` is increased requiring more predictors to be searched over at each node. One of the key appeals of RF is that since each tree is grown independently, the algorithm is very easy to parallelize, increasing its computational efficiency.

To date there are a number of different implementations of RF (see Table 2 for a brief list). Since there is always potential for implemented algorithms to get lost in translation we make no statement about each's accuracy as we have not used all of them. However, each has slightly different features and may be suitable for different data problems. The original code was written in Fortran by Leo Breiman and Adele Cutler and is available on their website [Breiman and Cutler 2010]. Their code was adapted for use within the R environment by Andy Liaw and Matthew Wiener in the package randomForest. Other R packages have been created as either add-ons (e.g. varSelRF) or as amendments of the RF algorithm (e.g. cforest). The original Fortan code was licensed to Salford Systems, and implemented with a GUI and is available for a licensing fee. It was one of the first versions capable of handling GWA data. Numerous open source versions are available, most geared towards handling large data problems. Possibly, the most developed is Random Jungle [Schwarz et al. 2010] implemented in C++ and able to handle hundreds of thousands of predictors.

6 Other Classifiers

Insight can be gained into an algorithm's function and utility by comparing it to other algorithms. We briefly mention three here: K-Nearest Neighbors (K-NN), Penalized Regression and Boosting.

DOI: 10.2202/1544-6115.1691

Implementation	Resource	Notes
Random Forests	`www.stat.berkeley.edu/~breiman/RandomForests/`	Fortran; No active development
randomForest	`cran.r-project.org/web/packages/randomForest/`	R package
randomSurvivalForest	`cran.r-project.org/web/packages/randomSurvivalForest/index.html`	R package Survival Data
Random Jungle	`www.randomjungle.org/rjungle/`	Handles GWA data

Table 2: Some common implementations of the RF algorithm

6.1 K-Nearest Neighbors

K-Nearest Neighbors (K-NN) goes back to the 1950s and is one of the simplest classification algorithms to implement. For simplicity, assume the input vector $\mathbf{x_0}$ is real valued, then calculate:

$$d_i = \|x_i - x_0\| \tag{11}$$

The K is a tuning parameter that determines how many neighbors to consider, ordered by d_i. The classification for x_0 is the majority vote of those K. A primary limitation of K-NN is that it provides little insight into VI.

Lin and Jeon [2006] compared RF to an adaptive form of K-NN. Particularly for classification, the relationship to K-NN stems from the fact that trees are grown to maximal depth, where often there will be only one class in the terminal node of a tree. Therefore over the forest of trees, the classification for a new observation will be a weighted version of a certain number of neighbors.

6.2 Penalized Regression

Penalized regression [Hastie et al. 2009] is an important class of algorithms that has increased in popularity with new computational "tricks." For classification this is done in the context of logistic regression. The general equation to optimize is:

$$max_{\beta_0,\beta}\left\{\sum_{i=1}^{N}[y_i(\beta_0+\beta^T x_i)-\log(1+e^{\beta_0+\beta^T x_i})]-\lambda\sum_{j=1}^{p}|\beta_j|^{\alpha}\right\} \tag{12}$$

α is a user set value that controls the type of penalty, while λ is a tuning parameter to optimize the function. When $\alpha = 1$ the penalty represents an ℓ_1 penalty and the algorithm is referred to as the LASSO. When $\alpha = 2$ the penalty represents an ℓ_2 penalty and the algorithm is the traditional Ridge regression. The Elastic Net algorithm allows the user to vary α between 1 & 2.

λ serves as a tuning parameter that controls the complexity of the model. The appeal of LASSO over Ridge regression is that the LASSO penalty will result in a more sparse solution. While the optimal Ridge solution, like RF, will shrink all coefficients towards 0 with few equal to 0, the optimal LASSO fit will have many coefficients at 0. This makes the LASSO ideal for variable selection.

The LASSO has been successfully implemented with GWA data [Wu et al. 2009]. The one concern with the LASSO, as opposed to RF, is that the method requires specifying a parametric linear model. The application by Wu et al. allows for a search for interaction terms, but it is unclear how successful it is with detecting complex genetic effects. In this sense tree structures are preferable.

