Cover Letter

Please consider this manuscript, entitled *The Lack of Cross-Validation Can Lead to Inflated Results and Spurious Conclusions: A Re-Analysis of the MacArthur Violence Risk Assessment Study*, for publication in the *Journal of Classification*. The research presented in this manuscript is original and the manuscript has not been submitted elsewhere. All ethical guidelines were followed in the conduct of this research. Correspondence concerning this article should be addressed to Ehsan Bokhari, Los Angeles Dodgers, 1000 Elysian Way, Los Angeles, CA, 90012. E-mail: ebokhari@msn.com. Telephone: 520-971-7054.

This paper discusses the importance of cross-validation in constructing a behavioral predictive model. To demonstrate, several classification tree models are constructed from the Violence Risk Assessment Study dataset that was used to develop the Classification of Violence Risk (COVR) assessment tool (Monahan et al., 2001). In the original construction, the instrument was not cross-validated; the results presented in the present paper suggest that the accuracy measures given in the original study were inflated and the model was overfit to the data. In addition, this article shows how adjusting the cutscore alters the cost ratio of false negatives to false positives and that by so doing, Monahan et al. (2001) implicitly set false negatives to be more costly than false positives.

Suggested reviewers for this article:

William Grove

University of Minnesota

grove001@umn.edu

Paul J. McCusker

Department of Veterans Affairs

paul.mccusker@va.gov

The Lack of Cross-Validation Can Lead to Inflated Results and Spurious Conclusions: A Re-Analysis of the MacArthur Violence Risk Assessment Study

Ehsan Bokhari

Lawrence Hubert

University of Illinois at Urbana-Champaign

Author Note

Ehsan Bokhari is now with the Los Angeles Dodgers, Los Angeles, California. Email: ebokhari@msn.com.

Abstract

Cross-validation is an important evaluation strategy in behavioral predictive modeling; without it, a predictive model is likely to be overly optimistic. Statistical methods have been developed that allow researchers to straightforwardly cross-validate predictive models by using the same data employed to construct the model. In the present study, cross-validation techniques were used to construct several decision-tree models with data from the MacArthur Violence Risk Assessment Study (Monahan et al., 2001). The models were then compared with the original (non-cross-validated) Classification of Violence Risk assessment tool. The results show that the measures of predictive model accuracy (AUC, misclassification error, sensitivity, specificity, positive and negative predictive values) degrade considerably when applied to a testing sample, compared with the training sample used to fit the model initially. In addition, unless false negatives (that is, incorrectly predicting individuals to be nonviolent) are considered more costly than false positives (that is, incorrectly predicting individuals to be violent), the models generally make few predictions of violence. The results suggest that employing cross-validation when constructing models can make an important contribution to increasing the reliability and replicability of psychological research.

The Lack of Cross-Validation Can Lead to Inflated Results and Spurious Conclusions: A Re-Analysis of the MacArthur Violence Risk Assessment Study

Cross-validation is an important part of constructing a behavioral predictive model. A failure to cross-validate may lead to inflated and overly-optimistic results, as Meehl and Rosen (1955) noted some sixty years ago: "If a psychometric instrument is applied solely to the criterion groups from which it was developed, its reported validity and efficiency are likely to be spuriously high" (p. 194). The Classification of Violence Risk (COVR; Monahan et al., 2001) assessment tool is an actuarial device designed to predict the risk of violence in psychiatric patients. The COVR is a computer-implemented program based on a classification tree construction method that has been praised for its "ease of administration" (McDermott, Dualan, & Scott, 2011, p. 4). When constructed, however, the COVR was not cross-validated; thus, the results from the construction sample may be overly optimistic (for example, see McCusker, 2007).

The research presented in this paper reanalyzes data from the MacArthur Violence Risk Assessment Study (VRAS) used to develop the COVR. We begin by describing a widely-applied method for cross-validation, commonly called K-fold cross-validation. Data are then presented from the MacArthur VRAS. Several classification tree models are built from the VRAS dataset demonstrating the importance of cross-validation. In addition, we show how differing cutscores (see Appendix A) implicitly affect the costs associated with false negatives and positives. The COVR implicitly assumes that false negatives (incorrect classifications of violent individuals) are more costly than false positives (incorrect classifications of nonviolent individuals).

A Brief Introduction to Cross-Validation

Cross-validation is an important tool for prediction, allowing the researcher to estimate the accuracy of a prediction tool in practice. Assessing the accuracy of a model with the same data used to create the model will give overly optimistic estimates of

accuracy because a model is typically fit by minimizing some measure of inaccuracy; thus, the model reflects both the true data pattern as well as error. Cross-validation is a strategy to separate these two entities.

Assume we have a dataset $(\mathbf{X}, \mathbf{y})$, where $\mathbf{X}$ is an $n \times p$ matrix containing n observations measured across p predictor variables, and $\mathbf{y}$ is an $n \times 1$ vector containing n observations measured on a single outcome variable (for example, the outcome of whether an act of violence was committed). In this scenario the outcome variable is known, and the construction of a model to predict the known values of $\mathbf{y}$ is typically referred to as *supervised learning*.

In prediction, interests generally center on modeling $\mathbf{y}$ as a function of $\mathbf{X}$, where it is assumed that for some function f,

$$\mathbf{y} = f(\mathbf{X}) + \boldsymbol{\varepsilon};$$

here, $\boldsymbol{\varepsilon}$ represents an $n \times 1$ vector of random error terms, assumed to have mean 0, finite variance, and be uncorrelated with the set of predictor variables. The primary goal is to estimate $f(\mathbf{X})$ so that a practical classification function,

$$\hat{\mathbf{y}} = \hat{f}(\mathbf{X}),$$

is constructed. The total error in prediction, $\mathbf{y} - \hat{\mathbf{y}}$, can be divided into two types of components: *reducible error* and *irreducible error*. Decomposing the mean-squared error gives

$$\begin{aligned}
\mathbb{E}\left[(\mathbf{y} - \hat{\mathbf{y}})^2\right] &= \mathbb{E}\left[(f(\mathbf{X}) + \boldsymbol{\varepsilon} - \hat{f}(\mathbf{X}))^2\right] \\
&= \mathbb{E}\left[(f(\mathbf{X}) - \hat{f}(\mathbf{X}))^2\right] + 2\mathbb{E}\left[(f(\mathbf{X}) - \hat{f}(\mathbf{X}))\boldsymbol{\varepsilon}\right] + \mathbb{E}(\boldsymbol{\varepsilon}^2) \\
&= \underbrace{\mathbb{E}\left[(f(\mathbf{X}) - \hat{f}(\mathbf{X}))^2\right]}_{\text{reducible error}} + \underbrace{\mathbb{V}(\boldsymbol{\varepsilon})}_{\text{irreducible error}},
\end{aligned}$$

where $\mathbb{E}(\cdot)$ and $\mathbb{V}(\cdot)$ represent the expected value and variance, respectively. These two types of error determine the accuracy of predictions. Typically $\mathbb{V}(\boldsymbol{\varepsilon})$ is unknown and hence

cannot be reduced; the reducible error can be minimized, which is of course the goal of predictive modeling. When the predicted function perfectly matches the true function, the total mean squared error equals $\mathbb{V}(\boldsymbol{\varepsilon})$; thus, $\mathbb{V}(\boldsymbol{\varepsilon})$ represents a lower bound for the total error. In practice, $f(\mathbf{X})$ is not known, so one can only strive to get close to this lower bound by constructing predictive models that produce the smallest total error values.

In constructing a predictive model, there are several available measures for assessing how well the model fits the data (for example, the mean squared error, the coefficient of determination [R^2], the proportion of predictions correct). It is important to note that this assessment of error is often made with the sample used to construct the model and not on predictions with an independent sample. When a model is constructed for purposes of prediction, the model's predictive accuracy on new data is most relevant. Suppose there is a given measure of accuracy, say γ, for assessing the model and this measure is obtained with the same data used to construct the model. A way of evaluating a model's predictive ability is to collect new data and measure how accurate the predictions are; that is, a new accuracy measure γ' is obtained. The difference between γ and γ' represents the drop in how well the model predicts (assuming that a larger γ is associated with better accuracy, typically, $\gamma - \gamma' > 0$); this drop is known as *shrinkage*. Rather than assessing predictive accuracy with the same data relied on to build the model, the original data can be randomly split into two parts: the *training data* and the *testing data*. The training data is for constructing the model; the testing data is for estimating the predictive accuracy of the model. This process is typically more efficient in terms of time and cost than collecting new data after the model is developed, and can help prevent overfitting.

K-fold Cross Validation

Given n observations, cross-validation involves splitting the data so that a specified proportion, say q, of the data is present in the training set and the rest of the observations are in the testing set (that is, qn observations are in the training set and $(1-q)n$ are in the

testing set). This can be carried out multiple times, choosing a different training and testing set each time.

K-fold cross validation is one of the more popular cross-validation strategies and will be the only one discussed. K-fold cross-validation involves splitting the data into K subsets; the training set consists of the union of $K - 1$ subsets, and the testing set is defined by the remaining observations. This process is repeated so that each subset acts once as the testing sample. Because each replication of this process will produce results that vary, it is common to compute an average across all replications. The simplest form of K-fold cross validation is to let $K = 2$: the training set contains half of the observations, and the testing set the other. The most computationally costly form is to let $K = n$, so that each observation acts as the testing sample; this is commonly known as *leave-one-out cross-validation*. In addition to the computational costs, another disadvantage of leave-one-out cross-validation is that the variance of the estimate can be relatively large compared to other estimates—it is, however, approximately unbiased; setting $K < n$ provides an estimate of the test error with less variance but more bias (see Hastie, Tibshirani, & Friedman, 2009, pp. 242–243). A reasonable choice for K is commonly considered to be ten (Breiman & Spector, 1992). Appendix B presents a thorough discussion of what is known as the "bias-variance trade-off" in prediction, and how cross-validation can help determine the equilibrium point that simultaneously minimizes both the bias and the variance of a model.

Decision Trees

Decision trees, commonly referred to as *Classification and Regression Trees* (Breiman, Friedman, Olshen, & Stone, 1984), are popular statistical learning techniques generally used for prediction. Consider an $n \times p$ data matrix, $\mathbf{X}$, containing n observations measured across p predictor variables, and an $n \times 1$ vector $\mathbf{y}$ containing n observations measured across a single outcome variable. The outcome variable contained in $\mathbf{y}$ is the variable of

interest with respect to prediction. When the outcome variable is continuous, regression trees are constructed; when categorical, classification trees are constructed. For violence prediction in general, the outcome variable is binary (that is, categorical with two classes) representing the presence or absence of an act of violence; because of this specification, our focus is solely on classification trees.

Classification trees are constructed by first splitting the data into two disjoint subsets based on one of the p predictor variables. Within each subset, further partitioning of the data is done, and within the resulting subsets this process continues until some user-specified stopping criterion is reached; the complete procedure is known as *recursive partitioning*. Recursive partitioning is a *top-down greedy* algorithm: top-down because the algorithm begins with a tree with no splits and works "down" to a tree with many splits (and once a split is made, it remains); greedy because each split made is the "best" conditioned on the given splits (and not on possible future splits). An observation that falls into a subset with no further splits (called a *terminal node*) is classified based on all the observations within that subset, which is typically the modal observation (that is, the most prevalent outcome within the node), but other choices are possible, as will be discussed shortly.

The first split occurs at the *root node* of the tree; extending *branches* from the root node lead to subsample nodes, called *leaves*. As mentioned, the splits continue until a specified criterion is met, such as a constraint on the minimum number of observations in a given leaf, or a criterion based on significance testing. After a tree is constructed, it can be *pruned* to reduce the number of branches, eliminating those that add less to the tree's predictive ability. The notion behind pruning is to create a subtree that has better predictive accuracy on new data, and thus, the level of pruning is commonly determined by cross-validation.

The data are subjected to splitting with the goal of grouping the observations so as to minimize the number of observations incorrectly classified. There are several ways to

assess goodness of fit in classification, one being the *Gini index* (Breiman et al., 1984; Gini, 1912). Given R classes for an outcome variable, the Gini index for a group of observations is given by

$$G = \sum_{r=1}^{R} g_r(1 - g_r) = 1 - \sum_{r=1}^{R} g_r^2,$$

where g_r $(r = 1, \ldots, R)$ is the proportion in the group from the rth class. The Gini index can be thought of as a measure of impurity. Note that when $g_r = 1$, $1 - g_r = 0$, and $G = 0$ indicating perfect purity. When $g_1 = \cdots = g_R = 1/R$ (that is, the proportion of observations are evenly divided among the R classes), the index is maximal at $G = 1 - \frac{1}{R}$. The Gini index is commonly used for determining the splits of the trees, where the split minimizing the Gini index is chosen at each step of the partitioning.

Decision trees are popular because they are easy to interpret, but they are not the best statistical learning method in terms of predictive accuracy. Predictive accuracy can be enhanced by various ensemble methods such as *tree bagging* and *random forests* (Breiman, 2001; these are more generally known as *random subspace methods*). *Bagging* (the term *bag* is a short-hand phrase for bootstrap aggregation; see Breiman, 1996) is an ensemble learning method designed to avoid the overfitting of a model and is commonly used with classification trees (that is, tree bagging). Suppose a dataset, $\mathbf{X}$, contains n observations on p predictors. Similar to the bootstrap method, B training sets of size L are generated, where $1 \leq L \leq n$, by randomly sampling (with replacement) from $\mathbf{X}$; each training set is fit by the model and, after aggregating, an average over the B replications provides a predicted response for each observation.

Note that some of the observations in the ith training set, $\mathbf{B}^{(i)}$, may be duplicate. The larger L is, the more likely there will be at least one duplicate observation; the probability of such an event is $1 - \frac{n!}{n^L(n-L)!}$. The probability that any given observation is not selected is $(1 - \frac{1}{n})^L$. If $L = n$ and as $n \to \infty$, the probability approaches $e^{-1} \approx .37$. For a large enough n and when the training sample is equal to n, it would be expected that on average, about 63% of the bootstrap sample consists of unique observations. The 63%

represents a probabilistic lower bound; for $L < n$, one would expect more than 63% of the sample to be unique (for example, in the trivial case where $L = 1$, there are no duplicate observations). The approximately 37% of the observations not used in fitting the model on the ith replication are called *out-of-bag* (OOB); thus, the OOB observations are the testing set and can be used to assess predictive accuracy. For any given observation, by aggregating over the subset of B replications—where the observation was not used to fit the model—the average OOB prediction accuracy can be calculated and compared to the cross-validated error; this comparison gives us the average OOB error difference. The OOB errors can also be used to assess the importance of predictors by randomly permuting the OOB data across variables one at a time, and estimating the OOB error after permutation—a large increase in the OOB error indicates the variable's importance in the model.

Random forests (Breiman, 2001) are tree bagging methods that randomly select a subset of predictors to be used at each split. The advantage here is that it can "decorrelate" the trees by preventing a single variable from dominating the analysis; for instance, if one predictor is very strong, it will likely be the root node for a majority of the trees constructed; the subsequent nodes will be similar as well (that is, the trees will be highly correlated). By convention (and default in R), $\sqrt{p}$ predictors are randomly selected at each node for classification trees.

The advantage of ensemble tree models is that they tend to reduce the variance found in single decision tree models, leading to more accurate results by aggregating over a number of single decision trees. For more information on decision trees and other statistical learning models, the reader is referred to Kuhn and Johnson (2013), Hastie et al. (2009), and James, Witten, Hastie, and Tibshirani (2013); the latter two references are freely available online.

Here, a classification tree was first developed similar to that done in Monahan et al. (2001); next random forests were built to create better classification tree models with

better predictive accuracy. All classification tree models were constructed in R (R Core Team, 2014). Before proceeding to the results, however, it is necessary to discuss how a classification tree classifies observations and how this process is related to the costs of false positives and negatives.

Misclassification Costs

In general, there are two types of misclassifications that are of concern: false positives and false negatives. A false negative incorrectly predicts the absence of whatever the model is designed to predict (for example, predicting nonviolence in a violent individual); a false positive incorrectly predicts the presence (for example, predicting violence in a nonviolent individual). These two types of misclassifications can have drastically different consequences, and one may assign differing costs to each.

Suppose in a given terminal node there are n observations, of which n_r are from class r ($r = 1, \ldots, R$). An observation is classified into class r based on the modal class at that given terminal node; thus, if $n_r > n_{r'}$ for all $r \neq r'$, all observations within the terminal node are classified as belonging to class r. The empirical posterior probability for each class can be defined as the number of observations in the terminal node coming from a particular class divided by the total number of observations; thus, the estimated posterior probability is

$$\hat{P}(r|\mathbf{x}) = \frac{n_r}{n},$$

where $\mathbf{x}$ is a vector of predictor variables associated with the observation. Given this definition, an observation is classified as coming from class r when $\hat{P}(r|\mathbf{x}) > \hat{P}(r'|\mathbf{x})$ for all $r \neq r'$.

As an addition to the classification process, costs can be assigned to misclassifications; the cost function is labeled $C_s(r)$ and represents the cost of classifying an observation into class s when it truly belongs in class r (note that $C_r(r) = 0$). By

including a cost function, an observation is classified into class r by minimizing

$$\sum_{r=1}^{R} \hat{P}(r|\mathbf{x})C_s(r)$$

across all s. Note that when $C_s(r)$ is the same for all $r = 1, \ldots, R$ (that is, the costs are equal across all classes), the previous situation obtains and an observation is classified based on the modal class.

Given two classes (that is, $r = 1, 2$, where 1 could represent nonviolent individuals and 2, those who are violent), an observation is classified into class $r = 2$ when

$$\hat{P}(2|\mathbf{x})C_1(2) > \hat{P}(1|\mathbf{x})C_2(1).$$

With respect to classification of nonviolent and violent individuals, $C_1(2)$ and $C_2(1)$ are, respectively, the costs associated with a false negative and a false positive. Alternatively, the above inequality can be written as

$$\frac{C_1(2)}{C_2(1)} > \frac{\hat{P}(1|\mathbf{x})}{\hat{P}(2|\mathbf{x})}.$$

The lower bound, $\frac{\hat{P}(1|\mathbf{x})}{\hat{P}(2|\mathbf{x})}$, is the conditional odds in favor of the event 1 because $\hat{P}(2|\mathbf{x}) = 1 - \hat{P}(1|\mathbf{x})$; for example, the odds in favor of an individual not being violent, given the data.

If $C_1(2) = C_2(1)$, an observation is classified as coming from class 2 when $\hat{P}(2|\mathbf{x}) > \hat{P}(1|\mathbf{x})$, or equivalently,

$$\frac{\hat{P}(2|\mathbf{x})}{\hat{P}(1|\mathbf{x})} > 1.$$

Bayes Theorem allows this to be rewritten as

$$\frac{\frac{\hat{P}(\mathbf{x}|2)\hat{P}(2)}{\hat{P}(\mathbf{x})}}{\frac{\hat{P}(\mathbf{x}|1)\hat{P}(1)}{\hat{P}(\mathbf{x})}} = \frac{\hat{P}(\mathbf{x}|2)\hat{P}(2)}{\hat{P}(\mathbf{x}|1)\hat{P}(1)} > 1.$$

Considering $\hat{P}(\mathbf{x}|2)$ and $\hat{P}(\mathbf{x}|1)$ fixed, the classification cutscore can be changed by adjusting $\hat{P}(1)$ and $\hat{P}(2)$; these probabilities are the sample base rates (note that for $r = 1, 2$, $\hat{P}(2) = 1 - \hat{P}(1)$). Thus, adjusting the prior probabilities is an equivalent way of adjusting costs.

As will be discussed shortly, Monahan et al. (2001) suggested the cutscore for classification of high-risk individuals as twice the sample base rate of violence (approximately .37). This implies that an individual is classified as violent when the individual belongs to a terminal node where $\hat{P}(2|\mathbf{x}) > .37$, and implicitly assigns unequal costs to false positives and negatives. Explicitly, let $\hat{P}(2|\mathbf{x}) = .37$ (and consequently, $\hat{P}(1|\mathbf{x}) = .63$) so

$$\frac{C_1(2)}{C_2(1)} = \frac{\hat{P}(1|\mathbf{x})}{\hat{P}(2|\mathbf{x})} = \frac{.63}{.37} = 1.67.$$

By lowering the cutscore to .37 for classification of violence, the authors imply that, given the specified prior probabilities, incorrectly classifying an individual as nonviolent (a false negative) is 1.67 times worse than incorrectly classifying an individual as violent (a false positive).

Most authors of actuarial measures for violence risk assessment are reluctant to discuss the costs of false positives versus false negatives (Mossman, 2006, 2013; Vrieze & Grove, 2008); an exception to this is Richard Berk. In his book, *Criminal Justice Forecasts of Risk: A Machine Learning Approach* (Berk, 2012), he suggests that

> *the costs of forecasting errors need to be introduced at the very beginning when the forecasting procedures are being developed* [original emphasis]. Then, those costs can be built into the forecasts themselves. The actual *forecasts* [original emphasis] need to change in response to relative costs. (p. 20)

In the examples that Berk provides (regarding parole release), he suggests that the ratio of false negatives to false positives be as high as twenty to one (also, see Berk, 2011). The reasoning for such an extreme ratio, as justified by Berk (2012), is that the agency the model was built for was "very concerned about homicides that could have been prevented" (p. 5). Thus, the "agency" was willing to accept that a large number of potentially non-violating parolees were not granted parole; the proportion of those predicted to fail that actually did was only about .13 for the sample data used in the text (see Table 1.1 in Berk, 2012)).

The MacArthur Violence Risk Assessment Study

The Classification of Violence Risk (COVR; Monahan et al., 2006) is an assessment instrument developed from the MacArthur Violence Risk Assessment Study (VRAS). The COVR is computer-implemented and designed to estimate the risk of violence in psychiatric patients; given the appropriate credentials, it is available for purchase from Psychological Assessment Resources (`http://www4.parinc.com`). The COVR assigns patients to one of five risk groups defined by the "likelihood that the patient will commit a violent act toward another person in the next several months" (Monahan et al., 2006, p. 728). Table 1 gives the five risk groups defined by their best point estimates and 95% confidence intervals.

The development of the COVR is detailed in Monahan et al. (2001; also see Steadman et al., 2000, Monahan et al., 2000, and Banks et al., 2004 for less detailed reviews). The COVR was based on a sample of 939 recently-discharged patients from acute inpatient psychiatric facilities in three locations within the United States: Pittsburgh, Pennsylvania; Kansas City, Missouri; and Worcester, Massachusetts. Patients were restricted to those who were white, African-American, or Hispanic; English-speaking; between the ages of 18–40; and charted as having thought, personality, or affective disorder, or engaged in substance abuse.

According to the original MacArthur study (Monahan et al., 2001), violence is defined as "acts of battery that resulted in physical injury; sexual assaults; assaultive acts that involved the use of a weapon; or threats made with a weapon" (p. 17). A second category of violent incidents was labeled as "other aggressive acts" (Monahan et al., 2001, p. 17) including non-injurious battery; verbal threats were not considered. The outcome variable of violence is dichotomous—either the patient committed an act of violence or did not. It does not consider the number of violent acts or their severity. The patients were interviewed once or twice during the twenty weeks after discharge. Of the 939 patients, 176 were considered violent; thus, the base rate for violence in this sample is .187.

The authors identified 134 potential risk factors, listed in detail in Monahan et al.

(2001). Using SPSS's CHAID (chi-squared automatic interaction detection) algorithm (SPSS, Inc., 1993), the authors developed a classification tree based on the given risk factors. The final classification model was constructed by an iterative classification tree (ICT) process: after an initial classification tree was developed, those who were still unclassified (that is, those within .09 to .37 estimated probabilities of committing violence according to the model) were reanalyzed using the same CHAID algorithm. After four iterations, 462 patients were classified as low risk (less than .09 probability of committing violence), 257 were classified as high risk (greater than .37 probability of committing violence), and 220 remained unclassified. The cutoffs of .09 and .37 were chosen because they represent, respectively, one half and twice the base rate of violence in the sample.

The authors' goal was to create an actuarial tool that was "clinically feasible"; thus, it was to include only risk factors that could be computed easily. Of the 134 original risk factors, 28 were eliminated that "would be the most difficult to obtain in clinical practice" (Monahan et al., 2001, p. 108), as determined by the length of the instrument measuring the risk factor (more than twelve items was considered too long), or the risk factor not being readily or easily ascertainable by mental health professionals. After doing so, the same ICT method was applied to the 106 remaining risk factors using three iterations.

The correlation between the predictions made by the clinically-feasible and original ICT models was .52; the authors noted the low correlation:

> The fact that these [two] prediction models are comparably associated with the criterion measure, violence (as indicated by the ROC analysis), but only modestly associated with each other [as indicated by the correlation coefficient], suggested to us that each model taps into an important, but different, interactive process that relates to violence. (p. 117)

The authors then constructed nine additional ICT models using the 106 clinically-feasible variables; for each of the nine trees the authors "forced a different initial variable" (p. 118; that is, the root nodes for the ten trees differed). The ten models led to ten classifications

for each individual (high, average, or low) and each individual was assigned a score corresponding to their risk level (1, 0, or -1, respectively); the scores were then summed to create a composite score ranging from -10 to 10. The authors remarked, "As two models predict violence better than one, so ten models predict violence better than two (that is, the area under the ROC curve was .88 for ten models compared to .83 for two models)" (p. 122).

The authors questioned whether ten models were necessary; to determine this empirically they performed stepwise logistic regression and concluded that only five of the ten were needed, leading to composite scores ranging from -5 to 5 (the AUC remained the same, .88). The composite scores were divided into five distinct groups based on the following ranges: $[-5, -3]$, $[-2, -1]$, $[0, 1]$, $[2, 3]$, and $[4, 5]$ (these five groups correspond to, respectively, the very-low, low, average, high, and very-high risk groups found in Table 1; the probabilities represent the proportion of those violent within each group).

The authors did not cross-validate their model. As Monahan et al. (2001) state on page 106, "Dividing the sample leaves fewer cases for the purpose of model construction" and, quoting Gardner, Lidz, Mulvey, and Shaw (1996), "wastes information that ought to be used estimating the model." When their ICT models were constructed in the late 1990s and early 2000s, computing power was not what it is now, but cross-validation on a dataset of 939 was certainly possible (although possibly not in the version of SPSS relied on). With today's computing power there is little reason not to cross-validate a model or to argue that cross-validation "wastes" data. As will be shown, cross-validated error can be drastically different from what is called the resubstitution error for the initially constructed model.

As noted, Monahan et al. (2001) cited the Gardner et al. (1996) source when they made their remark claiming cross-validation wastes information, so this reasoning did not necessarily originate with them. Looking at the Gardner et al. (1996) article referenced in the quote above, a footnote on page 43 states that a bootstrap cross-validation was performed on the authors' logistic regression model, a perfectly reasonable alternative.

Although Monahan et al. performed a bootstrap analysis to estimate the variability of the predictions (and where 1,000 bootstrap samples helped estimate 95% confidence intervals for the probability-of-violence point estimates given in Table 1), the base predictive model was not cross-validated.

VRAS Dataset

To illustrate the process of cross-validation, we used the data from the MacArthur Violence Risk Assessment Study (Monahan et al., 2001) to construct several decision-tree models. As noted, the data were obtained from 939 patients discharged from inpatient psychiatric facilities based in Pittsburgh, Kansas City, and Worcester, MA. The ages of the patients range from 18 to 40 (Mean = 29.9; Median = 30.0). Of the 939 patients, 538 (57%) were male; 645 (69%) were White, 273 (29%) were African-American, and 21 (2%) were Hispanic.

The response variable (`Violence`) is a binary outcome variable representing whether an act of violence took place within the follow up period (`Violence` = 1 if an act of violence occurred; `Violence` = 0 if not). Thirty-one predictor variables were included based on the results from the main effects logistic regression and iterative classification tree models in Monahan et al. (2001). The data are available for download through the MacArthur Research Network website (`http://www.macarthur.virginia.edu/risk.html`); the dataset for the present analysis was obtained directly from the MacArthur researchers—it is a "cleaned-up" version from the statistician on the project. The best attempt was made for preprocessing the data to match that in the Monahan et al. analysis. All software code, including the preprocessing as well as variable descriptions, can be found in the supplementary material appended to our report.

As a way of comparing how close our variables match those of the MacArthur authors, Pearson product-moment correlation coefficients between the predictor variables and `Violence` were compared to those found in Chapter 5 of Monahan et al. (2001; see

Tables 5.2, 5.3, and 5.5). Table 2 displays the estimated correlations for each predictor variable with the response variable as well as the reported correlations from Monahan et al. Although not a foolproof method for confirming that the variables were preprocessed in a similar manner, it certainly does indicate discrepancies that may exist. There is a lot of agreement (at least to two decimal places), but it is not complete. Seven of the 31 correlations disagree, six only by one percent. The largest discrepancy is between prior head injury ($r = .03$ vs. $r' = .06$).

VRAS Classification Tree Model

The number of observations at each node in a classification tree is referred to as the *leaf size*; the minimum leaf size is a constraint provided by the modeler (the default in R is 7; the minimum leaf size set by Monahan et al. 2001 was 50). Rather than using the default setting, we determined the minimum leaf size using cross validation. The minimum leaf size is plotted against the leave-one-out cross-validated error, shown in Figure 1 (solid line). Several minimum leaf sizes give a cross-validated error less than the base rate, implying that the expected error on new data is less than the error when simply predicting all individuals to be nonviolent (that is, what is often called "base-rate prediction"; see Appendix A for further details); the minimum cross-validated error of .179 was obtained at the minimum leaf size of 48. As a comparison, the dotted line in Figure 1 gives the resubstitution error at each minimum leaf size (that is, the misclassification error on the same data used to construct the model); as the tree becomes less complex, or less flexible, (that is, the minimum leaf size increases), the resubstitution error increases toward the base rate. Trees with a resubstitution error equal to the base rate represents those without any branches (that is, trees with only a root node so that no partitions are being made). The observation that the resubstitution error increases as the trees become less complex, and that the cross-validated error decreases and then increases, is an example of the trade-off between bias and variance in predictive models (see Appendix B; also Hastie et al., 2009).

Based on a minimum leaf size of 48, the initial classification tree constructed is shown in Figure 2. The resubstitution error (with a cutscore of .50) was .181, implying the misclassification of 170 observations; the cross-validated error was slightly lower, .179. Both measures indicate the model was outperforming base-rate prediction (the base rate for violence in the sample was .187 so base-rate prediction would be to predict all individuals to be nonviolent resulting in a misclassification error of .187; see Appendix A for more details). Based on a .50 cutscore, 48 individuals were classified as violent (27 of whom were) and the rest nonviolent.

The same analyses were repeated but the cost matrix was set so the cutscore for classifying individuals was twice the baserate (that is, .37); thus, false negatives were considered to be about 1.67 times more costly than false positives. The minimum leaf size was determined to also be 48 (in a similar fashion to that done using equal costs); this led to the same tree being produced.

If we adhered to Berk's (2012) 20:1 false negative to false positive ratio, the resubstitution error (with a minimum leaf size of 30) is .570 and the cross-validated error is .586. Because of these results and the fact that it is difficult to justify a 20:1 ratio for the VRAS definition of violence, this cost ratio is not considered in the remaining analyses.

Suppose one decided not to empirically determine a minimum leaf size but let the minimum leaf size be one, the default setting in some software (e.g., MATLAB). In doing so, the resubstitution error for such a tree was .009; only 8 of the 939 patients were misclassified. The sensitivity of the model (that is, the proportion of violent individuals predicted to be violent) was .966; the specificity (that is, the proportion of nonviolent individuals predicted to be nonviolent) was .997; the positive predictive value (PPV; that is, the proportion of violent predictions that were correct) was .988; the negative predictive value (NPV; that is, the proportion of nonviolent predictions that were correct) was .992; the AUC (the area under the receiver operating characteristic curve) was .999. No method in the literature for predicting violence comes close to these accuracy measures. Without

cross-validating this model, however, one blindly believes that an extremely capable model for predicting violence has been found; the cross-validated error was .296, providing strong evidence that the model overfit the data. Based on a cutscore of .37 rather than .50, the model with a minimum leaf size of one misclassified 12 individuals (resubstitution error of .013) but the cross-validated error was .279. Again, this exemplifies the overfitting of a model and the importance of cross-validation, and provides an example of how a more flexible model (smaller minimum leaf size) performs well on the data for which the model was fit but far worse on new data. A good fit does not imply a good model (Roberts & Pashler, 2000).

VRAS Random Forest Model

Random forest models were next implemented for the VRAS dataset. A subset of the VRAS dataset was randomly selected as the training sample with the remaining observations representing the testing sample. The testing sample contained 30% of the original data (282 observations); the training data contained the remaining 657 observations (the base rate for violence in the training sample was .181, and .202 in the testing sample). The random forest model was fit to the training sample for $B = 1000$ trees and a minimum leaf size of 10. After fitting 1000 trees, the random forest model was used to predict violence in the training set (that is, the observations used for fitting the model); the predictions were perfect. The 1000 trees generated were aggregated to estimate the probability an individual will be violent by computing the proportion of times the individual is classified as violent (an individual was classified as violent if the predicted probability exceeded .50; that is, costs were considered equal here).

The results discussed thus far are, as noted, based on the training data. The greater concern is with how well violence can be predicted in new observations with the random forest model; this is evaluated with the testing data. Of the 282 observations, two were predicted—one incorrectly—to be violent (see Table 3). The cross-validated error was .202,

equal to the base rate. The sensitivity and specificity were, respectively, .018 and .973; the positive and negative predictive values were, respectively, .500 and .800.

The next analysis was the same as the previous one except that the cost ratio of false negatives to false positives was set to 1.67. At the individual tree level an observation was classified as violent when it belonged to a terminal node where the proportion of violent individuals was greater than .37; at the aggregate level (that is, across all 1000 trees) an individual was predicted to be violent when classified as violent in more than 37% of the trees. The results for the training data were again perfect; for the testing data, the results can be found in the bottom of Table 4.

As expected, more predictions of violence were made. For the testing data, the model classified 8.2% of the sample as violent—compared to 0.4% when costs were equal. The random forest model appears to be performing fairly well on the testing data (cross-validated error: .177).

Out-of-bag Prediction

Rather than splitting the data prior to fitting the ensemble method, the entire dataset can be used and cross-validation error estimated from the OOB observations; this maximizes sample size (that is, nothing is "wasted") and a cross-validated error is still obtained. The results produced by a cutscore of .50 are displayed in the top of Table 5. The OOB error was .192, slightly more than the baserate. The sensitivity of the model was .057; the specificity was .982; the positive predictive value was .417; the negative predictive value was .819; and the AUC was .75.

Carrying out the same analysis but setting the classification cutscore to .37 produced similar results. As the bottom of Table 5 shows, the model classified 2.9% of individuals as violent. The sensitivity of the model was .074; the specificity is .980; the positive predictive value is .464; the negative predictive value is .821. The OOB error was .190 with an AUC of .75.

Variable Selection

Thus far the decision trees have included all thirty-one variables. Out-of-bag observations allow the quantification of variable importance to the classification trees. For each variable the values are randomly permuted and the increase (or decrease) in the OOB error calculated (that is, the difference in OOB error before and after permutation). This is carried out for every tree and normalized with the standard deviations of the differences. Variables with larger average differences can be quantified as more important than variables with smaller averages. A dot plot displaying the variable importance is given in Figure 3.

From Figure 3, several variables appear to be more important than others (in particular `PCL`), and several actually *decrease* the OOB error after permutation. It is interesting to note that `Schiz` is among the more important variables but `Age` is not. The decision for removing variables is conservative; only variables with average OOB error differences near and less than zero are removed (`PCS`, `Suicide`, `Consc`, `Age`, `DadArr`, `HeadInj`, and `Threats`); thus, the final model consists of twenty-four predictor variables.

Final Model

The final model was estimated with 1000 trees omitting the variables discussed in the previous section and with equal costs. The estimated cross-validated error using OOB observations for the final model is .186, a slight improvement upon the random forest model that included all thirty-one variables. The results are summarized in Table 6. The sensitivity of the model is .097; the specificity is .979; the PPV is .515; and the NPV is .825. The ROC plot is given in Figure 4; the AUC for the final model is .75.

Conclusion

The results given in this paper reiterate the argument that predicting violent behavior is extremely difficult. Unless unequal costs regarding false positives and negatives are assumed—particularly when false negatives are considered to be more costly than false

positives—the models made a small number of predictions of violence which consequently, led to very low sensitivity. Therefore, unless the model explicitly states false negatives as more costly than false positives, predictions of violence are infrequent and the sensitivity is abysmally low. Even when false negatives were stated to be 1.67 times greater than false positives, the sensitivity was far from adequate in the models. Harris and Rice (2013) claim "it can be reasonable for public policy to operate on the basis that a miss (for example, failing to detain a violent recidivist beforehand) is twice as costly as a false alarm (for example, detaining a violent offender who would not commit yet another violent offense)" (p. 106). Whether this is true, it is ethically questionable to assume that costs are anything but equal unless public policy explicitly states otherwise.

The analyses presented also demonstrate the importance of cross-validation. Without cross-validating a model, results and conclusions regarding accuracy can be misleading and overly optimistic. Each type of classification tree model constructed performed quite well when assessed on the data used to fit it, and almost always outperformed base-rate prediction. But when cross-validation methods were implemented, the results were dramatically different. Rarely did the model outperform base-rate prediction on the testing sample, and the resubstitution error was often higher, indicating that the models even failed to outperform base-rate prediction on the training data.

The random forest model was chosen for two reasons: (a) it is a decision tree model and the COVR is based on a decision tree model and (b) random forests have been shown to be highly accurate models in real world applications (Fernández-Delgado, Cernadas, Barro, & Amorim, 2014). To be sure, a logistic regression model and linear and quadratic discriminant analysis models were also fit for comparison (see the supplementary material for the full details). The results were similar, but the logistic regression model performed best and slightly outperformed the final random forest model in terms of cross-validated error and AUC. But because of the reasons just given, the random forest model was chosen to represent the final model. The final model also did not incorporate unequal costs; as

mentioned, unless public policy explicitly states that false negatives should be considered more costly than false positives with respect to the type of violent behavior being predicted in the VRAS sample, we feel that costs do not warrant adjustment. If costs are considered unequal, they should be determined a priori and not from an optimization based on the data; we agree with the quote from Berk (2012) when he says, "the costs of forecasting errors need to be introduced at the very beginning when the forecasting procedures are being developed." Optimizing a model based on differing cost ratios could lead to unethical decision making and unintended consequences.

Table 7 is from the VRAS study (derived from Table 6.7 in Monahan et al., 2001). When individuals who fall into the very high- and high-risk groups are classified as violent and all others as nonviolent, the model correctly classifies 86.0% of individuals, better than nearly every model presented in the current analysis. At this cutscore the model has a sensitivity and specificity of $\frac{(48+57)}{176} = .60$ and $\frac{(135+229+339)}{763} = .92$, respectively; the positive and negative predictive values are $\frac{(48+57)}{(63+102)} = .64$ and $\frac{(135+229+339)}{(183+248+343)} = .91$, respectively. Recall that the COVR was a combination of ten ICT models, five of which were kept. The authors claim that this "multiple model approach minimizes the problem of data overfitting that can result when a single 'best' model is used" (p. 127). Because the authors did not cross-validate, it is impossible to determine how much the model overfits the data, but it certainly seems that it does. As McCusker (2007) says,

> One could wonder whether the iterative classification tree methodology (a technique that involves repetitive sifting of risk factors) that was used to create the COVR ended up, in a sense, fitting the data in the development sample too specifically. Perhaps as a very carefully tailored garment will be expected to fit one individual perfectly but most other people not as well, so the algorithms of the COVR ought to be anticipated to classify other samples less exactly than they categorized the members of the development sample. (p. 682)

In November 2012, the journal *Perspectives on Psychological Science* released a

special issue dedicated to the lack of replicability in psychological research. The issue begins with a question from the editors: "Is there currently a crisis of confidence in psychological science reflecting an unprecedented level of doubt among practitioners about the reliability of research findings in the field?" (Pashler & Wagenmakers, 2012, p. 528); they immediately follow their question with an answer: "It would certainly appear that there is" (p. 528). The recent "replicability crisis" in psychology has given many reasons to question and doubt the results published in psychological journals. Because of its important role in reducing predictive error, cross-validation to construct predictive models can be expected to contribute to improved replicability. Our results, when compared with cross-validation studies of the COVR, provide an illustration of this assertion.

As important as cross-validation is for developing a model, the true test lies in how well the model does with an independent sample; that is, can the results be replicated? Therefore, the next step in validating the model is to assess the accuracy with an independent sample. To date, five studies have attempted to validate the COVR (Doyle, Shaw, Carter, & Dolan, 2010; McDermott et al., 2011; Monahan et al., 2005; Snowden, Gray, Taylor, & Fitzgerald, 2009; Sturup, Kristiansson, & Lindqvist, 2011). Table 8 displays a summary of the original study, the five validation studies, and the present study; many of the measures are more closely represented by the results found in the cross-validated models presented here. For instance, the AUC from the original study (Monahan et al., 2001) is .88 whereas the AUC for all validation studies are between .58 and .77 (the AUC for our final random forest model it is .75). The sensitivity in the original study is .60; in four of the five validation studies the sensitivity is below .50 (for our final random forest model it is .10). The positive predictive value for the construction study is .64 whereas four of the five validation studies have a PPV below .50 (the PPV for the final random forest model is .52).

The results from Table 8 suggest that the five validation studies did not replicate the findings of Monahan et al. (2001). Rather, the validation results give more evidence of the

results presented here; all measures provided in Table 8 are closer to the results from the final random forest model than the original study's model they are based on, aside from the sensitivity and specificity which is a direct result of the choice to not apply unequal costs in the final model. Thus, we conclude that the lack of cross-validation in a prediction model should also be reason for skepticism.

References

Banks, S., Robbins, P. C., Silver, E., Vesselinov, R., Steadman, H. J., Monahan, J., ... Roth, L. H. (2004). A multiple-models approach to violence risk assessment among people with mental disorder. *Criminal Justice and Behavior*, *31*, 324–340.

Berk, R. (2011). Asymmetric loss functions for forecasting in criminal justice settings. *Journal of Quantitative Criminology*, *27*, 107–123.

Berk, R. (2012). *Criminal justice forecasts of risk: A machine learning approach*. New York, NY: Springer.

Breiman, L. (1996). Bagging predictors. *Machine Learning*, *26*, 123–140.

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

Breiman, L., Friedman, J. H., Olshen, R. A., & Stone, C. J. (1984). *Classification and regression trees*. Belmont, CA: Wadsworth & Brooks.

Breiman, L., & Spector, P. (1992). Submodel selection and evaluation in regression. The X-random case. *International Statistical Review*, 291–319.

Doyle, M., Shaw, J., Carter, S., & Dolan, M. (2010). Investigating the validity of the Classification of Violence Risk in a UK sample. *International Journal of Forensic Mental Health*, *9*, 316–323.

Fernández-Delgado, M., Cernadas, E., Barro, S., & Amorim, D. (2014). Do we need hundreds of classifiers to solve real world classification problems? *The Journal of Machine Learning Research*, *15*, 3133–3181.

Gardner, W., Lidz, C. W., Mulvey, E. P., & Shaw, E. C. (1996). A comparison of actuarial methods for identifying repetitively violent patients with mental illnesses. *Law and Human Behavior*, *20*, 35–48.

Gini, C. (1912). *Variabilità e mutabilità: Contributo allo studio delle distribuzioni e delle relazioni statistiche [Variability and mutability: Contribution to the study of distributions and report statistics]*. Bologna, Italy: C. Cuppini.

Hare, R. D. (1980). A research scale for the assessment of psychopathy in criminal

populations. *Personality and Individual Differences*, *1*, 111–119.

Harris, G. T., & Rice, M. E. (2013). Bayes and base rates: What is an informative prior for actuarial violence risk assessment? *Behavioral Sciences and the Law*, *31*, 103-124.

Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The elements of statistical learning: Data mining, inference, and prediction (Second edition).* New York, NY: Springer.

James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). *An introduction to statistical learning.* New York, NY: Springer.

Kuhn, M., & Johnson, K. (2013). *Applied predictive modeling.* New York, NY: Springer.

McCusker, P. J. (2007). Issues regarding the clinical use of the Classification of Violence Risk (COVR) assessment instrument. *International Journal of Offender Therapy and Comparative Criminology*, *51*, 676–685.

McDermott, B. E., Dualan, I. V., & Scott, C. L. (2011). The predictive ability of the classification of violence risk (COVR) in a forensic psychiatric hospital. *Psychiatric Services*, *62*, 430–433.

Meehl, P. E., & Rosen, A. (1955). Antecedent probability and the efficiency of psychometric signs, patterns, or cutting scores. *Psychological Bulletin*, *52*, 194–215.

Monahan, J., Steadman, H. J., Appelbaum, P. S., Grisso, T., Mulvey, E. P., Roth, L. H., ... Silver, E. (2006). The Classification of Violence Risk. *Behavioral Sciences and the Law*, *24*, 721–730.

Monahan, J., Steadman, H. J., Robbins, P. C., Appelbaum, P. S., Banks, S., Grisso, T., ... Silver, E. (2005). An actuarial model of violence risk assessment for persons with mental disorders. *Psychiatric Services*, *56*, 810–815.

Monahan, J., Steadman, H. J., Robbins, P. C., Silver, E., Appelbaum, P. S., Grisso, T., ... Roth, L. H. (2000). Developing a clinically useful actuarial tool for assessing violence risk. *The British Journal of Psychiatry*, *176*, 312–319.

Monahan, J., Steadman, H. J., Silver, E., Appelbaum, P. S., Robbins, P. C., Mulvey, E. P., ... Banks, S. (2001). *Rethinking risk assessment: The MacArthur study of mental*

disorder and violence. New York, NY: Oxford University Press.

Mossman, D. (2006). Critique of pure risk assessment or, Kant meets *Tarasoff*. *University of Cincinnati Law Review*, *75*, 523–609.

Mossman, D. (2013). Evaluating risk assessments using receiver operating characteristic analysis: Rationale, advantages, insights, and limitations. *Behavioral Sciences and the Law*, *31*, 23–39.

Pashler, H., & Wagenmakers, E.-J. (2012). Editors' introduction to the special section on replicability in psychological science: A crisis of confidence? *Perspectives on Psychological Science*, *7*, 528–530.

Pollack, I., & Norman, D. A. (1964). A non-parametric analysis of recognition experiments. *Psychonomic Science*, *1*, 125–126.

R Core Team. (2014). *R: A language and environment for statistical computing (version 3.1.1).* Vienna, Austria. Retrieved from `http://www.R-project.org/`

Roberts, S., & Pashler, H. (2000). How persuasive is a good fit? A comment on theory testing. *Psychological Review*, *107*, 358–367.

Snowden, R. J., Gray, N. S., Taylor, J., & Fitzgerald, S. (2009). Assessing risk of future violence among forensic psychiatric inpatients with the classification of violence risk (COVR). *Psychiatric Services*, *60*, 1522–1526.

SPSS, Inc. (1993). *SPSS for Windows (Release 6.0).* Chicago, IL: SPSS, Inc.

Steadman, H. J., Silver, E., Monahan, J., Appelbaum, P. S., Robbins, P. C., Mulvey, E. P., ... Banks, S. (2000). A classification tree approach to the development of actuarial violence risk assessment tools. *Law and Human Behavior*, *24*, 83–100.

Sturup, J., Kristiansson, M., & Lindqvist, P. (2011). Violent behaviour by general psychiatric patients in Sweden: Validation of Classification of Violence Risk (COVR) software. *Psychiatry Research*, *188*, 161–165.

Vrieze, S. I., & Grove, W. M. (2008). Predicting sex offender recidivism. I. Correcting for item overselection and accuracy overestimation in scale development. II. Sampling

error-induced attenuation of predictive validity over base rate information. *Law and Human Behavior*, *32*, 266–278.

Table 1

The five risk categories for the Classification of Violence Risk (COVR) assessment tool along with point estimate risks (in probabilities) and respective confidence intervals (CI) (Monahan et al., 2006).

Category	Risk	Point Estimate	95% CI
5	Very High	.76	[.65, .86]
4	High	.56	[.46, .65]
3	Average	.26	[.20, .32]
2	Low	.08	[.05, .11]
1	Very Low	.01	[.00, .02]

Table 2

Pearson product-moment correlations of predictor variable with response variable, `Violence`*, in reanalyzed dataset (r) and reported correlations in Monahan et al. (2001) (r′).*

Variable	r	r'	Variable	r	r'
Age	−.07	−.07	HeadInj	.03	.06
BISnp	.05	.05	LegalStatus	.11	.11
BPRSa	−.08	−.08	NASb	.17	.16
BPRSh	.08	.08	NegRel	.05	.06
BPRSt	−.04	−.04	PCL	.26	.26
ChildAbuse	.14	.14	PCS	.03	.03
Consc	.09	.10	PriorArr	.24	.24
DadArr	.15	.15	PropCrime	.11	.11
DadDrug	.14	.16	RecViol2	.14	.14
DrugAbuse	.16	.17	Schiz	−.12	−.12
Emp	−.05	−.05	SNMHP	−.10	−.10
FantEsc	.13	.13	SubAbuse	.18	.18
FantSing	.10	.10	Suicide	−.01	−.01
FantTarg	.12	.12	tco	−.09	−.10
Function	−.01	−.01	Threats	.06	.06
GranDel	−.01	−.01			

Table 3

Predicting violence with a random forest model using the testing data. If a patient had a predicted probability greater than .50, a prediction of violence was made; otherwise a prediction of no violence was made.

		Violence		
		Yes (A)	No ($\bar{A}$)	Row Totals
Prediction	Yes (B)	1	1	2
	No ($\bar{B}$)	56	224	280
	Column Totals	57	225	282

Table 4

Predicting violence with a random forest model. If a patient had a predicted probability greater than twice the base rate (.37), a prediction of violence was made; otherwise a prediction of no violence was made.

		Violence		
		Yes (A)	No ($\bar{A}$)	Row Totals
Prediction	Yes (B)	15	8	23
	No ($\bar{B}$)	42	217	259
	Column Totals	57	225	282

Table 5

Predicting violence with a random forest model on the entire sample. The top of the table uses a cutscore of .50: If a patient had a predicted probability greater than twice the base rate (.50), a prediction of violence was made; otherwise a prediction of no violence was made. The bottom table uses a cutscore of .37.