The relationship between RF and Ridge regression stems from variable importance. Unlike algorithms such as LASSO and Boosting, which tend to produce a sparse solution, placing weight on only few variables, both RF and Ridge result in shrunken VI measures, allowing many variables to "speak." This is desirable when most variables are associated with the outcome resulting in a more stable solution with emphasis spread across the variables. However, when association is due only to correlation (LD) with a true causal variant, this results in what have been called biased importance scores [Strobl et al. 2007].

6.3 Boosting

Boosting has seen a large application in machine learning fields but has had no known applications to genetic data (a pubmed search of the terms "boosting" and "SNP" yielded no results). While there is extensive literature on boosting we briefly mention their appeal as a learner and some thoughts as to why it may not be appropriate for genetic data.

Boosting is an ensemble algorithm that like RF has trees as the base learner. However, unlike RF, these trees are not fully grown trees, often containing only a few nodes referred to as *stumps* (the number of nodes is a tuning parameter). While the trees in RF (and bagging) are identically distributed, the trees in boosting are not. Instead, each observation in the training set receives a weight that is updated based on some classification error (generally an exponential loss), with greater error, resulting in greater weight. Therefore each iteration of Boosting attempts to fit the classifier on those observations which is hardest to classify. By doing this,

DOI: 10.2202/1544-6115.1691

Boosting is able to both decrease variance (because of the aggregation of classifiers) and bias (by doing a better job on those that are miss-classified).

Multiple studies have shown Boosting to be as good as and often better than bagging and RF [Breiman 1996b; Dietterich 2000b; Hastie et al. 2009]. Moreover, Boosting, has similarities with LASSO, in that it tends to result in a much sparser solution than does RF.

However there are two problems with Boosting, one computational and one more systemic. Due to this bias reduction mechanism, Boosting is prone to overfitting. Therefore there needs to be constant CV to determine when to stop growing stumps. Computationally this is very expensive. However, a more systemic problem is that the performance of Boosting degrades quickly with noisy data, particularly compared to bagging and randomization procedures [Dietterich 2000b]. Therefore while a very attractive algorithm, Boosting is not entirely appropriate for genetic data which contains large samples and many irrelevant variables.

7 Conclusion

This paper walked through the theoretical background of RF, while highlighting relevant research. While in many ways RF is a black box algorithm, it can also easily be broken down into its components: classification, trees, bagging & randomization. Understanding how the algorithm works, particularly the components that control the bias and variance, allows the user to better control the output via the different tuning parameters.

Having worked with RF, we agree with Breiman, that it is a great "off-the-shelf" algorithm. Despite the particular challenges presented by genetic data, reasonable settings of the tuning parameters can often easily be found. The underlying algorithm is relatively fast and has the capacity to handle large GWA studies. The non-parametric tree structure allows for the existence of conditional and higher-order effects, ideal for capturing complex genetic relationships that are likely to exist. Users can manipulate tuning parameters to determine the settings most appropriate for their own data needs. VI measures relevant to the specific question of interest can be created. As genetic analyses transition from a focus on VI to risk prediction, RF should continue to prove valuable. In all, RF is well suited as a stand-alone analytic tool or a first step in conjunction with other methods.

This exposition barely touches on the many subtle questions that can be asked after fitting RF. Within the collection of trees there is a lot of information about the relationship between the variables and observations allowing for the exploration of clustering, proximities and visualization. While appealing, experience

has shown that these subtle relationships can only be gleaned when the overall predictor is fairly strong. Furthermore, we concur with critics of RF, that it is not a perfect algorithm. The VI measure that it produces are inherently ad-hoc and lacks any statistical properties (some work has been attempted to explore some statistical properties but so far none have been determined). The lack of sparsity makes it ill suited for variable selection.

As the "No Free Lunch" states, there is no perfect algorithm [Wolpert 1996]. As with all modeling situations it is important to find the right tool for the job. For large genetic data, RF can be the right tool, though it is does need to be used appropriately and with insight, nonetheless it is an important addition to the arsenal of tools available for genetic epidemiologists.

References

Breiman, L. (1996a): "Bagging predictors," *Machine Learning*, 24, 123–140.

Breiman, L. (1996b): "Bias, variance, and arcing classifiers," Technical report, UC Berkeley.

Breiman, L. (1996c): "Heuristics of instability and stabilization in model selection," *Annals of Statistics*, 24, 2350–2383.

Breiman, L. (1996d): "Out-of-bag estimation," Technical report, UC Berkeley.

Breiman, L. (2001): "Random forests," *Machine Learning*, 45, 5–32.