.50 cutscore				
		Violence		
		Yes (A)	No ($\bar{A}$)	Row Totals
Prediction	Yes (B)	10	14	24
	No ($\bar{B}$)	166	749	915
	Column Totals	176	763	939

.37 cutscore				
		Violence		
		Yes (A)	No ($\bar{A}$)	Row Totals
Prediction	Yes (B)	13	15	28
	No ($\bar{B}$)	163	748	911
	Column Totals	176	763	939

Table 6

Predicting violence with the final random forest model with equal costs.

		Violence		
		Yes (A)	No ($\bar{A}$)	Row Totals
Prediction	Yes (B)	17	16	33
	No ($\bar{B}$)	159	747	906
	Column Totals	176	763	939

Table 7

COVR risk groups from Monahan et al. (2001, cf. Table 6.7, p. 125).

		Violence			
		Yes	No	Row Totals	Proportion Violent
Risk Group	Very High	48	15	63	.76
	High	57	45	102	.56
	Average	48	135	183	.26
	Low	19	229	248	.08
	Very Low	4	339	343	.01
	Column Totals	176	763	939	.19

Table 8

Summary of COVR studies.

			Study					
	Original	Present	1	2	3	4	5	Average
Violence Base Rate	.19	.19	.23	.52	.24	.15	.06	.16
Violence Selection Ratio	.17	.04	.35	.21	.05	.14	.03	.13
Reported AUC	.88	.75	.70	.73	.58	.73	.77	—
Sensitivity	.60	.10	.75	.37	.00	.41	.21	.39
Specificity	.92	.98	.77	.96	.93	.91	.98	.92
PPV	.64	.52	.49	.91	.00	.45	.36	.48
NPV	.91	.82	.91	.59	.75	.90	.95	.89
Misclassification Error	.14	.19	.24	.35	.29	.16	.07	.17

Note. Aside from *Reported AUC*, all statistics are calculated using data in the form of 2×2 contingency tables where a COVR score of 4 or 5 leads to a prediction of violence and all lower scores do not.

Original: Monahan et al. (2001); Present: Results are based on the final random forest model with equal costs (see Table 6); Study 1: Monahan et al. (2005); Study 2: Snowden et al. (2009); Study 3: Doyle et al. (2010); Study 4: McDermott et al. (2011); Study 5: Sturup et al. (2011); Average: Weighted average of the five validation studies.

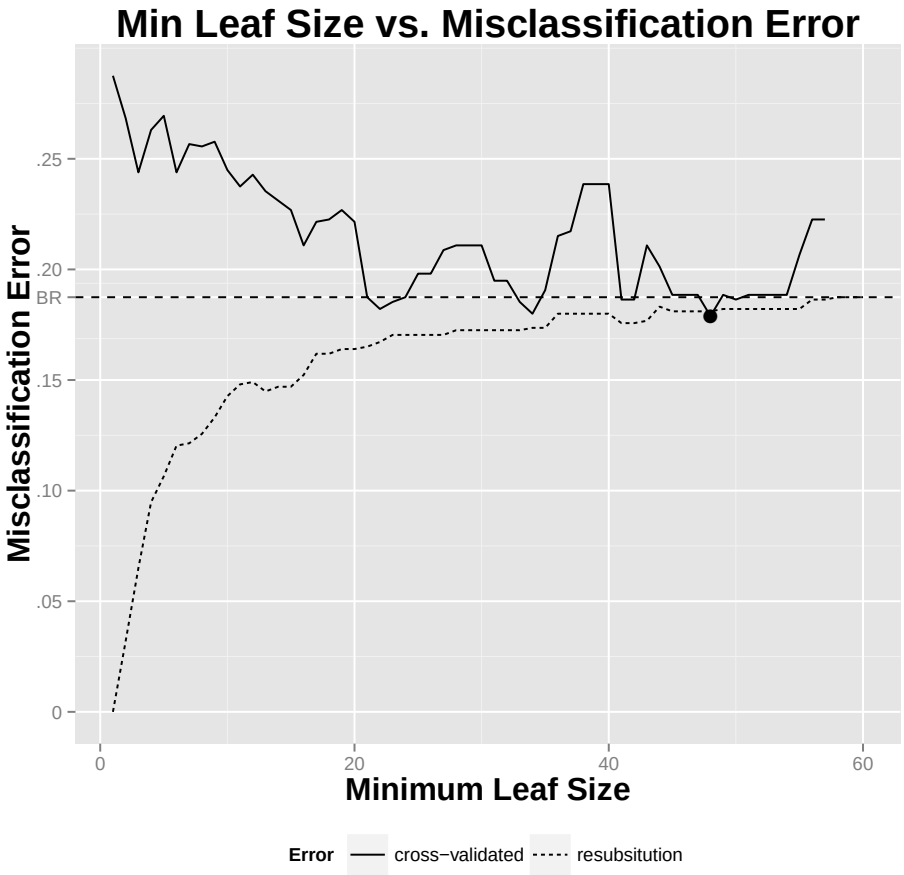


Figure 1. Determining the minimum leaf size for a classification tree with leave-one-out cross-validation error. The dotted line displays the resubstitution error; the solid line, the cross-validated error. The minimum leaf size was determined to be 48.

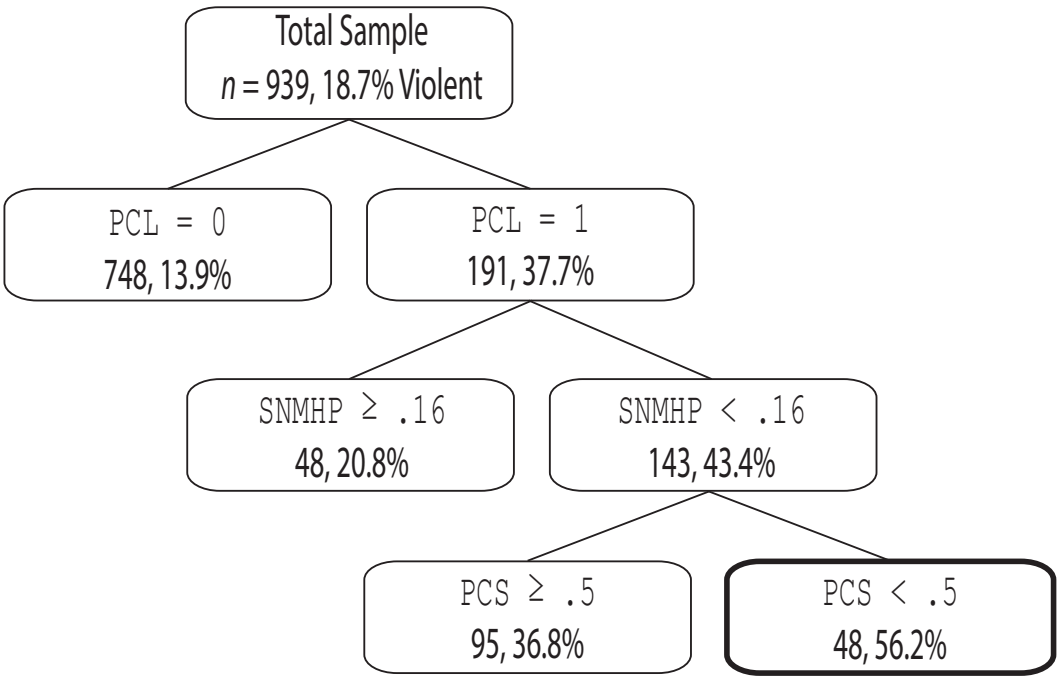


Figure 2. Classification tree with a minimum leaf size of 48 and equal costs. The bold box represents the node where predictions of violence are made. Note that PCL is the Psychopathy Checklist (Hare, 1980); SNMHP is the proportion of social network members who are also mental health professionals; and PCS is the Perceived Coercion Scale (Monahan et al., 2001).

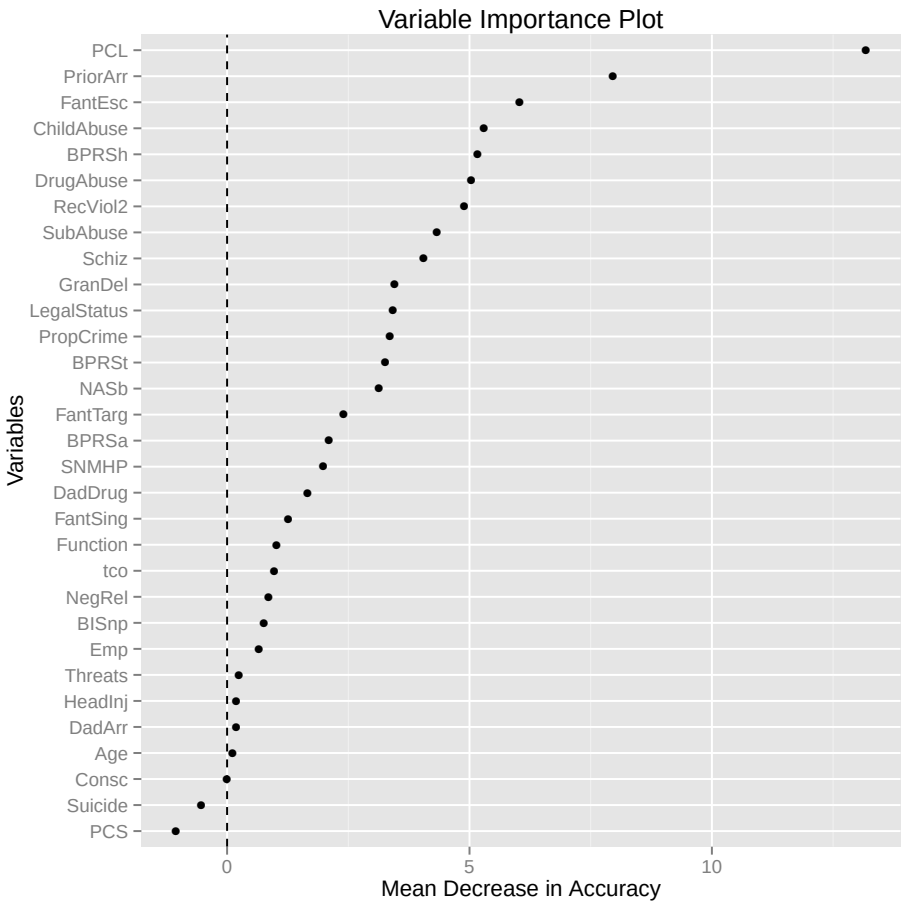


Figure 3. Measure of importance for each of the thirty-one variables. The importance measure is the average of the differences in out-of-bag error before and after permutation across all trees.

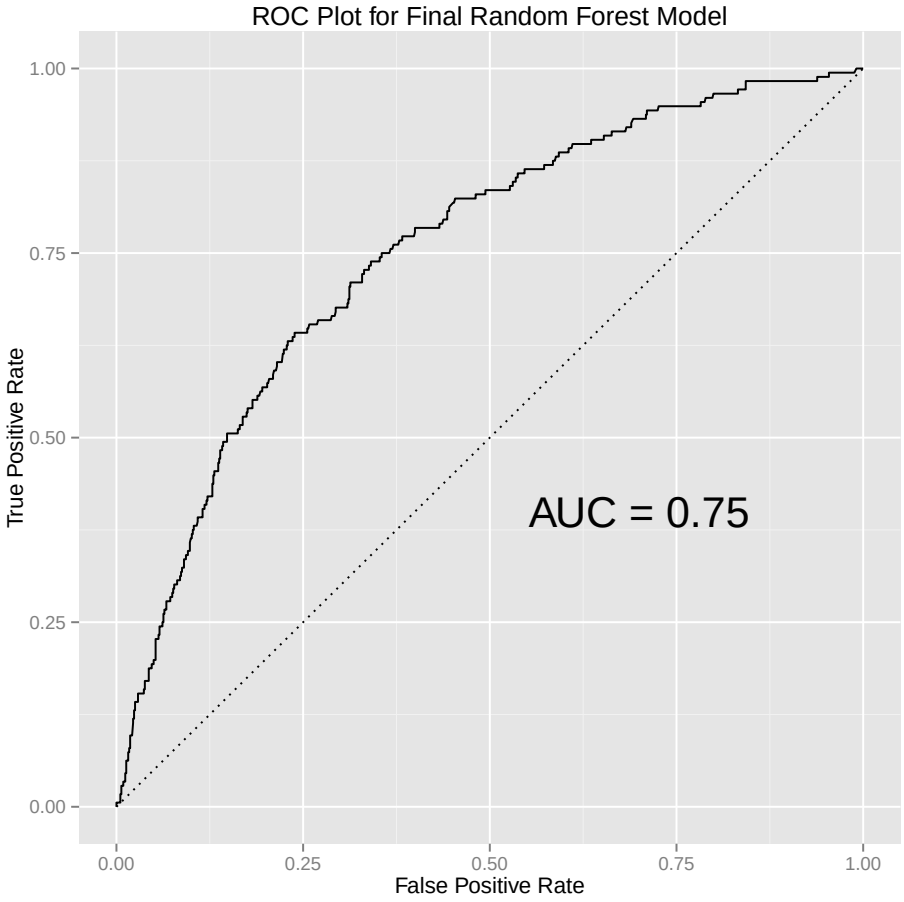


Figure 4. ROC plot for the final random forest model.

Appendix A

Definitions

Frequencies and Probabilities Associated with Contingency Tables

Suppose a diagnostic test is designed to determine whether a person has "it," whatever "it" may be; for example, in predicting violence, the test indicates whether the person will or will not be violent. Let B denote the event that the test is positive indicating the person has "it," and $\bar{B}$, the event that the test is negative indicating that the person does not have "it." Now, consider the events of whether a person truly has "it" or truly does not have "it" and denote these two events as A and $\bar{A}$, respectively. The events B and $\bar{B}$ will be called the *diagnostic test results*, and the events A and $\bar{A}$ the *states of nature*.

Given the diagnostic test result and state of nature, a 2×2 contingency table can be constructed, as shown in Table A1. This table provides the frequencies of marginal events (for example, n_B is the number of people who test positive), or of joint events (for example, n_{BA} is the number of people who have "it" and tested positive). In terms of violence prediction, n_B is the number predicted to be violent and n_{BA} is the number predicted to be and who were violent. The frequencies within the table have familiar names: n_{BA} is the number of *true positives*, $n_{B\bar{A}}$ is the number of *false positives*, $n_{\bar{B}A}$ is the number of *false negatives*, and $n_{\bar{B}\bar{A}}$ is the number of *true negatives*. Of particular importance are the marginal frequencies: n_A, representing the *base frequency* for those who have "it," and $n_{\bar{A}}$, the base frequency for those who do not. The marginal frequencies, n_B and $n_{\bar{B}}$, are the base frequencies for positive and negative diagnostic test outcomes, respectively, and are often called *selection frequencies*.

In addition to frequencies, various marginal, joint, and conditional probabilities can be defined. For example, $P(A) = \frac{n_A}{n}$; $P(A \cap B) = \frac{n_{BA}}{n}$; $P(A|B) = \frac{n_{BA}}{n_B}$; $P(B|A) = \frac{n_{BA}}{n_A}$; and so forth. These conditional probabilities are of general interest, and again it is worth noting some of their names. Conditionalizing on the state of nature gives the following: $P(B|A) = \frac{n_{BA}}{n_A}$ is the *sensitivity* or *true positive rate*; $P(B|\bar{A}) = \frac{n_{B\bar{A}}}{n_{\bar{A}}}$ is the *false positive*

rate; $P(\bar{B}|A) = \frac{n_{\bar{B}A}}{n_A}$ $(= 1 - \text{sensitivity})$ is the *false negative rate*; and $P(\bar{B}|\bar{A}) = \frac{n_{\bar{B}\bar{A}}}{n_{\bar{A}}}$ $(= 1 - \text{false positive rate})$ is the *specificity* or *true negative rate*. Conditionalizing on the diagnostic test result, $P(A|B) = \frac{n_{BA}}{n_B}$ is called the *positive predictive value* (PPV); $P(\bar{A}|\bar{B}) = \frac{n_{\bar{B}\bar{A}}}{n_{\bar{B}}}$ is the *negative predictive value* (NPV). The marginal probabilities represent the *base rates* for those who have "it" $(P(A))$ and those who do not $(P(\bar{A}))$ (also called *prior probabilities*); and those who are predicted to have "it" $(P(B))$ and those who are not $(P(\bar{B}))$ (also called *selection rates*). Finally, the *accuracy*, or *proportion correct*, is $P(B|A)P(A) + P(\bar{A}|\bar{A})P(\bar{A})$ which is equal to the sum of the diagonal entries divided by the sample size: $\frac{n_{BA}+n_{\bar{A}\bar{B}}}{n}$.

Table A1

A general 2×2 *contingency table.*

		State of Nature		
		A (positive)	$\bar{A}$ (negative)	Totals
Diagnostic	B (positive)	n_{BA}	$n_{B\bar{A}}$	n_B
Test Result	$\bar{B}$ (negative)	$n_{\bar{B}A}$	$n_{\bar{B}\bar{A}}$	$n_{\bar{B}}$
	Totals	n_A	$n_{\bar{A}}$	n

Base-rate prediction, or naïve prediction, is to say that everyone has "it" or nobody has "it," depending on the larger base rate (clearly the base rates need to be known or confidently estimated). For example, if $P(A) \leq \frac{1}{2}$, then prediction using the base rates is to say that nobody has "it" because the accuracy will be greater than one-half. The term "clinical efficiency" (Meehl & Rosen, 1955) refers to a test outperforming prediction using the base rates. A general condition for when a test outperforms base-rate predictions (assuming that $P(A) \leq \frac{1}{2}$) is

$$P(B|A)P(A) + P(\bar{A}|\bar{A})P(\bar{A}) > P(\bar{A});$$

that is, the accuracy is greater than the larger base-rate.

A 2×2 contingency table like that in Table A1 may represent the results of a diagnostic test at a specific cutscore. Often, a number of cutscores might be definable for a test; a common procedure to summarize the results at a number of cutscores is taken from signal detection theory.

Signal Detection Theory

Signal Detection Theory (SDT) is a framework for assessing uncertainty in decision making; it is also commonly used in evaluating diagnostic measures in psychology and medicine, among other areas. SDT allows the researcher to quantify the ability of a diagnostic tool to distinguish meaningful patterns and known processes (called the signal) from random patterns or chance processes (called the noise). Based on the given data, three important parameters are estimated: the strength of the signal relative to the noise, called the *discriminability index* (denoted d'); the decision criterion of the diagnostic measure, called the *cutscore* (denoted x_c); and an indication of *response bias* (denoted β).

Given that the signal is truly present, a correct decision (i.e., stating that the signal is present) is called a *hit* (or true positive); an incorrect decision (i.e., stating that noise is present) is called a *miss* (or false negative). When noise is truly present, a correct decision (i.e., stating that noise is present) is called a *correct rejection* (or true negative); an incorrect decision (i.e., stating that the signal is present) is called a *false alarm* (or false positive). Similar to our earlier discussion, these can be framed in terms of rates: the *hit rate* (true positive rate; sensitivity) is the number of correct responses given the presence of the signal; the *false alarm rate* (false positive rate; $1 -$ specificity) is the number of incorrect responses given the presence of noise. In SDT, these are the two rates of interest; the other two are easily calculated from them and therefore provide no additional information.

Two distributions can be constructed; one represents the signal, the second represents the noise (see Figure A1). To simplify our discussion, they are considered to be normal distributions with equal variances but differing means. Without loss of generality, the

mean of the signal distribution is assumed greater than that of the noise distribution. The difference between the two means of the distributions is indicative of the strength of the signal in relation to the noise; this is the discriminability index, d'. Based on the hit and false alarm rates, d' is calculated as $\Phi^{-1}(\text{hit rate}) - \Phi^{-1}(\text{false alarm rate})$, where $\Phi(\cdot)$ represents the cumulative standard normal distribution. Note that d' requires several parametric assumptions regarding normal distributions; there are nonparametric alternatives, but these are not discussed here (e.g., the measure commonly denoted as A'; Pollack & Norman, 1964). A d' of 0 indicates that the diagnostic measure cannot distinguish signal from noise. The criterion point distinguishing a positive response—with respect to the presence of the signal—from a negative response is the cutscore, x_c. Given x_c, the area under the noise distribution to the right of x_c represents the probability of a false alarm (i.e., the false alarm rate); the area under the signal distribution to the left of x_c represents the probability of a miss (i.e., $1 -$ hit rate). Another detail gathered from x_c is the density (or height) of the two distributions corresponding to x_c. These two densities represent the likelihoods of signal (the height of the signal distribution at x_c) and noise (the height of the noise distribution at x_c). The response bias of the test, β, is the ratio of these two: $\frac{\text{signal likelihood}}{\text{noise likelihood}}$. If $\beta > 1$, the diagnostic test favors (i.e., is biased toward) a positive decision (stating that the signal is present) over a negative one (stating that the signal is absent); if $\beta < 1$, the diagnostic test favors a negative decision over a positive one; if $\beta = 1$, there is no response bias as neither decision is favored over the other. Another way of thinking about β is that when an observation x is greater than x_c, the ratio of the two likelihoods at the point x is greater than β. More realistically, the two distributions can differ not only in their means but in their variances as well; further, the prior probability of the noise distribution is not likely to be equal to the prior for the signal distribution.

ROC Plots

There are numerous (possibly infinitely many) different choices for a cutscore, x_c. A unique hit and false alarm rate exists at each cutscore; these can be plotted in two dimensions with the hit rate (often labeled sensitivity or true positive rate) along the vertical axis and the false alarm rate (often labeled 1 − specificity or false positive rate) along the horizontal axis. This plot is referred to as a *receiver operating characteristic* (ROC) curve (see Figure A2).

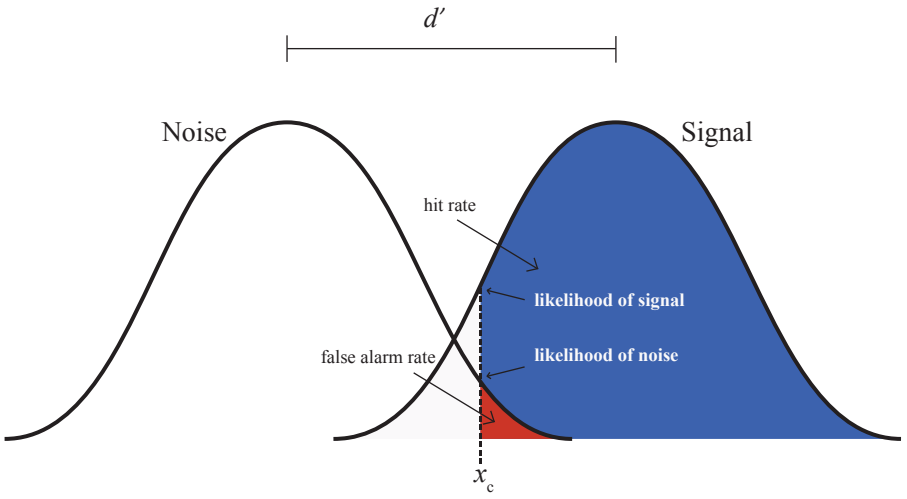


Figure A1. Two normal distributions representing the distribution of the signal (right) and the noise (left).

The minimum along each axis is 0; the maximum is 1. The point $(0, 0)$ corresponds to hit and false alarm rates of 0; that is, a cutscore larger than any attainable score. This is equivalent to stating that the signal is always absent or predicting nobody to have "it." The other extreme is at $(1, 1)$ and corresponds to hit and false alarm rates of 1; that is, it is representative of a cutscore that is smaller than any attainable score. In this situation the signal is always stated to be present; that is, predicting everyone to have "it." Neither of the two extreme situations are useful—a diagnostic test is unnecessary if one is consistently predicting one way or the other. What is of interest are the values in between that

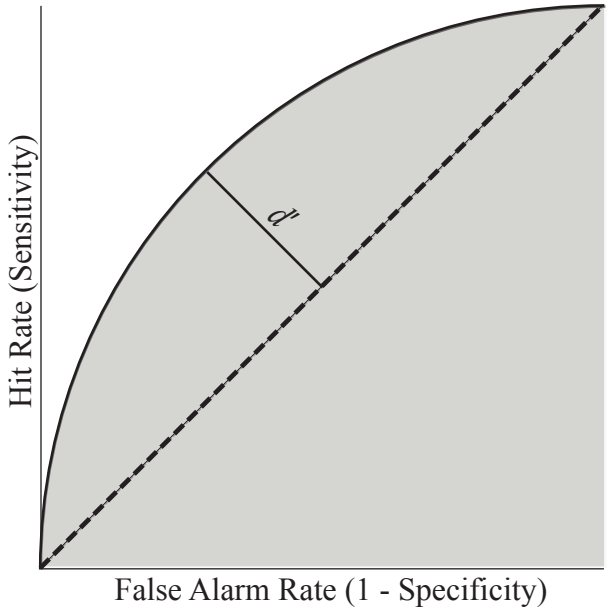


Figure A2. A receiver operating characteristic (ROC) plot. The points along the curve represent the different hit and false positive rate combinations achieved using different cutscores. The shaded area represents the area under the curve.

correspond to meaningful cutscores. By moving along the curve a cutscore can be chosen that corresponds to different hit and false alarm rates.