Breiman, L. and A. Cutler (2010): "Random forests," `http://www.stat.berkeley.edu/ breiman/RandomForests/`.

Breiman, L., J. Friedman, R. Olshen, and C. Stone (1984): *Classification and Regression Trees*, New York: Chapman & Hall.

Briggs, F. B., S. E. Bartlett, B. A. Goldstein, J. Wang, J. L. McCauley, R. L. Zuvich, P. L. De Jager, J. D. Rioux, A. J. Ivinson, A. Compston, D. A. Hafler, S. L. Hauser, J. R. Oksenberg, S. J. Sawcer, M. A. Pericak-Vance, J. L. Haines, and L. F. Barcellos (2010a): "Evidence for crhr1 in multiple sclerosis using supervised machine learning and meta-analysis in 12,566 individuals," *Human molecular genetics*, 19, 4286–95.

Briggs, F. B., B. A. Goldstein, J. L. McCauley, R. L. Zuvich, P. L. De Jager, J. D. Rioux, A. J. Ivinson, A. Compston, D. A. Hafler, S. L. Hauser, J. R. Oksenberg, S. J. Sawcer, M. A. Pericak-Vance, J. L. Haines, and L. F. Barcellos (2010b): "Variation within dna repair pathway genes and risk of multiple sclerosis," *American journal of epidemiology*, 172, 217–24.

Bühlmann, P. and B. Yu (2002): "Analyzing bagging," *Annals of Statistics*, 30, 927–961.

DOI: 10.2202/1544-6115.1691

Bureau, A., J. Dupuis, K. Falls, K. L. Lunetta, B. Hayward, T. P. Keith, and P. Van Eerdewegh (2005): "Identifying SNPs predictive of phenotype using random forests," *Genetic Epidemiology*, 28, 171–182.

Cantor, R. M., K. Lange, and J. S. Sinsheimer (2010): "Prioritizing GWAS results: A review of statistical methods and recommendations for their application," *Am. J. Hum. Genet.*, 86, 6–22.

Clayton, D. G. (2009): "Prediction and interaction in complex diseases," *Plos Genetics*, 5.

Cordell, H. J. (2009): "Detecting genegene interactions that underlie human diseases," *Nature Review Genetics*, 10, 392–404.

Cutler, A. (1999): "Fast classification using perfect random trees," Technical report, Utah State University.

Cutler, A. and J. R. Stevens (2006): "Random forests for microarrays," *Methods in enzymology*, 411, 422–32.

De Jager, P. L., L. B. Chibnik, J. Cui, J. Reischl, S. Lehr, K. C. Simon, C. Aubin, D. Bauer, J. F. Heubach, R. Sandbrink, M. Tyblova, P. Lelkova, E. Havrdova, C. Pohl, D. Horakova, A. Ascherio, D. A. Hafler, E. W. Karlson, M. S. Freedman, G. Edan, H. P. Hartung, C. H. Polman, L. Kappos, X. Montalban, D. Miller, P. O'Connor, H. P. Hartung, G. Comi, M. Filippi, L. Kappos, B. G. Arnason, S. Cook, D. S. Goodin, D. Jeffery, A. Traboulsee, G. C. Ebers, D. Langdon, D. S. Goodin, A. T. Reder, F. Zipp, S. Schimrigk, H. P. Hartung, M. Filippi, and J. Hillert (2009): "Integration of genetic risk factors into a clinical algorithm for multiple sclerosis susceptibility: a weighted genetic risk score," *Lancet Neurol*, 8, 1111–1119.

De Lobel, L., P. Geurts, G. Baele, F. Castro-Giner, M. Kogevinas, and K. Van Steen (2010): "A screening methodology based on random forests to improve the detection of gene-gene interactions," *European journal of human genetics*, 18, 1127–32.

Díaz-Uriarte, R. and S. Alvarez de Andrés (2006): "Gene selection and classification of microarray data using random forest," *BMC Bioinformatics*, 7, 3.

Dietterich, T. (2000a): "Ensemble methods in machine learning," *Lecture Notes in Computer Science*, 1857, 1–15.

Dietterich, T. (2000b): "An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization," *Machine Learning*, 40, 139–157.

Dietterich, T. and E. B. Kong (1995): "Machine learning bias, statistical bias, and statistical variance of decision tree algorithms," Technical report, Oregon State University.

Friedman, J. H. (1997): "On bias, variance, 0/1—loss, and the curse-of-dimensionality," *Data Mining and Knowledge Discovery*, 1, 55–77.

Friedman, J. H. and P. Hall (2007): "On bagging and nonlinear estimation," *Journal of Statistical Planning and inference*, 137, 669–683.

García-Margariños, M., I. López-de Ullbarri, R. Cao, and A. Salas (2009): "Evaluating the ability of tree-based methods and logistic regression for the detection of snp-snp interaction," *Ann Hum Genet*, 73, 360–369.

Gareth, J. M. (2003): "Variance and bias for general loss functions," *Machine Learning*, 51, 115–135.

Genuer, R., J. M. Poggi, and C. Tuleau (2008): "Random Forests: some methodological insights," Technical report, INRIA, URL `http://hal.inria.fr/inria-00340725/en/`.

Glaser, B., I. Nikolov, D. Chubb, M. L. Hamshere, R. Segurado, V. Moskvina, and P. Holmans (2007): "Analyses of single marker and pairwise effects of candidate loci for rheumatoid arthritis using logistic regression and random forests," *BMC Proceedings*, 1 Suppl 1, S54.

Goldstein, B. A., A. E. Hubbard, and L. F. Barcellos (2011): "A generalized approach for testing the association of a set of predictors with an outcome: A gene based test," Technical Report 274, UC Berkeley.

Goldstein, B. A., A. E. Hubbard, A. Cutler, and L. F. Barcellos (2010): "An application of random forests to a genome-wide association dataset: Methodological considerations & new findings," *BMC Genetics*, 11, 49.

Hastie, T., R. Tibshirani, and J. Friedman (2009): *Elements of Statistical Learning*, New York: Springer, 2 edition.

Ishwaran, H., U. B. Kogalur, E. H. Blackstone, and M. S. Lauer (2008): "Random Survival Forests," *The Annals of Applied Statistics*, 2, 841–860.

Janssens, A. C., J. P. Ioannidis, C. M. van Duijn, J. Little, M. J. Khoury, and Grips Group (2011): "Strengthening the reporting of Genetic RIsk Prediction Studies: the GRIPS Statement," *PLoS Med.*, 8, e1000420.

Jiang, R., W. Tang, X. Wu, and W. Fu (2009): "A random forest approach to the detection of epistatic interactions in case-control studies," *BMC bioinformatics*, 10, S65.

Kohavi, R. and D. H. Wolpert (1996): "Bias plus variance decomposition for zero-one loss functions," in *Machine Learning: Proceedings of the Thirteenth International Conference*

Lee, S. S., L. Sun, R. Kustra, and S. B. Bull (2008): "Em-random forest and new measures of variable importance for multi-locus quantitative trait linkage analysis," *Bioinformatics*, 24, 1603–10.

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

DOI: 10.2202/1544-6115.1691

Lunetta, K. L., L. B. Hayward, J. Segal, and P. Van Eerdewegh (2004): "Screening large-scale association study data: exploiting interactions using random forests," *BMC Genetics*, 5, 32.

Maenner, M. J., L. C. Denlinger, A. Langton, K. J. Meyers, C. D. Engelman, and H. G. Skinner (2009): "Detecting gene-by-smoking interactions in a genome-wide association study of early-onset coronary heart disease using random forests," *BMC proceedings*, 3, S88.

McCarthy, M. I., G. R. Abecasis, L. R. Cardon, D. B. Goldstein, J. Little, J. P. Ioannidis, and J. N. Hirschhorn (2008): "Genome-wide association studies for complex traits: consensus, uncertainty and challenges," *Nat. Rev. Genet.*, 9, 356–369.

McKinney, B. A., D. M. Reif, M. D. Ritchie, and J. H. Moore (2006): "Machine learning for detecting gene-gene interactions: a review," *Applied bioinformatics*, 5, 77–88.

Meng, Y., Q. Yang, K. T. Cuenco, L. A. Cupples, A. L. Destefano, and K. L. Lunetta (2007): "Two-stage approach for identifying single-nucleotide polymorphisms associated with rheumatoid arthritis using random forests and bayesian networks," *BMC proceedings*, 1, S56.

Meng, Y. A., Y. Yu, L. A. Cupples, L. A. Farrer, and K. L. Lunetta (2009): "Performance of random forest when SNPs are in linkage disequilibrium," *BMC Bioinformatics*, 10, 78.