A 45° line is often placed on the plot, running from the point $(0, 0)$ to $(1, 1)$. This line is called the *line of no discrimination* and represents a diagnostic test performing no better than chance; ideally all points on the ROC curve are above this line. The discriminability index, d', is represented by the distance of the curve to the line of discrimination; specifically, it is the orthogonal distance measured at the point $(.5, .5)$ to the curve. As the two distributions are further spread apart, this distance becomes larger.

As a measure of diagnostic accuracy, the *area under the ROC curve* (AUC; also called the *concordance index*) is commonly used. The total area ranges from 0 to 1; however, an AUC less than .50 is infrequently found because it represents a test that performs worse than chance. The AUC can be interpreted as the probability that a randomly selected individual who has "it" will have a larger score on the diagnostic test than a randomly

selected individual who does not have "it." There are ways to test whether two ROC curves are different from each other (e.g., whether one diagnostic test outperforms another), or when an ROC curve is different from a diagnostic test that is no better than chance (i.e., an ROC curve equal to the line of no discrimination). As a final point, it is noted that the ROC curve is not influenced by the base rates.

Appendix B

Bias-Variance Trade-Off

Let $x = (x_1, x_2, \ldots, x_p)$ be a collection of p predictor variables and let Y be a response variable. Suppose

$$Y = f(x) + \varepsilon,$$

where ε is an error random variable with $\mathbb{E}(\varepsilon|X) = 0$ and $\mathbb{V}(\varepsilon|X) = \sigma_\varepsilon^2$; that is, there exists a function $f(\cdot)$ for modeling the response variable, plus unknown error ε. Note that all expectations and variances are conditioned on the data, X, but this notation is suppressed for simplicity.

A *loss function* measures the error between the estimated function, $\hat{f}(x)$, and the true one, $f(x)$; a common type of loss function is the *squared error loss*,

$$\mathcal{L}_2(f(x), \hat{f}(x)) = \left(f(x) - \hat{f}(x)\right)^2.$$

The quality of an estimator can be quantified by taking the expectation of the loss function; this is called the *risk function* and defined as

$$\mathcal{R}(f(x), \hat{f}(x)) = \mathbb{E}(\mathcal{L}(f(x), \hat{f}(x))).$$

The interest in prediction reduces to estimating the function, $f(x)$, given the observed data, $(\mathbf{X}, \mathbf{y})$, where $\mathbf{X}$ is an $n \times p$ matrix of n observations on p predictors and $\mathbf{y}$ is an $n \times 1$ vector containing the outcome; the estimated function will be denoted as $\hat{\mathbf{y}} \equiv \hat{f}(x) \equiv \hat{f}(x|\mathbf{X})$. Given the observed data, $(\mathbf{X}, \mathbf{y})$, the estimated risk function for the estimator $\hat{\mathbf{y}}$ using the squared loss function, is

$$\frac{1}{n}\|\mathbf{y} - \hat{\mathbf{y}}\|_2^2 = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2,$$

where $\|\cdot\|_2$ is the Euclidean, or ℓ_2, norm (for simplicity, the subscript 2 is dropped); this is commonly known as the *mean squared error*.

As presented in the text, the risk function for the squared error,

$$\mathcal{R}(y, \hat{y}) = \mathbb{E}\left(\mathcal{L}_2(y, \hat{y})\right) = \mathbb{E}\left[(y - \hat{y})^2\right],$$

can be decomposed into two parts; here we show how this is done:

$$\mathbb{E}\left[(y - f(x) + f(x) - \hat{y})^2\right] = \mathbb{E}\left[(y - f(x))^2\right] + \mathbb{E}\left[(f(x) - \hat{y})^2\right] + 2\mathbb{E}\left[(y - f(x))(f(x) - \hat{y})\right].$$

The first term reduces to

$$\mathbb{E}\left[(y - f(x))^2\right] = \mathbb{E}\left[(f(x) + \varepsilon - f(x))^2\right] = \mathbb{E}\left(\varepsilon^2\right) = \mathbb{V}(\varepsilon).$$

The last term (disregarding the 2) reduces to

$$\begin{aligned}
\mathbb{E}\left[(y - f(x))(f(x) - \hat{y})\right] &= f(x)\mathbb{E}(y) - f(x)^2 + \mathbb{E}(y\hat{y}) - f(x)\mathbb{E}(\hat{y}) \\
&= f(x)^2 - f(x)^2 + \mathbb{E}\left((f(x) + \varepsilon)\hat{y}\right) - f(x)\mathbb{E}(\hat{y}) \\
&= \mathbb{E}\left(f(x)\hat{y} + \varepsilon\hat{y})\right) - f(x)\mathbb{E}(\hat{y}) \\
&= f(x)\mathbb{E}(\hat{y}) - f(x)\mathbb{E}(\hat{y}) = 0.
\end{aligned}$$

Thus,

$$\mathcal{R}(y, \hat{y}) = \mathbb{E}\left[(f(x) - \hat{y})^2\right] + \mathbb{V}(\varepsilon). \tag{B1}$$

As mentioned in the text, the first term, $\mathbb{E}\left[(f(x) - \hat{y})^2 | X\right]$, is the *reducible error*. The better the estimator $\hat{y}$, the closer the reducible error is to zero; the reducible error is equal to zero when $\hat{y} = f(x)$. The second term in Equation (B1), $\mathbb{V}(\boldsymbol{\varepsilon}|X)$, is the *irreducible error* and represents a lower bound for the risk function. Even when $f(x)$ is perfectly estimated (that is, $\hat{y} = f(x)$), $\mathcal{R}(y, \hat{y}) = \mathbb{V}(\boldsymbol{\varepsilon}|X) > 0$.

The reducible error can be further decomposed as follows:

$$\begin{aligned}
\mathbb{E}\left[(f(x) - \hat{y})^2\right] &= \mathbb{E}\left[(f(x) - \mathbb{E}(\hat{y}) + \mathbb{E}(\hat{y}) - \hat{y})^2\right] \\
&= \mathbb{E}\left[(f(x) - \mathbb{E}(\hat{y}))^2\right] + \mathbb{E}\left[(\mathbb{E}(\hat{y}) - \hat{y})^2\right] + 2\mathbb{E}\left[(f(x) - \mathbb{E}(\hat{y}))(\mathbb{E}(\hat{y}) - \hat{y})\right] \\
&= \mathbb{E}\left[\text{bias}(\hat{y})^2\right] + \mathbb{V}(\hat{y}) + 2\left[\mathbb{E}\left(f(x)\mathbb{E}(\hat{y})\right) - \mathbb{E}(f(x)\hat{y}) - \mathbb{E}(\mathbb{E}(\hat{y})\mathbb{E}(\hat{y})) + \mathbb{E}\left(\mathbb{E}(\hat{y})\hat{y}\right)\right] \\
&= \mathbb{E}\left[\text{bias}(\hat{y})^2\right] + \mathbb{V}(\hat{y}) + 2\left[f(x)\mathbb{E}\left(\hat{y}\right) - f(x)\mathbb{E}(\hat{y}) - \mathbb{E}(\hat{y})^2 + \mathbb{E}(\hat{y})^2\right] \\
&= \text{bias}(\hat{y})^2 + \mathbb{V}(\hat{y}).
\end{aligned}$$

Thus,

$$\mathcal{R}(y, \hat{y}) = \left(\text{bias}(\hat{y})\right)^2 + \mathbb{V}(\hat{y}) + \mathbb{V}(\varepsilon). \tag{B2}$$

The first term is the squared bias associated with the estimator for the function. The bias measures how much, on average, the estimated function over- or underestimates the true function, $f(x)$; an unbiased estimator has a squared bias equal to zero. The second term is the variance of the estimator and represents how much, on average, the estimated function deviates from the true function. Thus, reducible error depends on both the variance and the bias of the estimator.

As alluded to in the text, as some models become more complex (for example, reducing the minimum leaf size in a decision tree), the training error tends to decrease whereas the testing error increases. As Equation (B2) suggests, this increase in testing error can be attributed to either an increase in the variance of the predictor or an increase in the squared bias, or both. Higher variance implies that a change in a set of observations (for example, applying a model to a new data set) can lead to dramatic changes in the model error; higher bias implies that the model assumes a less complex relationship than is true (for example, assuming a linear relationship when the true relationship is nonlinear). An increase in model complexity generally leads to an increase in the variance and a decrease in the squared bias (Hastie et al., 2009, p. 38). The test error may initially decrease as the model becomes more complex but it eventually increases; in contrast, the training error always decreases because the more complex model fits the data more closely. Fitting too complex of a model leads to increased test error and can be thought of as overfitting. Similarly, fitting too simple of a model that leads to increased test error can be considered underfitting. Ideally, the researcher wants to find the minimal point, or equilibrium, where the model neither under- or overfits the data; this minimum can be estimated using cross-validation.

As an illustration, consider the function

$$f(x) = 5 - 3x + 2x^2 - 7x^3 + \varepsilon;$$

one-hundred observations were simulated from the above function, where $\mathbb{V}(\varepsilon|X) = 1$, and plotted in Figure B1; the black curve represents the true cubic function. Figure B2 plots

the linear, quadratic, cubic, and quartic least-squares line fit to the data.

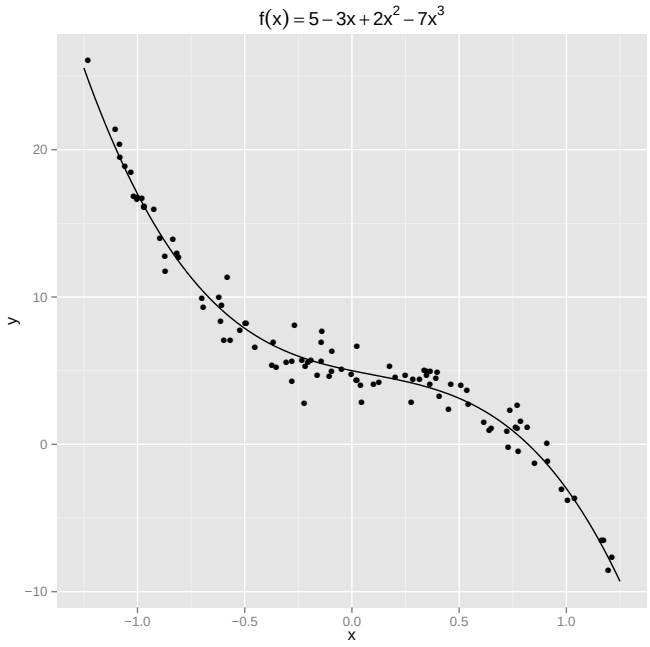


Figure B1. Scatter plot of data simulated from $f(x) = 5 - 3x + 2x^2 - 7x^3 + \varepsilon$ where $\mathbb{V}(\varepsilon|X) = 1$. The black line represents the true relationship.

Next, test error for estimating the function using different polynomial regression models is estimated using leave-one-out cross-validation; in Figure B3, this is plotted against the model complexity (that is, the order of the polynomial which ranges from first-order up to twelfth-order). The horizontal dashed line represents $\mathbb{V}(\varepsilon|X) = 1$, the irreducible error. The training error (blue line) approaches this line quickly and eventually decreases to 0. The testing error is at a minimum for the cubic model, as expected. The test error for the fourth-order model is only slightly larger; but moving to a tenth- or higher-order model leads to large increases in the testing error.

The test error can be split into the squared bias and the variance of the model and this is shown in Figure B4. The squared bias is at a maximum for the linear model and is still large for the quadratic model but quickly decreases for the cubic model. Further

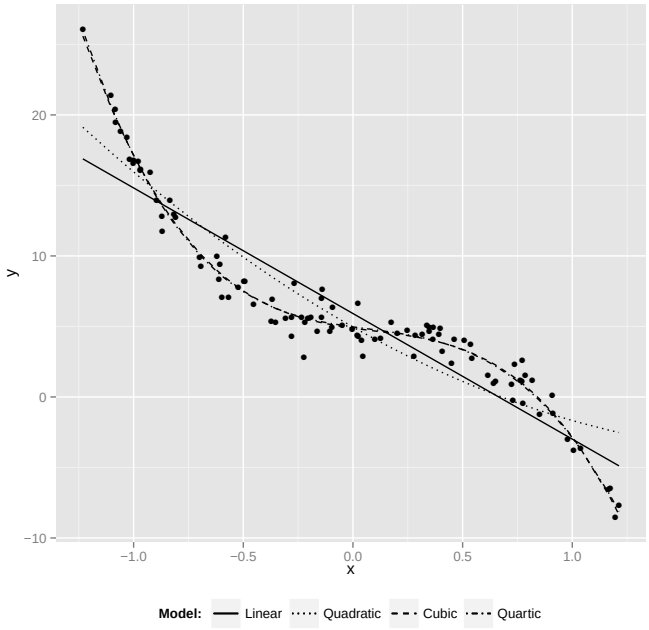


Figure B2. Scatter plot of data simulated from $f(x) = 5 - 3x + 2x^2 - 7x^3 + \varepsilon$ where $\mathbb{V}(\varepsilon|X) = 1$. The lines represent the different least squares fits.

increasing the complexity of the model leads to near-zero bias. In contrast, the variance is near zero for the linear model and remains near zero until the tenth-order model when the variance rapidly increases with greater complexity.

It is clear that the linear and quadratic models underfit the data and the quartic- and higher-order models overfit the data (although this overfitting is hardly noticeable until the degree is greater than or equal to 10). This simple example illustrates the importance of the bias-variance trade-off; in building the best predictive model, often one must sacrifice an increase in bias for less variance to minimize the overall test error.

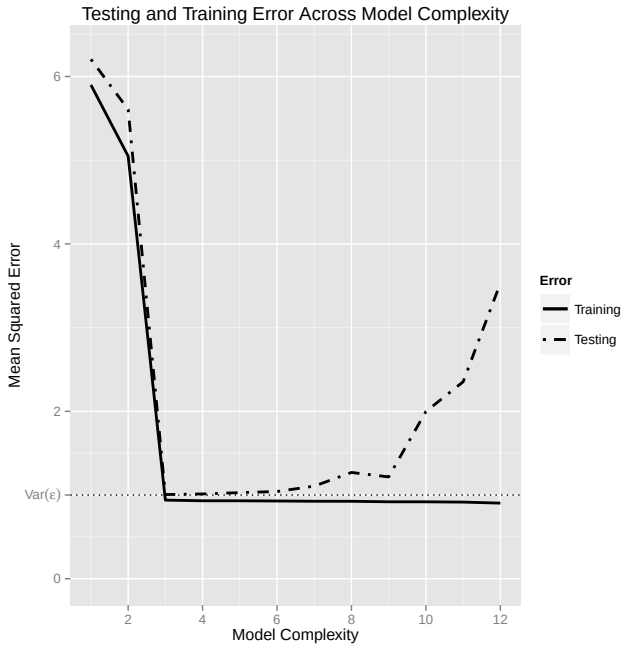


Figure B3. Training and testing error plotted against different levels of model complexity (that is, different order polynomial regression models).

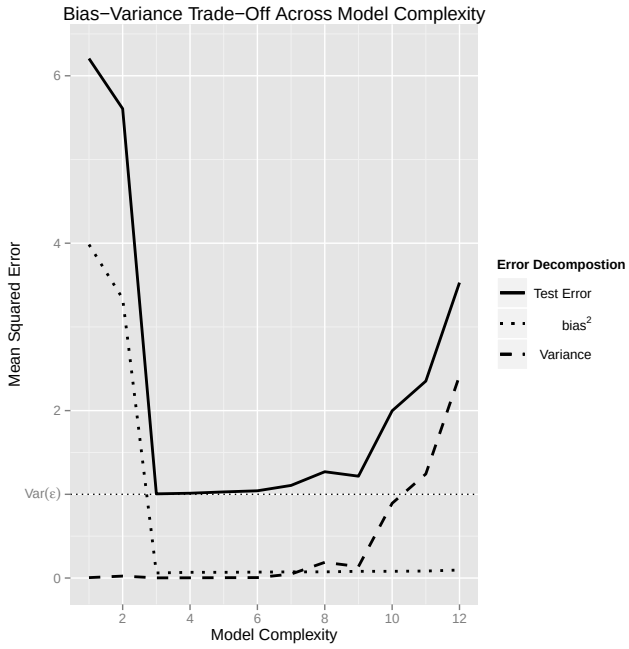


Figure B4. Testing error broken down into its two main components, $\Big(\text{bias}(\hat{f}(x))\Big)^2$ and $\mathbb{V}(\hat{f}(x)|X)$, across different levels of model complexity.

Supplementary R Syntax for *The Lack of Cross-Validation Can Lead to Inflated Results and Spurious Conclusions: A Re-Analysis of the MacArthur Violence Risk Assessment Study*

Supplementary R Syntax — Data Preprocessing

Variable Description	Variable Coding	Original Variables Used
Response Variable		
Violence	`Violence`	`F12VIOL`
Predictor Variables		
Age	`Age`	`AGE`
BIS Non-Planning Subscale	`BISnp`	`BISPLN`
BPRS Activation Subscale	`BPRSa`	`OACTV`
BPRS Hostile-Suspiciousness Subscale	`BPRSh`	`OHOST`
BPRS Total Score	`BPRSt`	`OBPRS`
Child Abuse Seriousness	`ChildAbuse`	`Q5.5.1`, `Q5.5.2`, `Q5.5.3`, `Q5.5.4`, `Q5.5.5`, `Q5.5.6`
Loss of Consciousness (Head Injury)	`Consc`	`NEU2A`
Father Arrest History	`DadArr`	`Q5.20A`, `Q5.20B`
Father's Drug Use	`DadDrug`	`Q5.19A`, `Q5.19B`
Drug Abuse (DSM-III-R)	`DrugAbuse`	`DSM16A`, `DSM16B`, `DSM17A`, `DSM17B`
Employed Prior to Hospitalization	`Emp`	`Q4.4`
Violent Fantasies: Escalating Seriousness	`FantEsc`	`Q7.1`, `Q7.7`
Violent Fantasies: Single Target Focus	`FantSing`	`Q7.1`, `Q7.6`
Violent Fantasies: Target Present	`FantTarg`	`Q7.1`, `Q7.8`
Level of Functioning	`Function`	`Q9.1`, `Q9.2`, `Q9.3`, `Q9.4`, `Q9.5`, `Q9.6`
Grandiose Delusions	`GranDel`	`DEL03.1`
Previous Head Injuries	`HeadInj`	`NEU4B.1`, `NEU4B.2`, `NEU4B.3`, `NEU4B.4`, `NEU4B.5`, `NEU4B.6`, `NEU4B.7`, `NEU4B.8`, `NEU4B.9`
Legal Status for Hospitalization	`LegalStatus`	`LEGALR`
Novaco Anger Scale Behavioral Subscale	`NASb`	`NASBEH`
Number of Negative Relationships	`NegRel`	`Q10.10N`, `Q10.11N`, `Q10.12N`, `Q10.13N`
Psychopathy Checklist: Screening Version	`PCL`	`PCLTOT`
MacArthur Perceived Coercion Scale	`PCS`	`Q1.8`, `Q1.11`, `Q1.14`, `Q1.21`, `Q1.22`
Prior Arrest Frequency	`PriorArr`	`FREQARR`
Property Crime Arrest	`PropCrime`	`PROPARR`
Violent Before Hospitalized	`RecViol2`	`VIOL`
Schizophrenic	`Schiz`	`DSM2A`, `DSM5A`
Proportion of Social Network are Mental Health Professionals	`SNMHP`	`SNMHP`
Substance Abuse	`SubAbuse`	`DSM14A`, `DSM14B`, `DSM15A` `DSM15B` `DSM16A`, `DSM16B`, `DSM17A`, `DSM17B`
Admission Reason: Suicide	`Suicide`	`QREAS.02`
Threat/Control Override Symptoms	`tco`	`K1.1`, `K1.3`, `K2.1`, `K2.3`, `K3.1`, `K3.3`, `K4.1`, `K4.3`, `K8.1`, `K8.3`, `K9.1`, `K9.3`, `K10.1`, `K10.3`, `K12.1`, `K12.3`
Threats at Admission	`Threats`	`QREAS.20`, `QREAS.21`

Table 1: Variables used in analyses, from the MacArthur Violence Risk Assessment Study (Monahan et al., 2001).

Table 1 above displays the variables included in the initial analyses. The first column is a brief description; the second column is the coding used in the analysis; the third column consists of the variable codes used in the original VRAS dataset. All variables come from the SPSS file `baseline.sav` except `F12VIOL` and `PCLTOT` that were from the SPSS file `follow_up_subjects.sav`.

We begin by loading the necessary packages; this assumes the packages are installed. If not, use `install.packages()` (e.g., `install.packages("dplyr")` installs the `dplyr` package).

```
# for reading in SPSS files
require(Hmisc)
# for data frame manipulation
library(dplyr)
# for creating data tables
library(data.table)
```

Data Preprocessing

First, we bring in the two data files and sort in order of `STUDYID` and then merge them into one master file data table called `COVR`.

```
# read in data files
COVR1 = data.table(spss.get('baseline.sav'))
COVR2 = data.table(spss.get('follow_up_subjects.sav'))

# sort data, select desired variables, and merge into one master data table
COVR1 = COVR1 %>%
     select(STUDYID, BISNPLN, OACTV, OHOST, OBPRS, LEGALR, SNMHP, NOVBEH, NEU2A,
            NEU4B.1:NEU4B.9, TYPEARR, FREQARR, PROPARR, DEL03.1, VIOL, DSM2A, DSM5A,
            DSM14A:DSM17B, AGE, Q1.8, Q1.11, Q1.14, Q1.21, Q1.22, Q4.4, Q5.5.1:Q5.5.6,
            Q5.19A, Q5.19B, Q5.20A, Q5.20B, Q7.1, Q7.6:Q7.8, Q9.1:Q9.6, Q10.10N, Q10.11N,
            Q10.12N, Q10.13N, QREAS.02, QREAS.20, QREAS.21, K1.1, K1.3, K2.1, K2.3, K3.1,
            K3.3, K4.1, K4.3, K8.1, K8.3, K9.1, K9.3, K10.1, K10.3, K12.1, K12.3) %>%
     arrange(STUDYID)
COVR2 = COVR2 %>%
     mutate(STUDYID = studyid, F12VIOL = f12viol, PCLTOT = pcltot) %>%
     select(STUDYID, F12VIOL, PCLTOT) %>%
     arrange(STUDYID)
COVRdata = inner_join(COVR1, COVR2, by = 'STUDYID')
rm('COVR1', 'COVR2')

# remove data with missing outcome variable
COVRdata = filter(COVRdata, !is.na(F12VIOL))
```