Nicodemus, K. K., W. Wang, and Y. Y. Shugart (2007): "Stability of variable importance scores and rankings using statistical learning tools on single-nucleotide polymorphisms and risk factors involved in gene x gene and gene x environment interactions," *BMC proceedings*, 1, S58.

Price, A. L., N. J. Patterson, R. M. Plenge, M. E. Weinblatt, N. A. Shadick, and D. Reich (2006): "Principal components analysis corrects for stratification in genome-wide association studies," *Nature Genetics*, 38, 904–909.

Rodin, A. S., A. Litvinenko, K. Klos, A. C. Morrison, T. Woodage, J. Coresh, and E. Boerwinkle (2009): "Use of wrapper algorithms coupled with a random forests classifier for variable selection in large-scale genomic association studies," *Journal of computational biology*, 16, 1705–18.

Schwarz, D. F., I. R. Knig, and A. Ziegler (2010): "On safari to random jungle: a fast implementation of random forests for high-dimensional data," *Bioinformatics*, 26, 1752–8.

Schwarz, D. F., S. Szymczak, A. Ziegler, and I. R. Konig (2009): "Evaluation of single-nucleotide polymorphism imputation using random forests," *BMC Proc*, 3 Suppl 7, S65.

Schwarz, D. F., S. Szymczak, A. Ziegler, and I. R. Knig (2007): "Picking single-nucleotide polymorphisms in forests," *BMC proceedings*, 1, S59.

Segal, M. R. (2004): "Machine learning benchmarks and random forests regression," Technical report, CBMB Working Paper.

Statnikov, A., L. Wang, and C. F. Aliferis (2008): "A comprehensive comparison of random forests and support vector machines for microarray-based cancer classification," *BMC bioinformatics*, 9, 319.

Strobl, C., A. L. Boulesteix, T. Kneib, T. Augustin, and A. Zeileis (2008): "Conditional variable importance for random forests," *BMC bioinformatics*, 9, 307.

Strobl, C., A. L. Boulesteix, A. Zeileis, and T. Hothorn (2007): "Bias in random forest variable importance measures: illustrations, sources and a solution," *BMC Bioinformatics*, 8, 25.

Sun, Y. V. (2010): "Multigenic modeling of complex disease by random forests." *Advances in Genetics*, 72, 73–99.

Sun, Y. V., Z. Cai, K. Desai, R. Lawrance, R. Leff, A. Jawaid, S. L. Kardia, and H. Yang (2007): "Classification of rheumatoid arthritis status with candidate gene and genome-wide single-nucleotide polymorphisms using random forests," *BMC proceedings*, 1 Suppl 1, S62.

Svetnik, V., A. Liaw, and C. Tong (2004): "Variable selection in random forest with application to quantitative structureactivity relationship," in N. Intrator and F. Masulli, eds., *Proceedings of the 7th Course on Ensemble Methods for Learning Machines*, Springer-Verlag.

The 1000 Genomes Project Consortium (2010): "A map of human genome variation from population-scale sequencing," *Nature*, 467, 1061–1073.

The International HapMap Consortium (2003): "The International HapMap Project," *Nature*, 426, 789–796.

The International Schizophrenia Consortium (2009): "Common polygenic variation contributes to risk of schizophrenia and bipolar disorder," *Nature*, 460, 748–752.

Tibshirani, R. (1996): "Bias, variance and prediction error for classification rules," Technical report, University of Toronto.

Tuglus, C. and M. J. van der Laan (2009): "Modified fdr controlling procedure for multi-stage analyses," *Statistical applications in genetics and molecular biology*, 8, Article 12.

Wang, M., X. Chen, and H. Zhang (2010): "Maximal conditional chi-square importance in random forests," *Bioinformatics (Oxford, England)*, 26, 831–7.

Wolpert, D. (1996): "The lack of a priori distinctions between learning algorithms," *Neural Computation*, 1341–1390.

Wolpert, D. H. and W. G. Macready (1999): "An efficient method to estimate bagging's generalization error," *Machine Learning*, 35, 41–55.

Wu, T. T., Y. F. Chen, T. Hastie, E. Sobel, and K. Lange (2009): "Genome-wide association analysis by lasso penalized logistic regression." *Bioinformatics*, 25, 714–21.

DOI: 10.2202/1544-6115.1691