Preprocess the data to create the variables used in the VRAS study.

```
ChildAbuseVars = COVRdata %>%
     select(Q5.5.1:Q5.5.6) %>%
     transmute(ChildAbuse = 5*(rowSums(cbind(Q5.5.4, Q5.5.5, Q5.5.6)) > 3) +
                    3*(rowSums(cbind(Q5.5.2, Q5.5.3)) > 2) + (as.numeric(Q5.5.1) > 1))
SubAbVars = COVRdata %>%
     select(DSM14A, DSM14B, DSM15A, DSM15B, DSM16A, DSM16B, DSM17A, DSM17B)
```

```
DrugAbVars = COVRdata %>%
     select(DSM16A, DSM16B, DSM17A, DSM17B)
HeadInjVars = COVRdata %>%
     select(NEU4B.1, NEU4B.2, NEU4B.3, NEU4B.4, NEU4B.5,
            NEU4B.6, NEU4B.7, NEU4B.8, NEU4B.9)
tcoPatient = select(COVRdata, K1.1, K2.1, K3.1, K4.1, K8.1, K9.1, K10.1, K12.1)
tcoClinical = select(COVRdata, K1.3, K2.3, K3.3, K4.3, K8.3, K9.3, K10.3, K12.3)

COVRdata = COVRdata %>%
     transmute(
          Violence = factor(ifelse(F12VIOL == 'Yes', 1, 0)),
          Age = as.numeric(AGE),
          BISnp = as.numeric(BISNPLN),
          BPRSa = as.numeric(OACTV),
          BPRSh = as.numeric(OHOST),
          BPRSt = as.numeric(OBPRS),
          ChildAbuse = ifelse(ChildAbuseVars$ChildAbuse >= 5, 3,
                              ifelse(ChildAbuseVars$ChildAbuse >= 3, 2,
                                     ifelse(ChildAbuseVars$ChildAbuse >= 1, 1, 0))),
          Consc = factor(ifelse(NEU2A == 'YES', 1, ifelse(NEU2A == 'NO', 0, NA))),

          DadDrug = factor(ifelse(((Q5.19A == 'DAILY' | Q5.19A == 'ONCE A WEEK' |
                                      Q5.19A == 'TWICE A WEEK') |
                                     (Q5.19B == 'DAILY' | Q5.19B == 'ONCE A WEEK' |
                                          Q5.19B == 'TWICE A WEEK')), 1,
                                ifelse(((is.na(Q5.19A) | Q5.19A == 'NA') &
                                           (Q5.19B == 'NA' | Q5.19B == 'DK')),
                                       NA, 0))),
          DadArr = factor(ifelse((Q5.20A == 'NEVER' &
                                      (Q5.20B == 'NA' | Q5.20B == 'DK') |
                                      (Q5.20B == 'NEVER' &
                                           (is.na(Q5.20A) | Q5.20A == 'NA')) |
                                      (Q5.20A == 'NEVER' & Q5.20B == 'NEVER')), 0,
                                 ifelse(((is.na(Q5.20A) | Q5.20A == 'NA') &
                                           (Q5.20B == 'NA' | Q5.20B == 'DK')),
                                        NA, 1))),
          DrugAbuse = factor(ifelse(rowSums(DrugAbVars == 'UNCERTAIN') == 4, NA,
                                    ifelse(rowSums(DrugAbVars == 'PRESENT') > 0, 1, 0))),
          Emp = factor(ifelse(Q4.4 %in% c('YES - FULL-TIME', 'YES - PART-TIME'), 1,
                              ifelse(Q4.4 == 'NO', 0, NA))),
          FantEsc = factor(ifelse((Q7.1 == 'YES' & Q7.7 == 'MORE SERIOUS'), 1, 0)),
          FantSing = factor(ifelse((Q7.1 == 'YES' & Q7.6 == 'SAME'), 1, 0)),
          FantTarg = factor(ifelse((Q7.1 == 'YES' & Q7.8 == 'YES'), 1, 0)),
          Function = ifelse(rowSums(is.na(cbind(Q9.1, Q9.2,
                                                Q9.3, Q9.4, Q9.5, Q9.6))) == 6, NA,
                            rowSums(cbind(Q9.1, Q9.2, Q9.3, Q9.4, Q9.5, Q9.6) - 1,
                                    na.rm = T)),
          GranDel = factor(ifelse(DEL03.1 == 'YES - CHECKED', 1, 0)),
          HeadInj = factor(ifelse(rowSums(HeadInjVars == 'YES, HEAD INJURY',
                                          na.rm = T) > 0, 1,
                                  ifelse(rowSums(is.na(HeadInjVars)) == 9, NA, 0))),
          LegalStatus = factor(ifelse(LEGALR == 'INVOLUNTARY', 1, 0)),
          NASb = as.numeric(NOVBEH),
```

```
          NegRel = rowSums(cbind(Q10.10N, Q10.11N, Q10.12N, Q10.13N)),
          PCL = factor(ifelse(PCLTOT > 12, 1, 0)),
          PCS = ifelse(rowSums(is.na(cbind(Q1.8, Q1.11, Q1.14, Q1.21, Q1.22))) == 5, NA,
                       rowSums(-1*(cbind(Q1.8, Q1.11, Q1.14, Q1.21, Q1.22)) + 2,
                               na.rm = T)),
          PriorArr = as.numeric(FREQARR) - 1,
          PropCrime = factor(ifelse(PROPARR == 'Yes', 1, 0)),
          RecViol2 = factor(ifelse(VIOL == 'Violence', 1, 0)),
          Schiz = factor(ifelse(((DSM2A == 'ABSENT' | DSM2A == 'UNCERTAIN') &
                                   (DSM5A == 'ABSENT'| DSM5A == 'UNCERTAIN')), 0, 1)),
          SNMHP = as.numeric(SNMHP),
          SubAbuse = factor(ifelse(rowSums(SubAbVars == 'UNCERTAIN') == 8, NA,
                                   ifelse(rowSums(SubAbVars == 'PRESENT') > 0, 1, 0))),
          Suicide = factor(ifelse(QREAS.02 == 'YES - CHECKED', 1, 0)),
          tco = factor(ifelse(rowSums((tcoPatient == 'YES') == (tcoClinical == 'YES'),
                                   na.rm = T) > 0, 1, 0)),
          Threats = factor(ifelse((QREAS.20 == 'YES - CHECKED' |
                                     QREAS.21 == 'YES - CHECKED'), 1,
                                  ifelse(rowSums(is.na(cbind(QREAS.20, QREAS.21))) == 2,
                                         NA, 0)))
          )
rm(list = ls()[ls() != 'COVRdata'])
```

Save the data table for construction of classification models.

```
save.image('COVRdata.rda')
```

Compute correlations between predictor variables and the response.

```
R = data.frame(sapply(COVRdata, as.numeric))
COVRr = apply(select(R, Violence), 2, cor, select(R, -Violence), 'pairwise.complete.obs')
rownames(COVRr) = colnames(select(R, -Violence))
round(COVRr, 2)
```

```
            Violence
Age            -0.07
BISnp           0.05
BPRSa          -0.08
BPRSh           0.08
BPRSt          -0.04
ChildAbuse      0.14
Consc           0.09
DadDrug         0.14
DadArr          0.15
DrugAbuse       0.16
Emp            -0.05
FantEsc         0.13
FantSing        0.10
FantTarg        0.12
Function       -0.01
GranDel        -0.01
HeadInj         0.03
```

LegalStatus	0.11
NASb	0.17
NegRel	0.05
PCL	0.26
PCS	0.03
PriorArr	0.24
PropCrime	0.11
RecViol2	0.14
Schiz	-0.12
SNMHP	-0.10
SubAbuse	0.18
Suicide	-0.01
tco	-0.09
Threats	0.06

Supplementary R Syntax — Classification Modeling

First load the necessary packages; this assumes the packages are installed. If not, use `install.packages()` (e.g., `install.packages("dplyr")` installs the `dplyr` package).

```
# for linear discriminant analysis
library(MASS)
# for imputation missing values
library(Hmisc)
# for cross-validation of models
library(boot)
# constructing ROC plots and computing AUC
library(ROCR)
# for data frame manipulation
library(dplyr)
# for plotting
library(ggplot2)
```

Next load preprocessed data.

```
load('COVRdata.rda')
```

Calculate the base rate of violence in the sample.

```
BR = mean(select(COVRdata, Violence) == 1)
```

```
Loading required namespace: data.table
```

Logisitic Regression Model

Monahan et al. (2001) constructed a main effects logistic regression (MELR) model to predict violence that was fit with forward-stepwise variable selection with a $p < .05$-threshold for retaining predictor variables. The present analysis constructs an MELR model but fitted with only the variables from the final model given by Monahan et al. The results are similar, but not exact (see Monahan et al., 2001, Table 5.1).

Before constructing the logistic regression model, we impute missing data by replacing all missing data with the mean of the non-missing data for continuous variables and the mode of the non-missing data for categorical variables, as was done by Monahan et al. (2001).

```
# function for computing the mode
varMode <- function(x) return(factor(names(table(x))[table(x) == max(table(x))]))

# function for missing value imputation; the function impute() is from Hsmic package
imputeNA <- function(x) {
     if (is.factor(x)) { #impute mode for factor variables

          return(factor(impute(x, varMode)))

     } else {
```

```
        return(impute(x, mean))

    }

}

# impute missing data
COVRdata = COVRdata %>%
    summarise_each(funs(imputeNA))
```

First select variables used in Monahan et al.'s (2001) logistic regression model.

```
logRegData = COVRdata %>%
    select(Violence, BISnp, BPRSa, BPRSh, BPRSt, ChildAbuse, Consc, DadDrug, DrugAbuse,
           Emp, FantEsc, FantSing, GranDel, LegalStatus, NASb, PCL, PriorArr, SNMHP, tco)
```

Next construct the logistic regression model.

```
logisticModel = glm(Violence ~ ., data = logRegData, family = binomial(logit))
summary(logisticModel)
```

```
Call:
glm(formula = Violence ~ ., family = binomial(logit), data = logRegData)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.7407  -0.6019  -0.4001  -0.2304   2.8571

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)  -2.89903    0.67142  -4.318 1.58e-05 ***
BISnp        -0.02796    0.01286  -2.174 0.029689 *
BPRSa        -0.15113    0.06327  -2.389 0.016914 *
BPRSh         0.11751    0.04022   2.922 0.003481 **
BPRSt        -0.03305    0.01605  -2.059 0.039469 *
ChildAbuse    0.37445    0.10422   3.593 0.000327 ***
Consc1        0.51808    0.26122   1.983 0.047330 *
DadDrug1      0.73625    0.27620   2.666 0.007685 **
DrugAbuse1    0.38073    0.22965   1.658 0.097344 .
Emp1         -0.47708    0.20007  -2.385 0.017098 *
FantEsc1      0.67541    0.32616   2.071 0.038375 *
FantSing1     0.56422    0.25876   2.181 0.029219 *
GranDel1      0.71089    0.34357   2.069 0.038533 *
LegalStatus1  0.51104    0.19855   2.574 0.010059 *
NASb          0.03796    0.01508   2.518 0.011808 *
PCL1          0.89817    0.21125   4.252 2.12e-05 ***
PriorArr      0.29782    0.08199   3.632 0.000281 ***
SNMHP        -1.85496    0.74709  -2.483 0.013032 *
tco1         -0.90025    0.34200  -2.632 0.008481 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 906.10  on 938  degrees of freedom
Residual deviance: 731.06  on 920  degrees of freedom
AIC: 769.06

Number of Fisher Scoring iterations: 5
```

We now compute the cross-validation error of logistic regression model, using leave-one-out cross-validation.

```
# function for calculating error when using classification cutscore of .5
LRcost50 = function(x, p = 0) mean(abs(x - p) > .5)
# estimated cross-validated error
cv.glm(logRegData, logisticModel, LRcost50)$delta[1]
```

```
[1] 0.1821086
```

```
# resubstitution error
LRpreds = predict(logisticModel, type = 'resp')
mean((LRpreds > .5) != (select(COVRdata, Violence) == 1))
```

```
[1] 0.1693291
```

```
# function for calculating error when using classification cutscore of .37
LRcost37 = function(x, p = 0) mean(abs(x - p) > .37)
# estimated cross-validated error
cv.glm(logRegData, logisticModel, LRcost37)$delta[1]
```

```
[1] 0.2406816
```

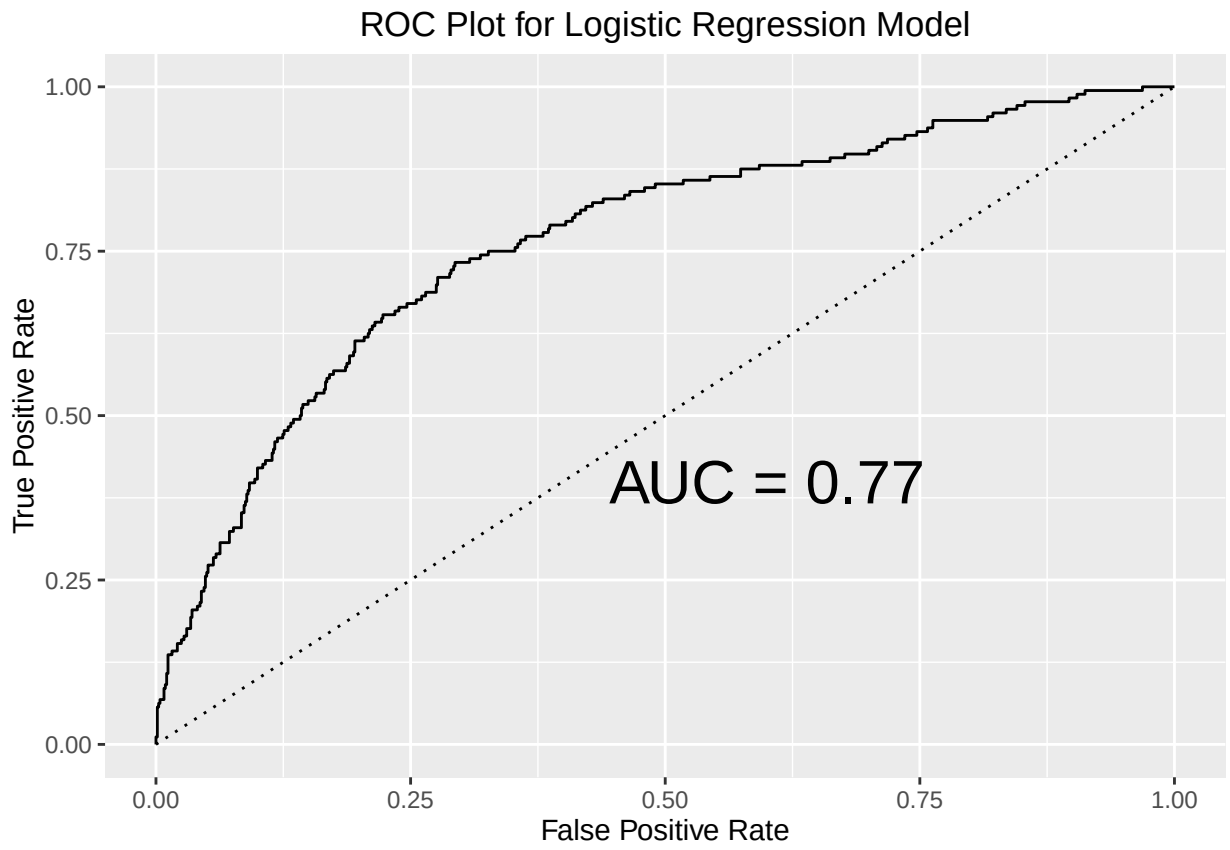
```
# resubstitution error
mean((LRpreds > 2*BR) != (select(COVRdata, Violence) == 1))
```

```
[1] 0.1789137
```

Next we plot an ROC curve and calculate the AUC. We want to use cross-validated results, not the results from the model. To do so, we need to manually compute the cross-validated estimates because `cv.glm()` unfortunately does not provide this.

```
err = double()
for (k in 1:939) {
     t = glm(Violence ~ ., data = logRegData, family = binomial(logit), subset = -k)
     # predicted probabilty of being violent
     err[k] = predict(t, COVRdata[k,], type = 'resp')
}


lrPreds = prediction(err, select(COVRdata, Violence))
lrPerf = performance(lrPreds, 'tpr', 'fpr')
```

```
AUC = round(performance(lrPreds, 'auc')@y.values[[1]], 2)
ggplot(data = NULL) +
     geom_line(aes(x = lrPerf@x.values[[1]],
                                y = lrPerf@y.values[[1]])) +
     ggtitle('ROC Plot for Logistic Regression Model') +
     xlab('False Positive Rate') +
     ylab('True Positive Rate') +
     geom_segment(aes(x = 0, y = 0, xend = 1, yend = 1),
                  linetype = 'dotted') +
     geom_text(aes(x = .6, y = .4, label = paste0('AUC = ', AUC), parse = T),
               size = 8)
```



Discriminant Function Analysis

This section applies discriminant function analysis (DFA) to classify individuals as violent, for both a linear fit (i.e., assuming the covariances among the two populations—nonviolent and violent—are equal) and a quadratic fit (i.e., the covariances are allowed to be unequal). This was not done by Monahan et al. (2001) but allows comparison with the other methods used (logistic regression and classification trees).

The same data with missing value imputation that was used in the logistic regression model is used for the discriminant analyses. All the data are used.

First, a linear discriminant function is constructed, with equal costs and unequal costs. Before doing so, the workspace is cleared, the base rate for violence is calculated, and the prior probabilities are established such that unequal costs are applied.

```r
# Clear workspace
rm(list = ls()[!(ls() %in% c('BR', 'COVRdata'))])
# cost of FN to FP
fnCost = (1-2*BR)/(2*BR)
# priors
priors = c((1 - BR)/(BR*fnCost + (1-BR)), BR*fnCost/(BR*fnCost + (1-BR)))
rm(fnCost)
```

We then compute resubstitution error and cross-validation error of the linear discriminant model, using leave-one-out cross-validation. This is found for both the model with equal costs and the one with unequal costs.

```r
# Equal costs
# resubstitution error
ldaModel_equal = lda(Violence ~ ., data = COVRdata)
mean(predict(ldaModel_equal)$class != COVRdata$Violence)
```

```
[1] 0.1735889
```

```r
# estimated cross-validated error
ldaModel_equal = lda(Violence ~ ., data = COVRdata, CV = T)
mean(ldaModel_equal$class != COVRdata$Violence)
```

```
[1] 0.1884984
```

```r
# Unequal costs
# resubstitution error
ldaModel_unequal = lda(Violence ~ ., data = COVRdata, prior = priors)
mean(predict(ldaModel_unequal)$class != COVRdata$Violence)
```

```
[1] 0.1842386
```

```r
# estimated cross-validated error
ldaModel_unequal = lda(Violence ~ ., data = COVRdata, CV = T, prior = priors)
mean(ldaModel_unequal$class != COVRdata$Violence)
```

```
[1] 0.2044728
```

Next a quadratic discriminant function is constructed, again with equal costs and unequal costs.

```r
# Equal costs
# resubstitution error
qdaModel_equal = qda(Violence ~ ., data = COVRdata)
mean(predict(qdaModel_equal)$class != COVRdata$Violence)
```

```
[1] 0.14377
```

```r
# estimated cross-validated error
qdaModel_equal = qda(Violence ~ ., data = COVRdata, CV = T)
mean(qdaModel_equal$class != COVRdata$Violence)
```

```
[1] 0.2300319
```

```
# Unequal costs
# resubstitution error
qdaModel_unequal = qda(Violence ~ ., data = COVRdata, prior = priors)
mean(predict(qdaModel_unequal)$class != COVRdata$Violence)
```

```
[1] 0.1522897
```

```
# estimated cross-validated error
qdaModel_unequal = qda(Violence ~ ., data = COVRdata, CV = T, prior = priors)
mean(qdaModel_unequal$class != COVRdata$Violence)
```

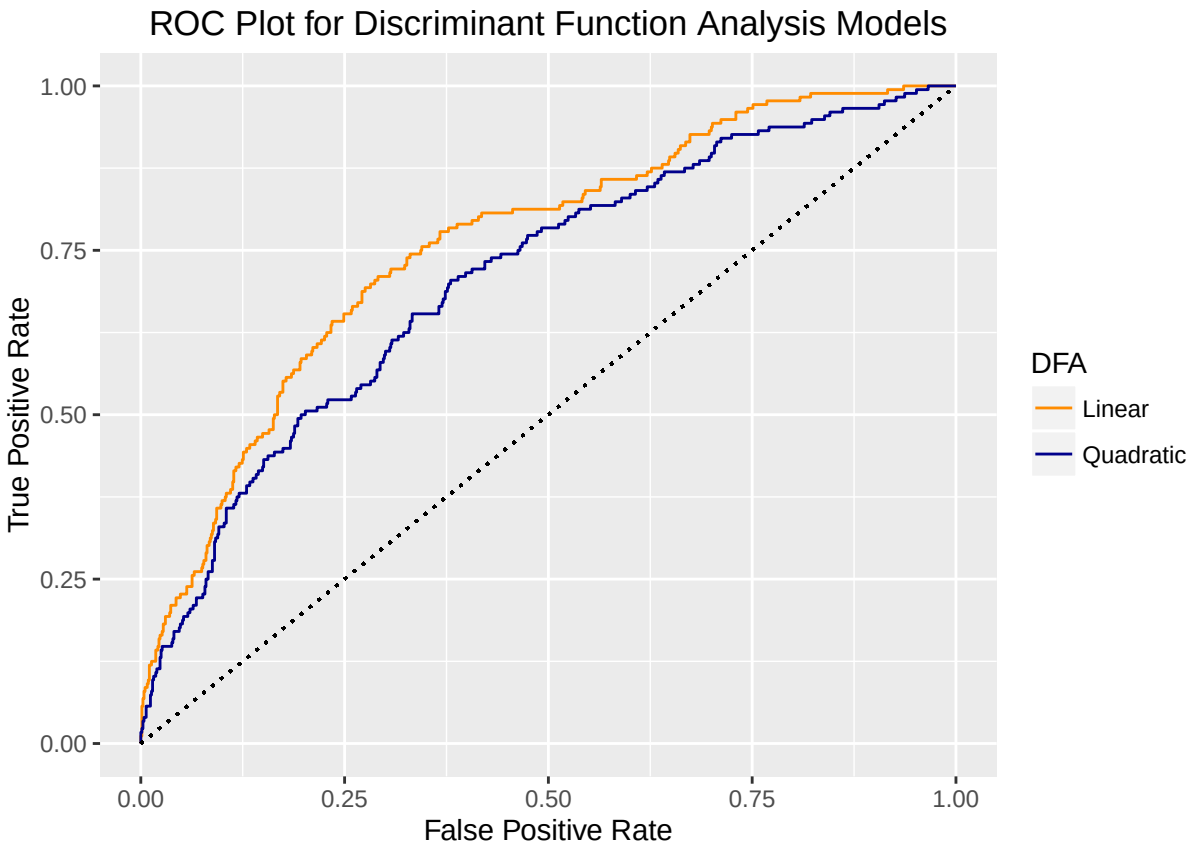
```
[1] 0.2513312
```

Plotting ROC curve.

```
ldaPreds = prediction(ldaModel_equal$post[,2], select(COVRdata, Violence))
ldaPerf = performance(ldaPreds, 'tpr', 'fpr')
qdaPreds = prediction(qdaModel_equal$post[,2], select(COVRdata, Violence))
qdaPerf = performance(qdaPreds, 'tpr', 'fpr')
AUC = data_frame(lda = round(performance(ldaPreds, 'auc')@y.values[[1]], 2),
                 qda = round(performance(qdaPreds, 'auc')@y.values[[1]], 2))

# AUC for models
AUC
```

```
Source: local data frame [1 x 2]

    lda   qda
  (dbl) (dbl)
1  0.76  0.71
```

```
# ROC plot
dfaStats = data_frame(x = c(ldaPerf@x.values[[1]], qdaPerf@x.values[[1]]),
                      y = c(ldaPerf@y.values[[1]], qdaPerf@y.values[[1]]),
                      DFA = rep(c('Linear', 'Quadratic'),
                                times = c(length(ldaPerf@x.values[[1]]),
                                          length(qdaPerf@x.values[[1]]))))
ggplot(data = dfaStats) +
     geom_line(aes(x = x, y = y, color = DFA)) +
     ggtitle('ROC Plot for Discriminant Function Analysis Models') +
     xlab('False Positive Rate') +
     ylab('True Positive Rate') +
     geom_segment(aes(x = 0, y = 0, xend = 1, yend = 1),
                  linetype = 'dotted') +
     scale_color_manual(values = c('darkorange', 'darkblue'))
```


ROC Plot for Discriminant Function Analysis Models
True Positive Rate
False Positive Rate
0.00
0.25
0.50
0.75
1.00
DFA
Linear
Quadratic

Supplementary R Syntax – Classification Trees

First load the necessary packages; this assumes the packages are installed. If not, use `install.packages()` (e.g., `install.packages("rpart")` installs the `rpart` package).

```
# for recursive partitioning trees
library(rpart)
# for plotting rpart trees
library(rpart.plot)
# for random forest model
library(randomForest)
# for plotting
library(ggplot2)
# for data frame manipulation
library(dplyr)
# for cross-validation and other machine learning tools
library(caret)
# for constructing ROC plots and computing AUC
library(ROCR)
```

Next load preprocessed data.

```
load('COVRdata.rda')
```

Calculate the base rate of violence in the sample.

```
BR = mean(select(COVRdata, Violence) == 1)
```

Classification Trees

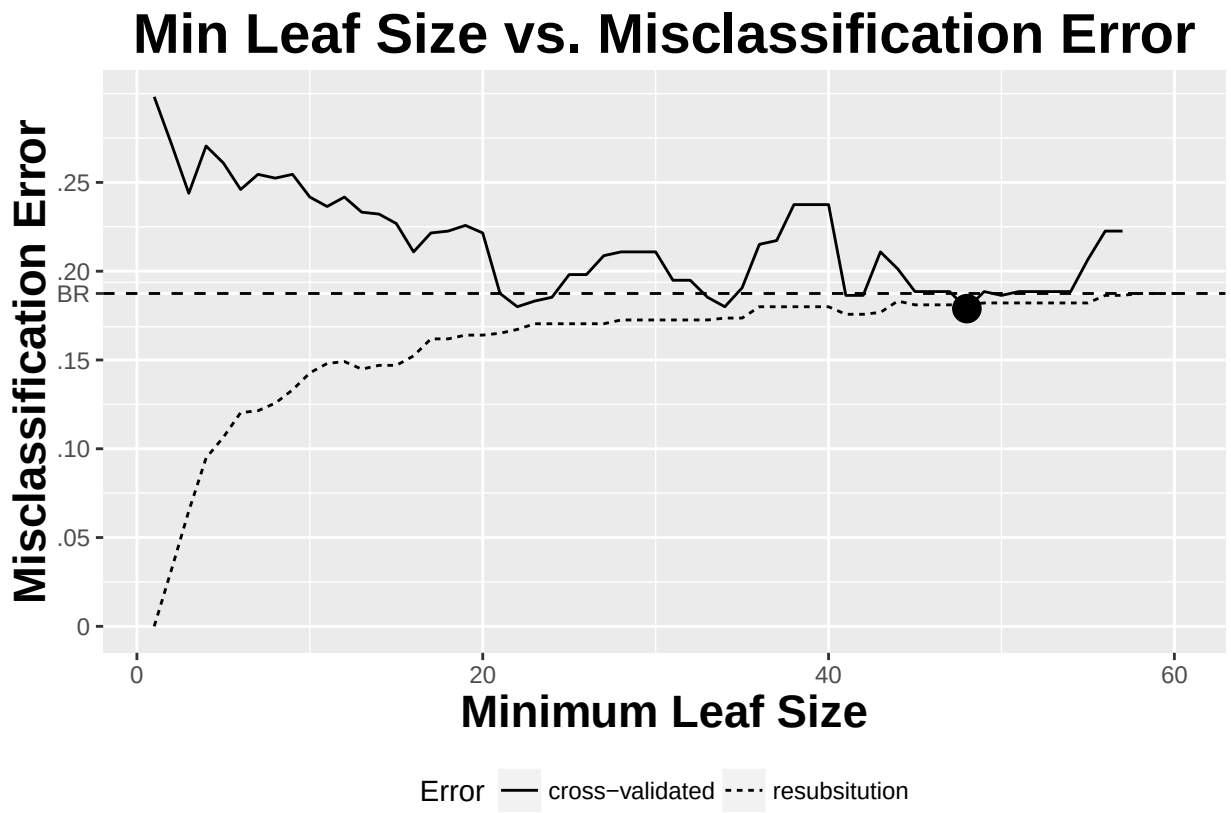
Using leave-one-out cross-validated error, we determine the minimum leaf size; the range of the minimum leaf size is $[1, 60]$. The minimum leaf size is specified with the `minbucket` statement and the `cp = 0` requests that the tree is not pruned. In addition we set the minimum split to be twice the minimum leaf (the minimum split is the minimum number needed to make a split; the default is three times the minimum leaf). Note that the relative error (`rel error`) and the expected error (`xerror`) are multiplied by the base rate giving, respectively, the estimated resubstitution and cross-validated error. The base rate is the root node error and the relative and expected errors are with respect to the root node error. Note that the cross-validated and resubstitution errors are from the *unpruned* trees. Missing values are treated as suggested by Breimen et al. (1984) and as was done by Monahan et al. (2001); this is automatically implemented in `rpart` with the `rpart.control` option `usesurrogate` set equal to 2 (default). The `xval` option is set equal to the sample size (this gives the estimate for leave-one-out cross-validation; the default is ten-fold).

```
CVerror = double() #cross-validated error
RSerror = double() #resubstitution error

for (j in 1:60) {

     t = rpart(Violence ~ ., COVRdata, control = rpart.control(minbucket = j,
                                                               minsplit = 2*j, cp = 0,
                                                               xval = nrow(COVRdata)))
```

```
    # error estimates
    CVerror[j] = t$cp[nrow(t$cp),4]*BR
    RSerror[j] = t$cp[nrow(t$cp),3]*BR

}

# create data frame with errors and leaf sizes
misclassError = data_frame('minleaf' = rep(1:60, 2), mcerror = c(RSerror, CVerror),
                           Error = rep(c('resubsitution', 'cross-validated'), each = 60))

# remove CVerror = 0
misclassError = misclassError %>%
    filter((mcerror != 0 & Error == 'cross-validated') | Error == 'resubsitution')

# optimal minimum leaf size
minLeaf = misclassError %>%
    filter(Error == 'cross-validated') %>%
    filter(rank(mcerror, ties.method = 'first') == 1)
minLeaf
```

```
Source: local data frame [1 x 3]

  minleaf   mcerror           Error
    (int)     (dbl)           (chr)
1      48 0.1789137 cross-validated
```

```
# plot error across minimum leaf size
ggplot(data = misclassError, aes(minleaf, mcerror, linetype = Error)) +
    geom_line() +
    geom_hline(yintercept = BR, lty = 2) +
    scale_y_continuous(breaks = c(0, .05, .1, .15, BR, .20, .25),
                       labels = c('0', '.05', '.10', '.15', 'BR', '.20','.25')) +
    theme(plot.title = element_text(size = 21, face = 'bold'),
          axis.title = element_text(size = 17, face = "bold")) +
    xlab('Minimum Leaf Size') + ylab('Misclassification Error') +
    labs(title = 'Min Leaf Size vs. Misclassification Error') +
    theme(legend.position = 'bottom') +
    geom_point(data = minLeaf, aes(y = mcerror, x = minleaf, size = 3), show_guide = F)
```



Now we construct a tree using the minimum leaf size that minimized the cross-validated error (this was found to be 48). A confusion matrix is given (the function `confsuionMatrix()` is available through the `caret` package). The tree is then plotted using the `prp()` function that is available through the `rpart.plot` package.

```
ctree = rpart(Violence ~ ., COVRdata,
                  control = rpart.control(minbucket = select(minLeaf, minleaf),
                                          minsplit = 2*select(minLeaf, minleaf), cp = 0))
# resubstitution error
mean(predict(ctree, COVRdata, type = 'class') != COVRdata$Violence)
```

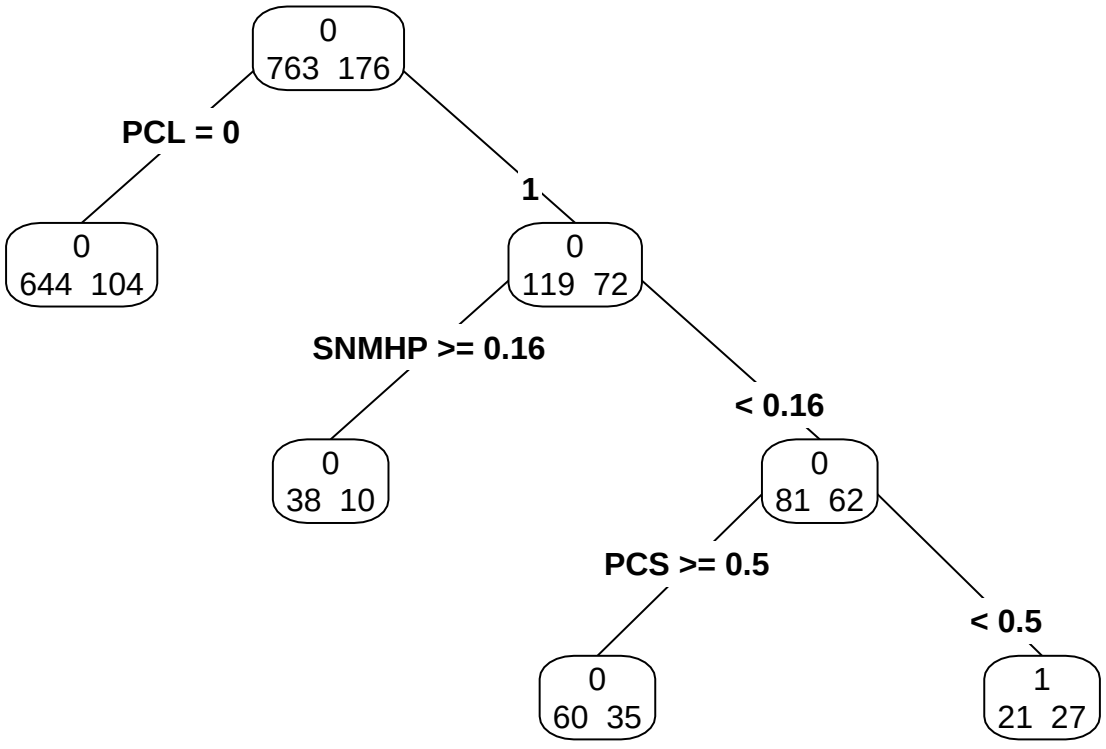
```
[1] 0.1810437
```

```
# confusion Matrix
confusionMatrix(predict(ctree, type = 'class'), COVRdata$Violence, positive = '1')
```

```
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 742 149
         1  21  27

               Accuracy : 0.819
                 95% CI : (0.7928, 0.8431)
```

```
    No Information Rate : 0.8126
    P-Value [Acc > NIR] : 0.3253

                  Kappa : 0.1748
 Mcnemar's Test P-Value : <2e-16

            Sensitivity : 0.15341
            Specificity : 0.97248
         Pos Pred Value : 0.56250
         Neg Pred Value : 0.83277
             Prevalence : 0.18743
         Detection Rate : 0.02875
   Detection Prevalence : 0.05112
      Balanced Accuracy : 0.56294

       'Positive' Class : 1
```

```r
# plot tree
prp(ctree, type = 4, extra = 1)
```

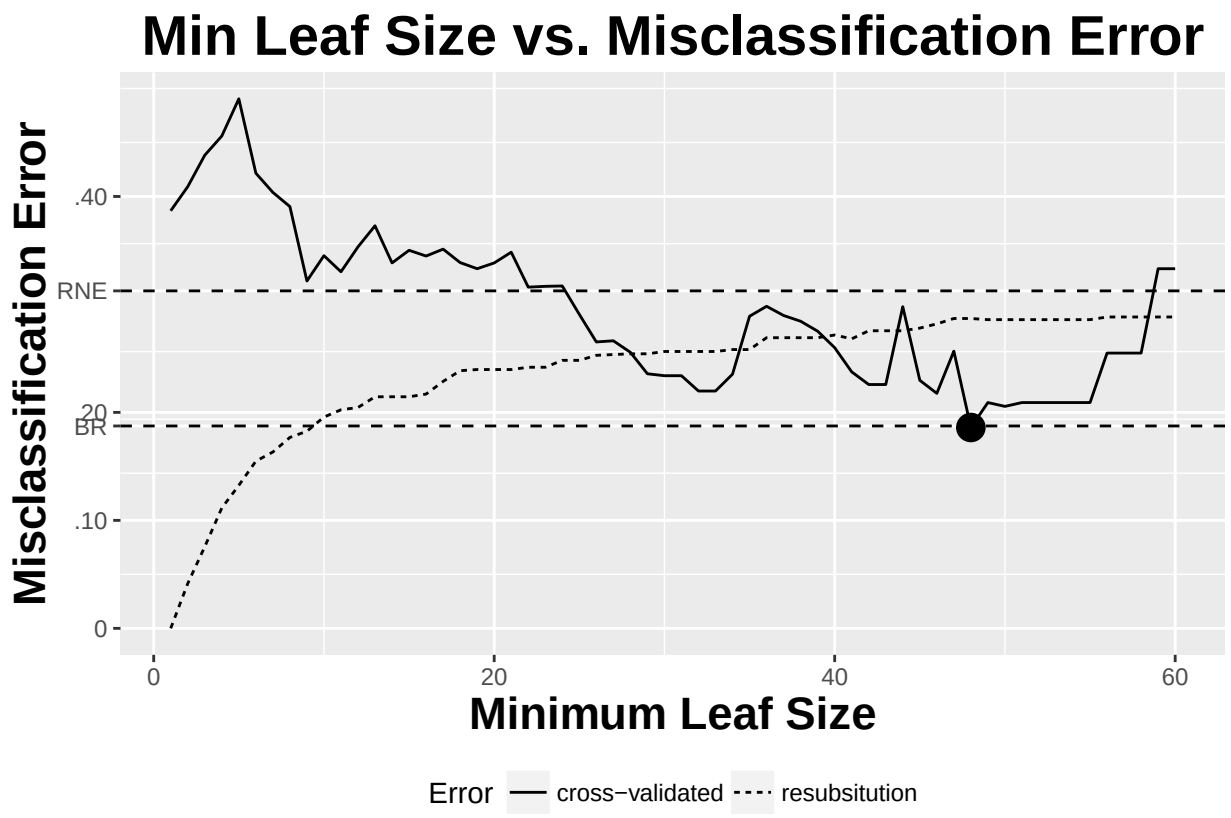


Next we implement unequal costs as implied by Monahan et al. (2001). We begin by creating a cost matrix; this is based on Monahan et al.'s (2001) choice of cutscore equal to 0.37.

```
# Clear the workspace.
rm(list = ls()[!(ls() %in% c('BR', 'COVRdata'))])
# cost matrix
twoBR = (1-2*BR)/(2*BR)
costMatrix = matrix(c(0, twoBR, 1, 0), 2)
```

We again determine the minimum leaf size, as was done with equal costs. Note that the root node error is 0.31 (= BR*twoBR; that is, the number of false negatives at the root note weighted by the cost of false negatives to false positives, 1.67).

```
CVerror = double() #cross-validated error
RSerror = double() #resubstitution error

for (j in 1:60) {

      t = rpart(Violence ~ ., COVRdata,  parms = list(loss = costMatrix),
                control = rpart.control(minbucket = j, minsplit = 2*j, cp = 0,
                                        xval = nrow(COVRdata)))

     # error estimates
     CVerror[j] = t$cp[nrow(t$cp),4]*BR*twoBR
     RSerror[j] = t$cp[nrow(t$cp),3]*BR*twoBR

}

# create data frame with error and leaf size
misclassError = data_frame('minleaf' = rep(1:60, 2), mcerror = c(RSerror, CVerror),
                           Error = rep(c('resubsitution', 'cross-validated'), each = 60))

# optimal minimum leaf size
minLeaf = misclassError %>%
     filter(Error == 'cross-validated') %>%
     filter(rank(mcerror, ties.method = 'first') == 1)

# plot error across minimum leaf size
ggplot(data = misclassError, aes(minleaf, mcerror, linetype = Error)) +
     geom_line() +
     geom_hline(yintercept = BR, linetype = 2) +
     geom_hline(yintercept = BR*twoBR, lty = 2) +
     scale_y_continuous(breaks = c(0, .1, BR, .2, BR*twoBR, .4),
                        labels = c('0', '.10', 'BR', '.20','RNE', '.40')) +
     theme(plot.title = element_text(size = 21, face = 'bold'),
           axis.title = element_text(size = 17, face = "bold")) +
     xlab('Minimum Leaf Size') + ylab('Misclassification Error') +
     labs(title = 'Min Leaf Size vs. Misclassification Error') +
     theme(legend.position = 'bottom') +
     geom_point(data = minLeaf, aes(y = mcerror, x = minleaf, size = 3), show_guide = F)
```



Next we construct and plot the tree using minimum leaf size, which was found to be 48. *This is exactly the same as the tree with equal costs* aside from one slight difference which does not affect the classification results: the non-terminal node at `SNMHP < .16` is classified as violent as opposed to nonviolent (as is the case when the costs were equal). A this node the proportion violent is 0.43 which is less than .5 but more than 0.37.

```
ctree = rpart(Violence ~ ., COVRdata, parms = list(loss = costMatrix),
                  control = rpart.control(minbucket = select(minLeaf, minleaf),
                                          minsplit = 2*select(minLeaf, minleaf), cp = 0,
                                          xval = nrow(COVRdata)))
# resubstitution error
mean(predict(ctree, COVRdata, type = 'class') != COVRdata$Violence)
```

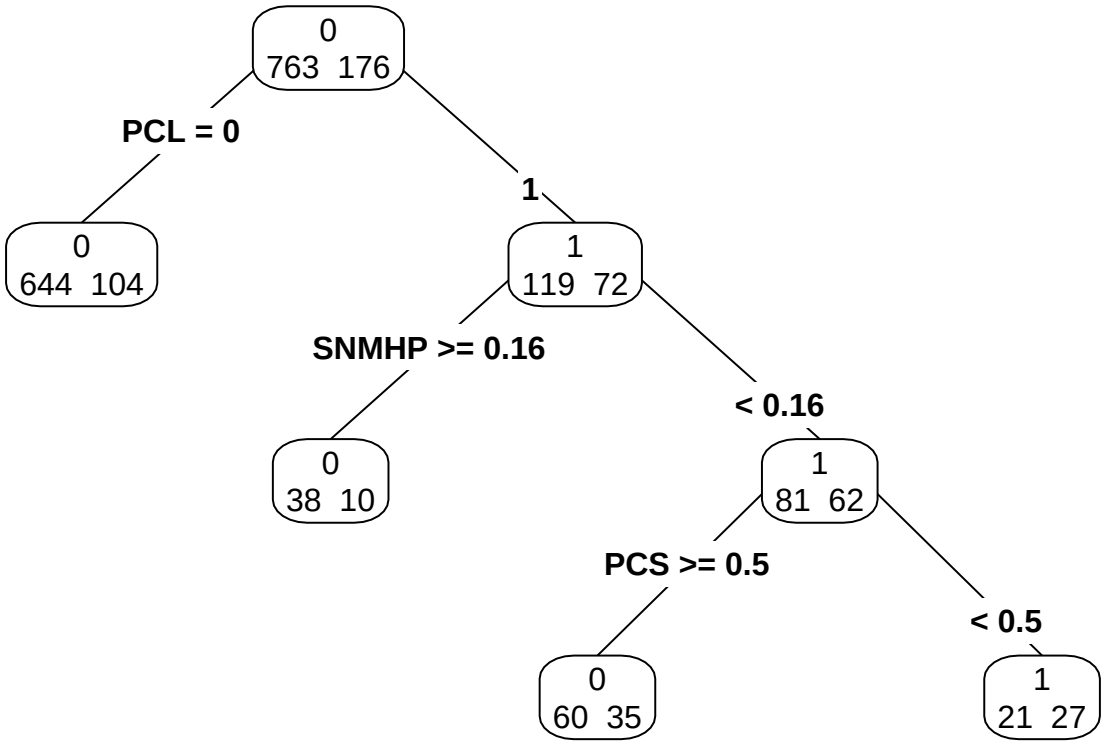
```
[1] 0.1810437
```

```
# confusion Matrix
confusionMatrix(predict(ctree, type = 'class'), COVRdata$Violence, positive = '1')
```

```
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 742 149
         1  21  27
```

```
               Accuracy : 0.819
                 95% CI : (0.7928, 0.8431)
    No Information Rate : 0.8126
    P-Value [Acc > NIR] : 0.3253

                  Kappa : 0.1748
 Mcnemar's Test P-Value : <2e-16

            Sensitivity : 0.15341
            Specificity : 0.97248
         Pos Pred Value : 0.56250
         Neg Pred Value : 0.83277
             Prevalence : 0.18743
         Detection Rate : 0.02875
   Detection Prevalence : 0.05112
      Balanced Accuracy : 0.56294

       'Positive' Class : 1
```

```r
# plot tree
prp(ctree, type = 4, extra = 1)
```



Now we construct a tree using unequal costs as suggested in Berk (2012). Here, the minimum leaf size is chosen (arbitrarily) to be 30, and pruned. The relative error (`rel error`) and expected error (`xerror`) are not appropriate to use for calculating resubstitution error and cross-validated error here because they are

in relation to the root node error, which is itself in relation to the specified priors and costs. Instead, we implement the `train()` function, an extremely versatile cross-validation tool from the `caret()` package.

```
# clear workspace
rm(list = ls()[!(ls() %in% c('BR', 'COVRdata', 'costMatrix'))])
# construct tree
ctree = train(select(COVRdata, -Violence), COVRdata$Violence,
              method = 'rpart', parms = list(loss = matrix(c(0, 20, 1, 0), 2)),
              control = rpart.control(minbucket = 30, minsplit = 60),
              trControl = trainControl('cv', 'LOOCV'), tuneLength = 20)

# leave-one-out cross-validated error
1 - ctree$results$Accuracy[which.max(ctree$results$Accuracy)]
```

```
[1] 0.5857295
```

```
# resubstitution error
mean(predict(ctree) != COVRdata$Violence)
```

```
[1] 0.5697551
```

Finally, to demonstrate overfitting with classification trees, we fit a tree with a minimum leaf of one and without pruning.

```
rm(list = ls()[!(ls() %in% c('BR', 'COVRdata', 'costMatrix'))])
# equal costs
ctree = rpart(Violence ~ ., COVRdata, control = rpart.control(minbucket = 1, cp = 0,
                                                              xval = nrow(COVRdata)))

# leave-one-out cross-validated error
ctree$cp[nrow(ctree$cp),4]*BR
```

```
[1] 0.2960596
```

```
# resubstitution error
ctree$cp[nrow(ctree$cp),3]*BR
```

```
[1] 0.008519702
```

```
# Confusion Matrix
ctreePredClass = predict(ctree, COVRdata, type = 'class')
confusionMatrix(ctreePredClass, COVRdata$Violence, positive = '1')
```

```
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 761   6
         1   2 170
```

```
               Accuracy : 0.9915
                 95% CI : (0.9833, 0.9963)
    No Information Rate : 0.8126
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.9718
 Mcnemar's Test P-Value : 0.2888

            Sensitivity : 0.9659
            Specificity : 0.9974
         Pos Pred Value : 0.9884
         Neg Pred Value : 0.9922
             Prevalence : 0.1874
         Detection Rate : 0.1810
   Detection Prevalence : 0.1832
      Balanced Accuracy : 0.9816

       'Positive' Class : 1
```

```r
# unequal costs
ctree2 = train(select(COVRdata, -Violence), COVRdata$Violence,
               method = 'rpart', parms = list(loss = costMatrix),
               control = rpart.control(minbucket = 1),
               trControl = trainControl('cv', 'LOOCV'), tuneGrid = expand.grid(.cp = 0))

# leave-one-out cross-validated error
1 - ctree2$results$Accuracy
```

```
[1] 0.2790202
```

```r
# resubstitution error
mean(predict(ctree2) != COVRdata$Violence)
```

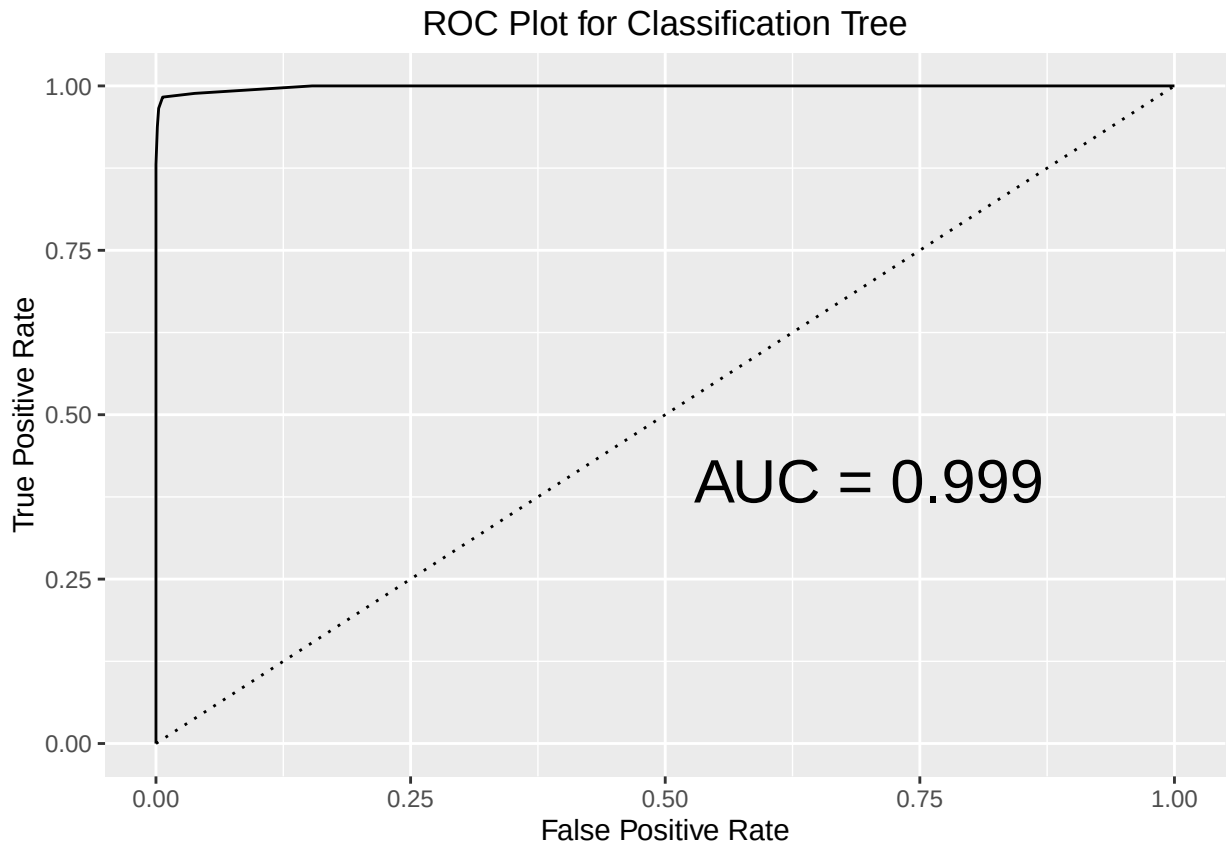
```
[1] 0.01277955
```

```r
# Confusion Matrix
confusionMatrix(predict(ctree2), COVRdata$Violence, positive = '1')
```

```
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 751   0
         1  12 176

               Accuracy : 0.9872
                 95% CI : (0.9778, 0.9934)
    No Information Rate : 0.8126
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.9591
```

```
 Mcnemar's Test P-Value : 0.001496

            Sensitivity : 1.0000
            Specificity : 0.9843
         Pos Pred Value : 0.9362
         Neg Pred Value : 1.0000
             Prevalence : 0.1874
         Detection Rate : 0.1874
   Detection Prevalence : 0.2002
      Balanced Accuracy : 0.9921

       'Positive' Class : 1
```

```
# ROC plot
ctreePredProbs = predict(ctree, COVRdata)[,2]
ctreePreds = prediction(ctreePredProbs, select(COVRdata, Violence))
ctreePerf = performance(ctreePreds, 'tpr', 'fpr')
AUC = round(performance(ctreePreds, 'auc')@y.values[[1]], 3)
ggplot(data = NULL) +
     geom_line(aes(x = ctreePerf@x.values[[1]],
                                 y = ctreePerf@y.values[[1]])) +
     ggtitle('ROC Plot for Classification Tree') +
     xlab('False Positive Rate') +
     ylab('True Positive Rate') +
     geom_segment(aes(x = 0, y = 0, xend = 1, yend = 1),
                  linetype = 'dotted') +
     geom_text(aes(x = .7, y = .4, label = paste0('AUC = ', AUC), parse = T),
               size = 8)
```



Random Forests

The next step is to construct the random forest models. Begin by clearing the workspace. Random forest models can be generated using the `randomForest()` function from the `randomForest` package.

```
rm(list = ls()[!(ls() %in% c('BR', 'COVRdata'))])
```

First we impute missing data. The function `rfImpute()` first imputes missing data by using the mean (for continuous data) or the mode (for categorical data). Next, a random forest is fit to the new dataset that no longer contains missing data. A proximity matrix is calculated such that the (i, j)th entry contains the proportion of times that the ith and jth observation fall into the same terminal node. For continuous data, the missing values are imputed using the weighted (by proximity) average across the variable of the observations; for categorical data, the missing values are equal to the value with the largest proximity, averaged across all the observations for the variable in question. Random seeds are set for replication of results.

```
set.seed(917)
COVRdataRF = rfImpute(Violence ~ ., COVRdata)
```

```
ntree      OOB      1      2
  300:  18.32%  0.92% 93.75%
ntree      OOB      1      2
  300:  18.32%  1.18% 92.61%
ntree      OOB      1      2
  300:  18.74%  1.83% 92.05%
```

```
ntree      OOB      1      2
  300:  19.06%  2.10% 92.61%
ntree      OOB      1      2
  300:  18.85%  1.57% 93.75%
```

The data are split into a training set and testing set. The testing set contains 282 (30%) observations; the training set contains the remaining 657 (70%) observations.

```
set.seed(1983)
COVRtrain = sample_frac(COVRdataRF, .7)
COVRtest = COVRdataRF %>%
     filter(!(row_number() %in% rownames(COVRtrain)))
```

Here, the random forest model is constructed using equal costs.

```
set.seed(91783)
covrRF = randomForest(Violence ~ ., data = COVRtrain, ntree = 1000)
```

A confusion matrix for the training data is created; the misclassification error (i.e., the proportion along the off-diagonal) is the resubstitution error.

```
confusionMatrix(predict(covrRF, COVRtrain), COVRtrain$Violence, positive = '1')
```

```
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 538   0
         1   0 119

               Accuracy : 1
                 95% CI : (0.9944, 1)
    No Information Rate : 0.8189
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 1
 Mcnemar's Test P-Value : NA

            Sensitivity : 1.0000
            Specificity : 1.0000
         Pos Pred Value : 1.0000
         Neg Pred Value : 1.0000
             Prevalence : 0.1811
         Detection Rate : 0.1811
   Detection Prevalence : 0.1811
      Balanced Accuracy : 1.0000

       'Positive' Class : 1
```

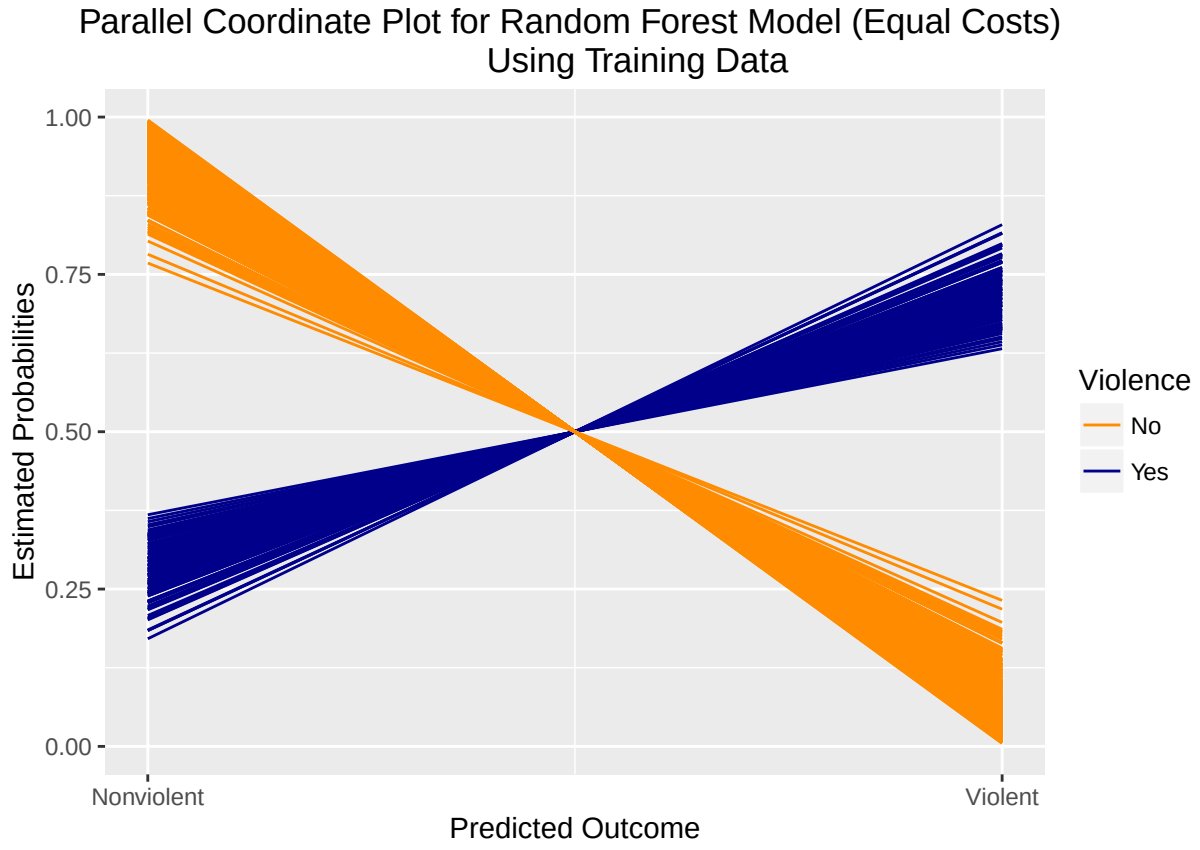
The cross-validated error can be estimated using the testing dataset.

```
confusionMatrix(predict(covrRF, COVRtest), COVRtest$Violence, positive = '1')
```

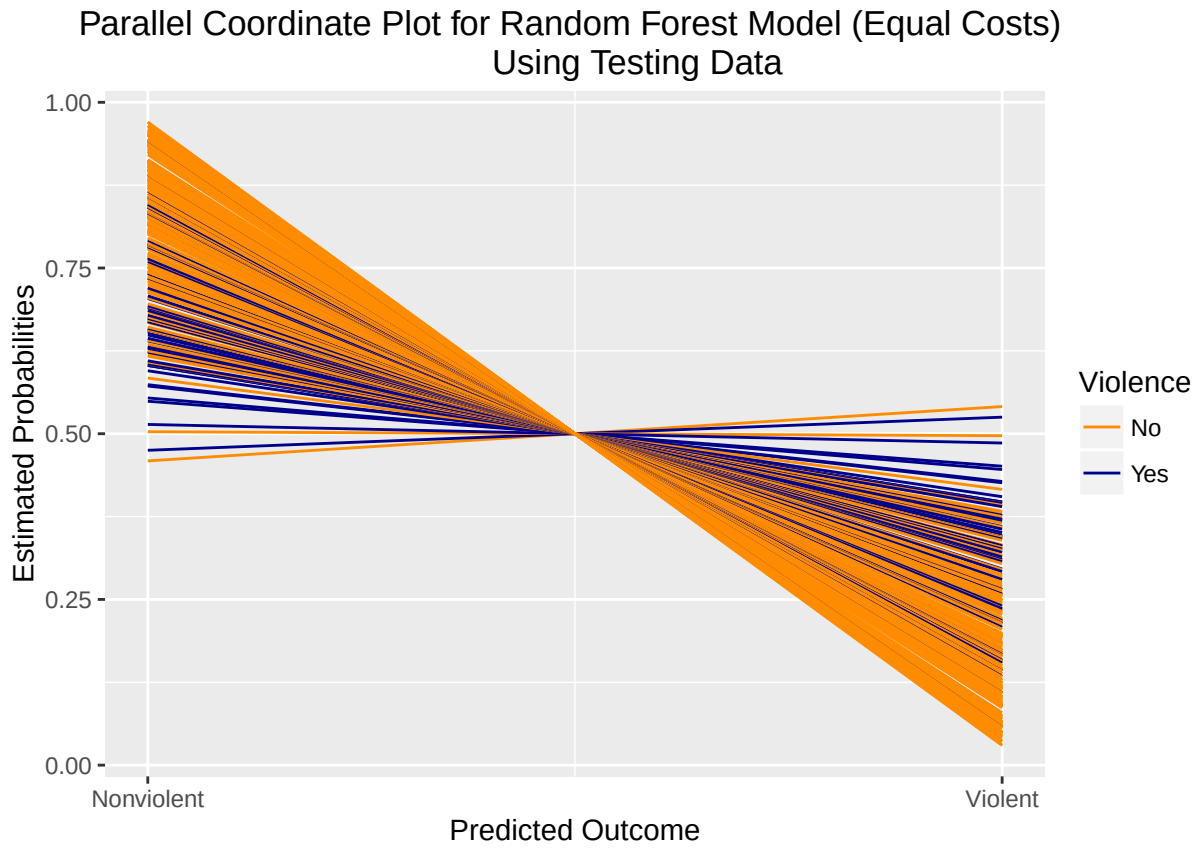
```
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 224  56
         1   1   1

               Accuracy : 0.7979
                 95% CI : (0.7462, 0.8432)
    No Information Rate : 0.7979
    P-Value [Acc > NIR] : 0.5354

                  Kappa : 0.0205
 Mcnemar's Test P-Value : 8.523e-13

            Sensitivity : 0.017544
            Specificity : 0.995556
         Pos Pred Value : 0.500000
         Neg Pred Value : 0.800000
             Prevalence : 0.202128
         Detection Rate : 0.003546
   Detection Prevalence : 0.007092
      Balanced Accuracy : 0.506550

       'Positive' Class : 1
```

To visualize the separation between the violent and non-violent individuals, we construct parallel coordinate plots for the training data and the testing data.

```
# data frame of estimated probabilities of violence and actual classification
rfPreds = data.frame(Preds = predict(covrRF, COVRtrain, type = 'prob'),
                     select(COVRtrain, Violence))
rfPreds = arrange(rfPreds, Preds.0)
ggplot(data = rfPreds) +
     geom_segment(aes(x = 0, xend = 1, y = Preds.0, yend = Preds.1, color = Violence)) +
     scale_x_continuous(breaks = c(0, 1),
                        labels = c('Nonviolent', 'Violent')) +
     xlab('Predicted Outcome') +
     ylab('Estimated Probabilities') +
     scale_color_manual(values = c('darkorange', 'darkblue'),
                        labels = c('No', 'Yes')) +
     ggtitle('Parallel Coordinate Plot for Random Forest Model (Equal Costs)
             Using Training Data')
```




```
# remove rfPreds
rm(rfPreds)

# data frame of estimated probabilities of violence and actual classification
rfPreds = data.frame(Preds = predict(covrRF, COVRtest, type = 'prob'),
                     select(COVRtest, Violence))
rfPreds = arrange(rfPreds, Preds.0)
ggplot(data = rfPreds) +
     geom_segment(aes(x = 0, xend = 1, y = Preds.0, yend = Preds.1, color = Violence)) +
     scale_x_continuous(breaks = c(0, 1),
                        labels = c('Nonviolent', 'Violent')) +
     xlab('Predicted Outcome') +
     ylab('Estimated Probabilities') +
     scale_color_manual(values = c('darkorange', 'darkblue'),
                        labels = c('No', 'Yes')) +
     ggtitle('Parallel Coordinate Plot for Random Forest Model (Equal Costs)
             Using Testing Data')
```



Now the random forest model is constructed using unequal costs.

```
rm(covrRF, rfPreds)
set.seed(917)
covrRF = randomForest(Violence ~ ., data = COVRtrain, ntree = 1000,
                      cutoff = c(1-2*BR, 2*BR))
```

The misclassification rates:

```
# resubstitution error
confusionMatrix(predict(covrRF, COVRtrain), COVRtrain$Violence, positive = '1')
```

```
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 538   0
         1   0 119

               Accuracy : 1
                 95% CI : (0.9944, 1)
    No Information Rate : 0.8189
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 1
```

```
 Mcnemar's Test P-Value : NA

            Sensitivity : 1.0000
            Specificity : 1.0000
         Pos Pred Value : 1.0000
         Neg Pred Value : 1.0000
             Prevalence : 0.1811
         Detection Rate : 0.1811
   Detection Prevalence : 0.1811
      Balanced Accuracy : 1.0000

       'Positive' Class : 1
```

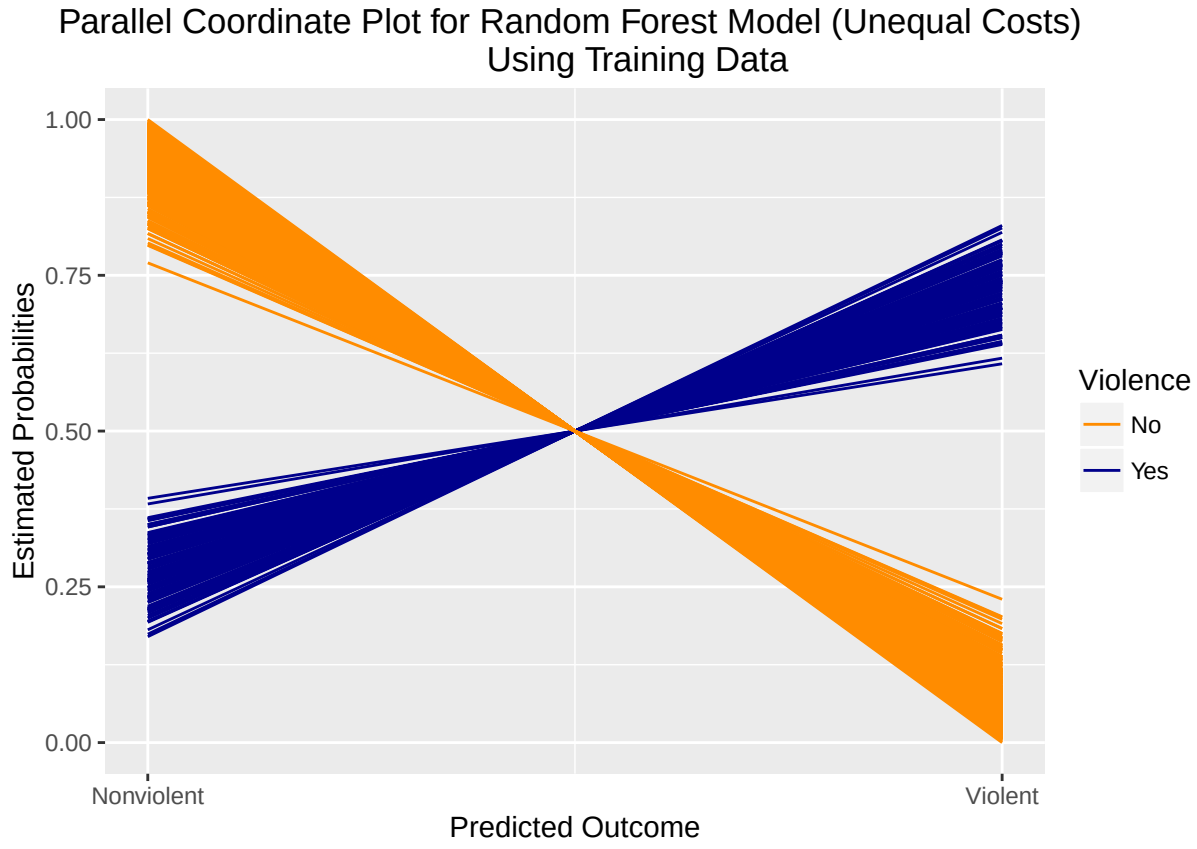
```
# cross-validated error
confusionMatrix(predict(covrRF, COVRtest), COVRtest$Violence, positive = '1')
```

```
Confusion Matrix and Statistics

          Reference
Prediction   0   1
         0 217  42
         1   8  15

               Accuracy : 0.8227
                 95% CI : (0.773, 0.8654)
    No Information Rate : 0.7979
    P-Value [Acc > NIR] : 0.1678

                  Kappa : 0.2928
 Mcnemar's Test P-Value : 3.058e-06

            Sensitivity : 0.26316
            Specificity : 0.96444
         Pos Pred Value : 0.65217
         Neg Pred Value : 0.83784
             Prevalence : 0.20213
         Detection Rate : 0.05319
   Detection Prevalence : 0.08156
      Balanced Accuracy : 0.61380

       'Positive' Class : 1
```

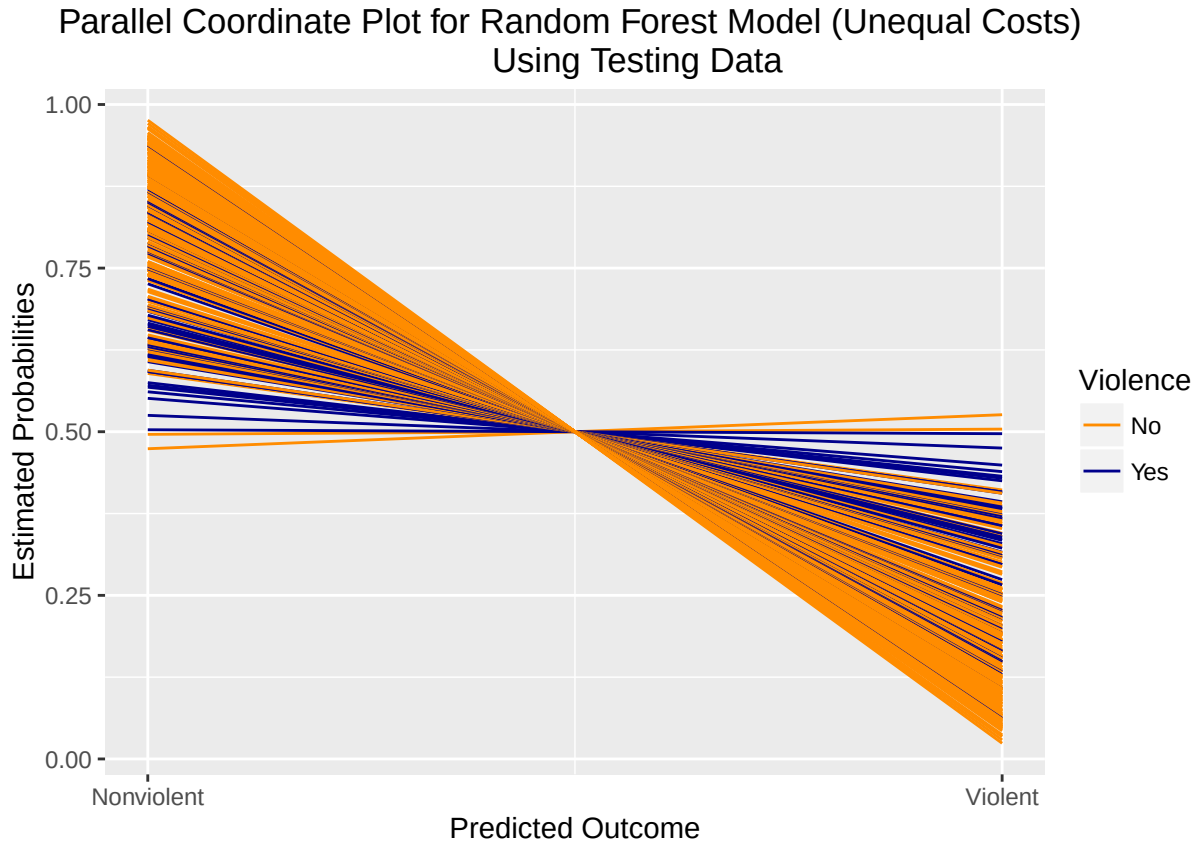
And the parallel coordinate plots:

```
# data frame of estimated probabilities of violence and actual classification
rfPreds = data.frame(Preds = predict(covrRF, COVRtrain, type = 'prob'),
                     select(COVRtrain, Violence))
rfPreds = arrange(rfPreds, Preds.0)
ggplot(data = rfPreds) +
     geom_segment(aes(x = 0, xend = 1, y = Preds.0, yend = Preds.1, color = Violence)) +
     scale_x_continuous(breaks = c(0, 1),
```

```
                        labels = c('Nonviolent', 'Violent')) +
    xlab('Predicted Outcome') +
    ylab('Estimated Probabilities') +
    scale_color_manual(values = c('darkorange', 'darkblue'),
                        labels = c('No', 'Yes')) +
    ggtitle('Parallel Coordinate Plot for Random Forest Model (Unequal Costs)
            Using Training Data')
```




```
# remove rfPreds
rm(rfPreds)

# data frame of estimated probabilities of violence and actual classification
rfPreds = data.frame(Preds = predict(covrRF, COVRtest, type = 'prob'),
                     select(COVRtest, Violence))
rfPreds = arrange(rfPreds, Preds.0)
ggplot(data = rfPreds) +
    geom_segment(aes(x = 0, xend = 1, y = Preds.0, yend = Preds.1, color = Violence)) +
    scale_x_continuous(breaks = c(0, 1),
                        labels = c('Nonviolent', 'Violent')) +
    xlab('Predicted Outcome') +
    ylab('Estimated Probabilities') +
    scale_color_manual(values = c('darkorange', 'darkblue'),
                        labels = c('No', 'Yes')) +
    ggtitle('Parallel Coordinate Plot for Random Forest Model (Unequal Costs)
            Using Testing Data')
```



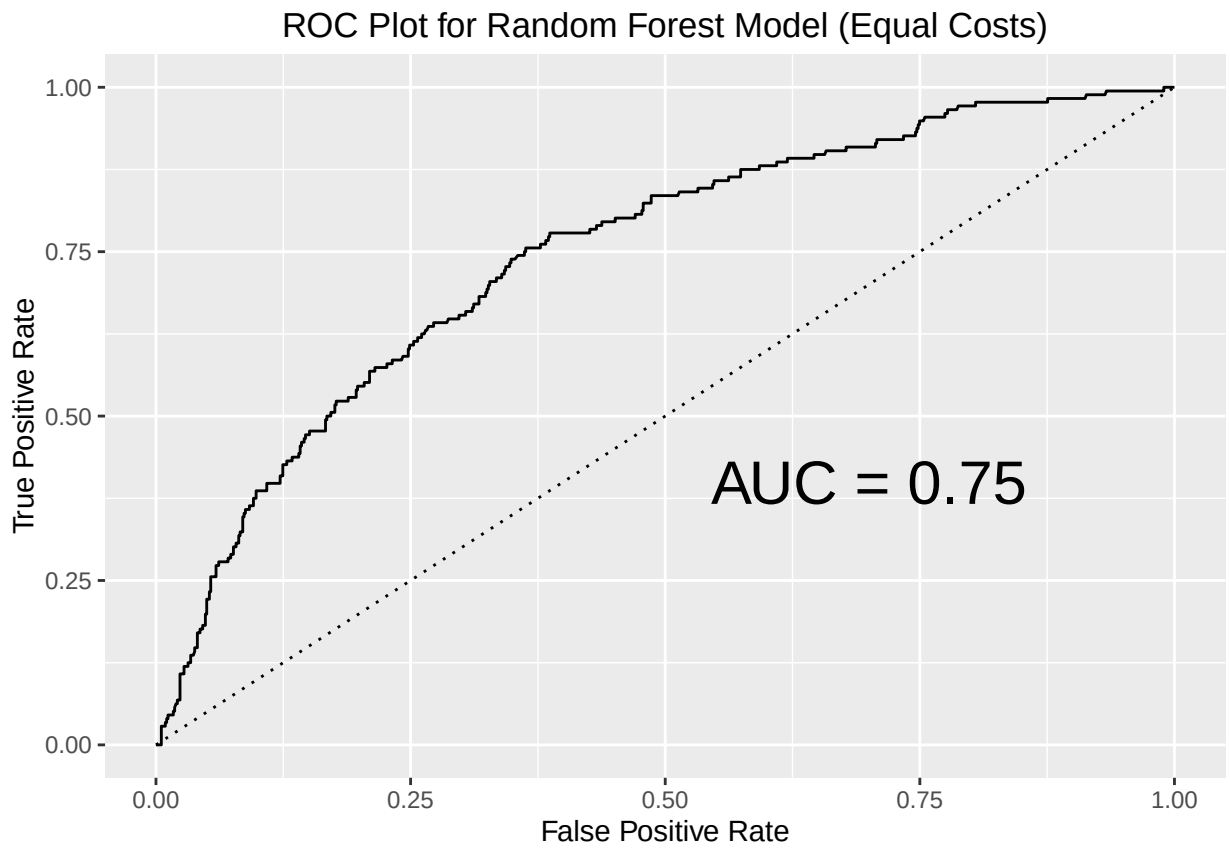
Now a random forest model with all the data and using the OOB error to estimate test error. First with equal costs.

```
# Clear workspace
rm(list = ls()[!(ls() %in% c('BR', 'COVRdata', 'COVRdataRF'))])
# Equal costs
set.seed(1983917)
covrRF = randomForest(Violence ~ ., data = COVRdataRF, ntree = 1000)
confusionMatrix((covrRF$votes > .5)[,2], COVRdataRF$Violence == 1, positive = 'TRUE')
```

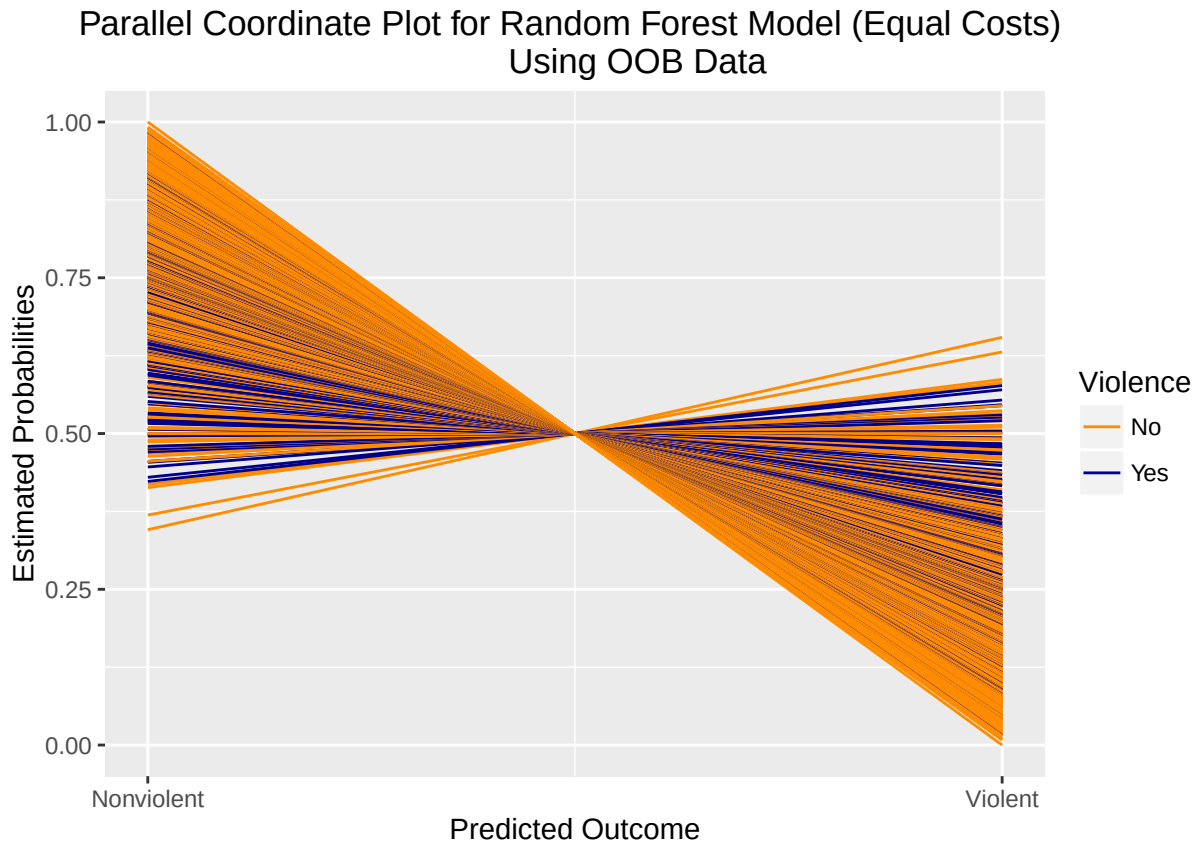
```
Confusion Matrix and Statistics

          Reference
Prediction FALSE TRUE
     FALSE   749  166
     TRUE     14   10

               Accuracy : 0.8083
                 95% CI : (0.7816, 0.833)
    No Information Rate : 0.8126
    P-Value [Acc > NIR] : 0.6494

                  Kappa : 0.0576
 Mcnemar's Test P-Value : <2e-16

            Sensitivity : 0.05682
```

```
            Specificity : 0.98165
         Pos Pred Value : 0.41667
         Neg Pred Value : 0.81858
             Prevalence : 0.18743
         Detection Rate : 0.01065
   Detection Prevalence : 0.02556
      Balanced Accuracy : 0.51923

       'Positive' Class : TRUE
```

```
# ROC plot
rfPreds = prediction(covrRF$votes[,2], select(COVRdataRF, Violence))
rfPerf = performance(rfPreds, 'tpr', 'fpr')
AUC = round(performance(rfPreds, 'auc')@y.values[[1]], 2)
ggplot(data = NULL) +
     geom_line(aes(x = rfPerf@x.values[[1]],
                                y = rfPerf@y.values[[1]])) +
     ggtitle('ROC Plot for Random Forest Model (Equal Costs)') +
     xlab('False Positive Rate') +
     ylab('True Positive Rate') +
     geom_segment(aes(x = 0, y = 0, xend = 1, yend = 1),
                  linetype = 'dotted') +
     geom_text(aes(x = .7, y = .4, label = paste0('AUC = ', AUC), parse = T),
               size = 8)
```



```
# Parallel coordinate plot
rfPreds = data.frame(Preds = covrRF$votes, select(COVRdataRF, Violence))
rfPreds = arrange(rfPreds, Preds.0)
ggplot(data = rfPreds) +
     geom_segment(aes(x = 0, xend = 1, y = Preds.0, yend = Preds.1, color = Violence)) +
     scale_x_continuous(breaks = c(0, 1),
                        labels = c('Nonviolent', 'Violent')) +
     xlab('Predicted Outcome') +
     ylab('Estimated Probabilities') +
     scale_color_manual(values = c('darkorange', 'darkblue'),
                        labels = c('No', 'Yes')) +
     ggtitle('Parallel Coordinate Plot for Random Forest Model (Equal Costs)
             Using OOB Data')
```



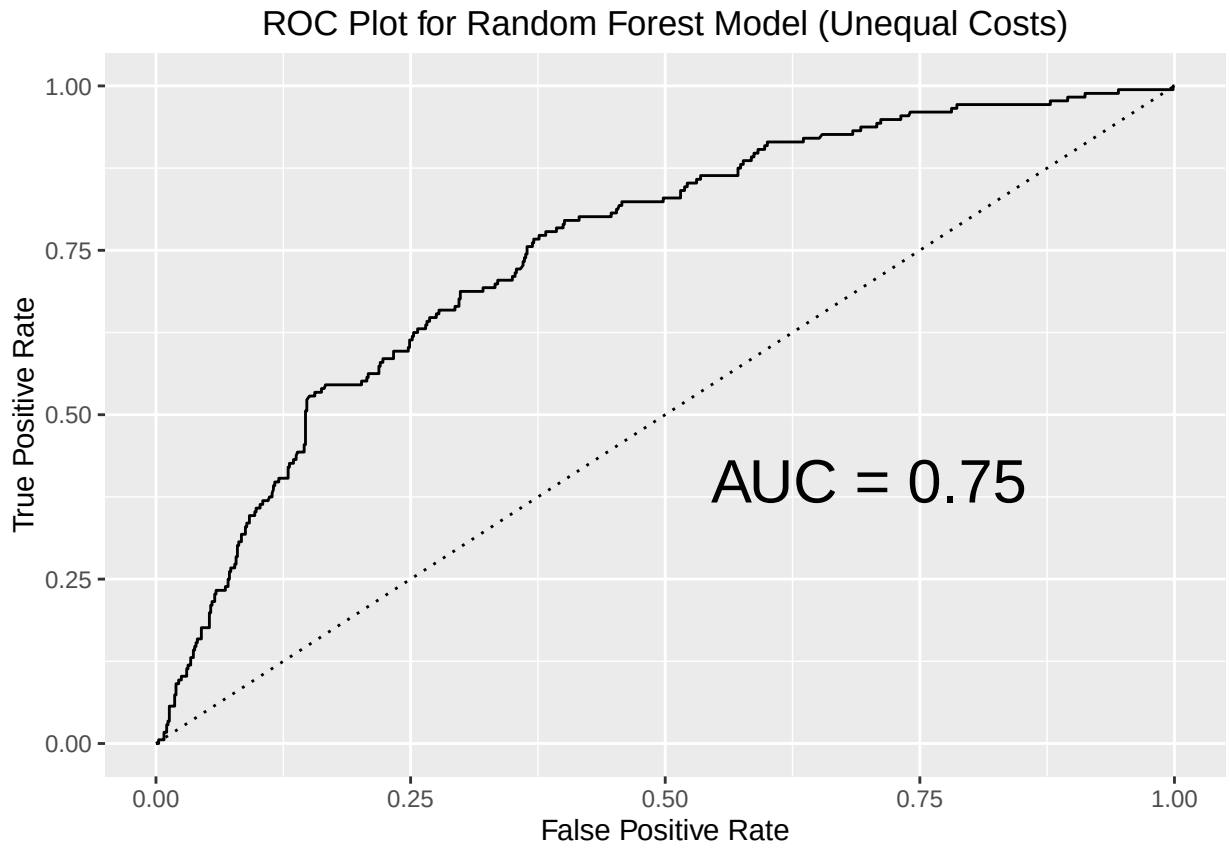
Now with unequal costs.

```
rm(covrRF, rfPreds, rfPerf, AUC)
set.seed(1983917)
covrRF = randomForest(Violence ~ ., data = COVRdataRF, ntree = 1000,
                      cutoff = c(1-2*BR, 2*BR))
confusionMatrix((covrRF$votes > .5)[,2], COVRdataRF$Violence == 1, positive = 'TRUE')
```

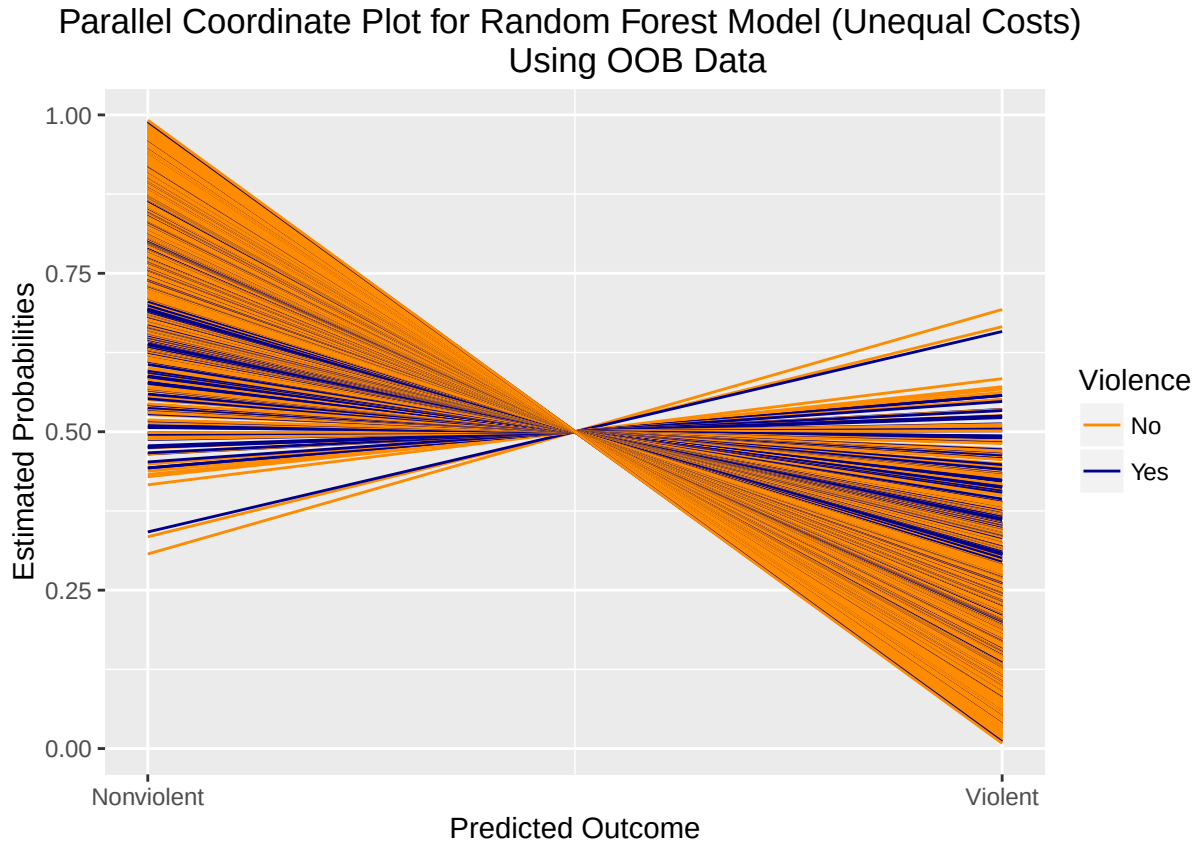
```
Confusion Matrix and Statistics

          Reference
```

```
Prediction FALSE TRUE
     FALSE   748  163
     TRUE     15   13

               Accuracy : 0.8104
                 95% CI : (0.7839, 0.835)
    No Information Rate : 0.8126
    P-Value [Acc > NIR] : 0.586

                  Kappa : 0.0801
 Mcnemar's Test P-Value : <2e-16

            Sensitivity : 0.07386
            Specificity : 0.98034
         Pos Pred Value : 0.46429
         Neg Pred Value : 0.82108
             Prevalence : 0.18743
         Detection Rate : 0.01384
   Detection Prevalence : 0.02982
      Balanced Accuracy : 0.52710

       'Positive' Class : TRUE
```

```r
# ROC plot
rfPreds = prediction(covrRF$votes[,2], select(COVRdataRF, Violence))
rfPerf = performance(rfPreds, 'tpr', 'fpr')
AUC = round(performance(rfPreds, 'auc')@y.values[[1]], 2)
ggplot(data = NULL) +
     geom_line(aes(x = rfPerf@x.values[[1]],
                                y = rfPerf@y.values[[1]])) +
     ggtitle('ROC Plot for Random Forest Model (Unequal Costs)') +
     xlab('False Positive Rate') +
     ylab('True Positive Rate') +
     geom_segment(aes(x = 0, y = 0, xend = 1, yend = 1),
                  linetype = 'dotted') +
     geom_text(aes(x = .7, y = .4, label = paste0('AUC = ', AUC), parse = T),
               size = 8)
```

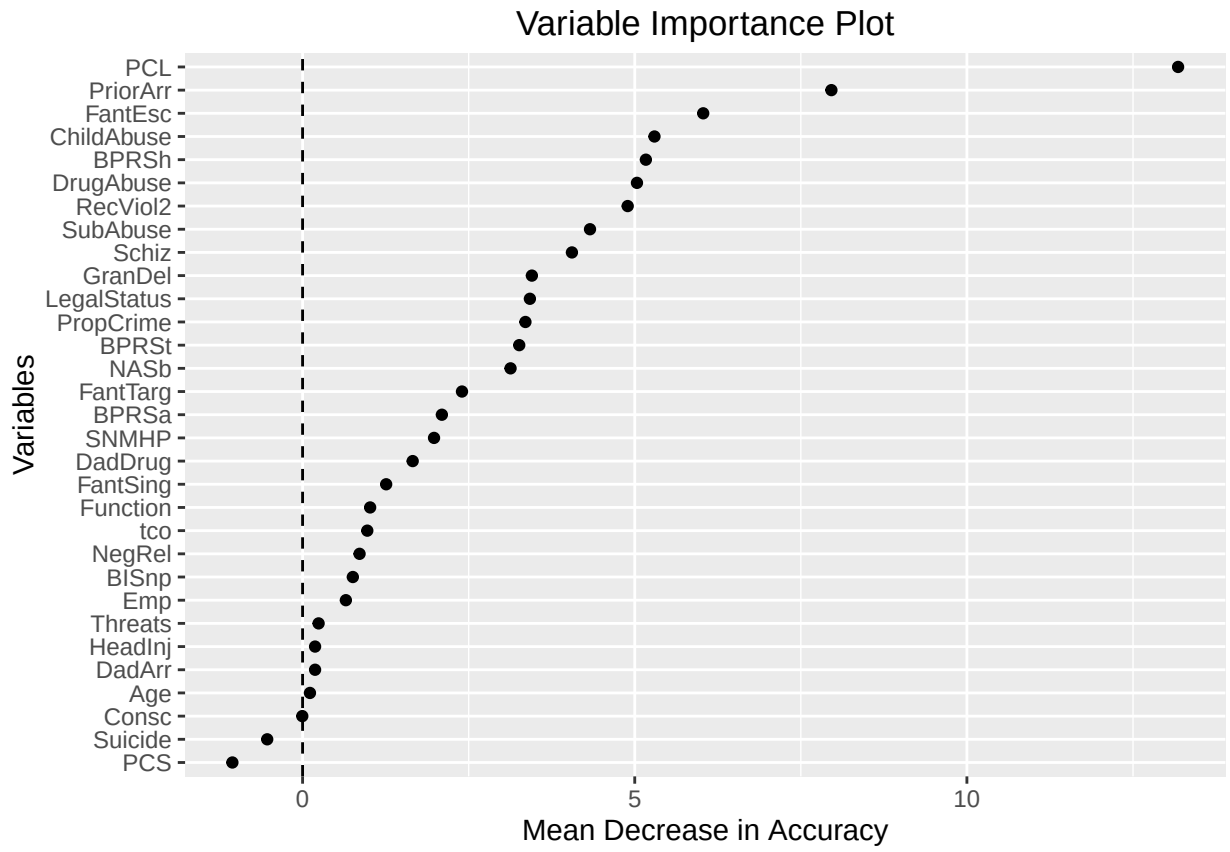



```
# Parallel coordinate plot
rfPreds = data.frame(Preds = covrRF$votes, select(COVRdataRF, Violence))
rfPreds = arrange(rfPreds, Preds.0)
ggplot(data = rfPreds) +
     geom_segment(aes(x = 0, xend = 1, y = Preds.0, yend = Preds.1, color = Violence)) +
     scale_x_continuous(breaks = c(0, 1),
                        labels = c('Nonviolent', 'Violent')) +
     xlab('Predicted Outcome') +
     ylab('Estimated Probabilities') +
     scale_color_manual(values = c('darkorange', 'darkblue'),
                        labels = c('No', 'Yes')) +
     ggtitle('Parallel Coordinate Plot for Random Forest Model (Unequal Costs)
             Using OOB Data')
```



Finally, using the OOB error to measure variable importance, we use a subset of variables for prediction, eliminating those that negatively affect or add very little to the accuracy. This is the "final model."

```
set.seed(1983917)
# measuring variable importance
covrRF = randomForest(Violence ~ ., data = COVRdataRF, ntrees = 1000, imp = T)

# variable importance plots
varImp = data_frame(Variables = rownames(importance(covrRF)),
                    Accuracy = importance(covrRF)[,3])
ggplot(data = varImp, aes(Accuracy, reorder(Variables, Accuracy))) +
     geom_point() +
     geom_vline(xintercept = 0, lty = 2) +
     xlab('Mean Decrease in Accuracy') +
     ylab('Variables') +
     ggtitle('Variable Importance Plot')
```

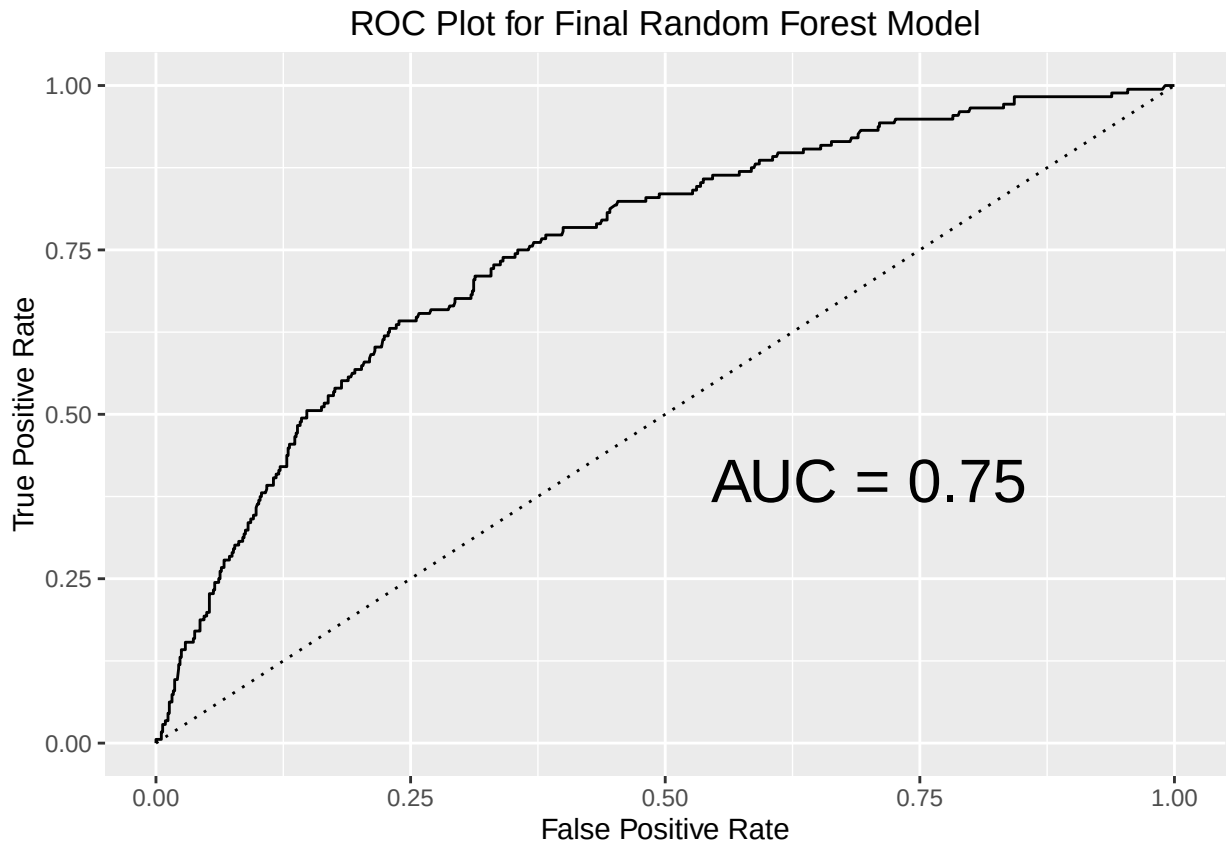



```
# select subset of variables that whose mean decrease in accuracy is greater than .5
COVRdataRF_Imp = select(COVRdataRF, Violence, (2:32)[select(varImp, Accuracy) > .5])

# final model
set.seed(1983917)
covrRF = randomForest(Violence ~ ., data = COVRdataRF_Imp, ntrees = 1000)
confusionMatrix((covrRF$votes > .5)[,2], COVRdataRF$Violence == 1, positive = 'TRUE')
```

```
Confusion Matrix and Statistics

          Reference
Prediction FALSE TRUE
     FALSE   747  159
     TRUE     16   17

               Accuracy : 0.8136
                 95% CI : (0.7872, 0.8381)
    No Information Rate : 0.8126
    P-Value [Acc > NIR] : 0.4868

                  Kappa : 0.11
 Mcnemar's Test P-Value : <2e-16

            Sensitivity : 0.09659
            Specificity : 0.97903
         Pos Pred Value : 0.51515
```

```
         Neg Pred Value : 0.82450
             Prevalence : 0.18743
         Detection Rate : 0.01810
   Detection Prevalence : 0.03514
      Balanced Accuracy : 0.53781

       'Positive' Class : TRUE
```

```
# ROC plot
rfPreds = prediction(covrRF$votes[,2], select(COVRdataRF, Violence))
rfPerf = performance(rfPreds, 'tpr', 'fpr')
AUC = round(performance(rfPreds, 'auc')@y.values[[1]], 2)
ggplot(data = NULL) +
     geom_line(aes(x = rfPerf@x.values[[1]],
                                 y = rfPerf@y.values[[1]])) +
     ggtitle('ROC Plot for Final Random Forest Model') +
     xlab('False Positive Rate') +
     ylab('True Positive Rate') +
     geom_segment(aes(x = 0, y = 0, xend = 1, yend = 1),
                  linetype = 'dotted') +
     geom_text(aes(x = .7, y = .4, label = paste0('AUC = ', AUC), parse = T),
               size = 8)
```



```
# Parallel coordinate plot
rfPreds = data.frame(Preds = covrRF$votes, select(COVRdataRF, Violence))
rfPreds = arrange(rfPreds, Preds.0)
ggplot(data = rfPreds) +
     geom_segment(aes(x = 0, xend = 1, y = Preds.0, yend = Preds.1, color = Violence)) +
     scale_x_continuous(breaks = c(0, 1),
                        labels = c('Nonviolent', 'Violent')) +
     xlab('Predicted Outcome') +
     ylab('Estimated Probabilities') +
     scale_color_manual(values = c('darkorange', 'darkblue'),
                        labels = c('No', 'Yes')) +
     ggtitle('Parallel Coordinate Plot for Final Random Forest Model')
```