Analysis of Robust Measures in Random Forest Regression

MAJ John R. Brence, Ph.D.
Adjunct Assistant Professor
Department of Systems Engineering
United States Military Academy
West Point, NY 10996
845-304-6416
john.brence@usma.edu

Donald E. Brown, Ph.D.
Department Chair and Professor
Department of Systems and Information Engineering
University of Virginia
Charlottesville, VA 22903
434-924-5393
brown@virginia.edu

ABSTRACT

Analysis of robust measures in Random Forest Regression (RFR) is an extensive empirical analysis on a new method, Robust Random Forest Regression (RRFR). The application and analysis of this tree-based method has yet to be addressed and may provide additional insight in modeling complex data. Our approach is based on the RFR with two major differences ~ the introduction of robust prediction and error statistic. The current methodology utilizes the node mean for prediction and mean squared error (MSE) to derive the in-node and overall error. Herein, we introduce and assess the use of a median for prediction and mean absolute deviation (MAD) to derive the in-node and overall error. Extensive research has shown that the median is a better prediction of the centrality of the distribution in the presence of large or unbounded outliers because the median inherently ignores these outliers basing its prediction on the ordered, central value(s) of the data. We have shown that RRFR performs well under extreme conditions; with datasets that include unbounded outliers or heteroscedastic conditions.

KEY WORDS

Random Forest, Outlier, Robust Statistics, Regression, Tree-based Methods

1 INTRODUCTION

The use of robust measures provides an interesting basis for this study. Robust statistics is generally thought of as the statistics of approximate parametric models (Hampel et.al.,1986). Using robust statistics allows us to explore relatively dirty datasets without requiring the somewhat archaic method of removing outliers or strange observations from a dataset prior to modeling. Application of robust measures to nonparametric models allows us to forgo our strict adherence to the usual statistical assumptions such as normality and linearity; however, nonparametric methods maintain a rather weak yet stringent adherence to the continuity of distribution or independence assumption (Hampel et.al., 1986). In this sense, robust theory provides us the ability to be creative in our approach and assists in determining the usefulness of applying robust measures to the RFR algorithm. This is extremely valuable because many of the nonparametric algorithms used today are naturally robust in some instances. In this vein, if a

somewhat-robust nonparametric algorithm is paired with correct robust measures, the resultant methodology may reign supreme as the best new modeling tool on the street. The possibility for this predictive improvement (or dominance) on a specific domain of problems provokes our desire to comprehensively explore the RRFR application.

Many related methods include some form of robust measure to handle complex datasets and the presence of outliers (Breiman et.al. 1984; Chadhuri et.al., 2001; Friedman. 1991; Loh, 2002; Torgo, 1999). This study performs an empirical analysis on a new method, RRFR, in comparison to RFR which did not include these measures in order to assess the efficacy of robust measures. The results of this analysis directly address the need for better predictive capabilities in modeling real world datasets. For example, both safety and scientific related datasets which inherently possess noisy data and, many times, are embedded with outliers.

One particular approach that has been successful in coping with complex datasets is to use nonparametric approaches that have the capacity to accommodate a wide variety of regression domains, principally by not imposing any pre-defined global form to the regression surface (Härdle, 1990). They do, however, assume a functional form at the local level; meaning that they do not fit a single, overall model to the entire distribution of data (Torgo, 1999).

Mean squared error is known to be very sensitive to outliers. Outliers may significantly distort the mean squared error statistic which would create a model that fails to capture the essence of the data. The squared differences in the mean squared error statistic tend to amplify the error of an outlier. Additionally, the presence of outliers greatly influences the mean, thereby making a prediction that is not representative of the associated training set.

The median, paired with mean absolute deviation, represent better (robust) statistics of centrality than the mean and mean squared error for skewed distributions (Torgo, 1999). The median tends to neglect some of the information contained in the sample; generally ignoring the outliers (Hoaglin et.al, 1983). When the underlying distribution of data is approximately normal, the median and mean have near equivalent statistical properties.

Figure 1 (Hampel et.al., 1986) illustrates a useful metaphor for why we use robust measures when analyzing data. When the tight-rope walker depends on least squares method to model (a), it is analogous to accepting high degree of risk by using only a pole for stability. As the tight-rope walker employs additional safety measures such as a net (b) or ladder (c), it is analogous to a modeler invoking methods for coping with outliers (b), and/or adopting robust statistical measures (c). In both situations, the risk of unintended consequences due to error diminishes.

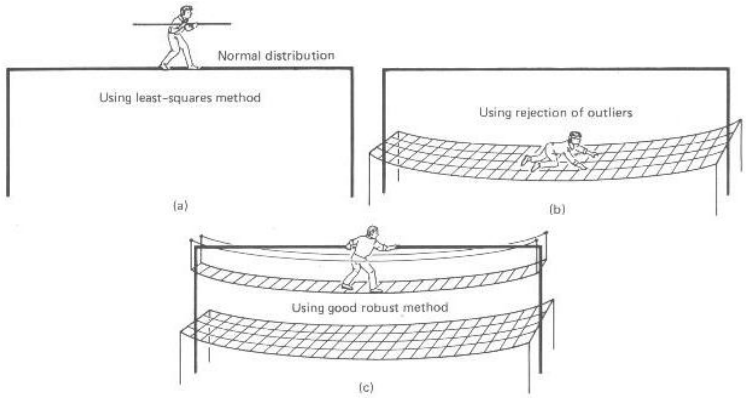

Figure 1: Various ways of analyzing data (Hampel et.al., 1986)

This is not to assert that robust statistics are flawless since a median is known to be sensitive to small errors in the data resulting from rounding or grouping that cause order differences in the central data values. *However, the effect of such errors diminishes when the robust statistic-based estimator averages over several observations* (Hoaglin et.al, 1983). This notion provides quite an appealing characteristic in the context of Random Forest Regression.

2 RANDOM FOREST REGRESSION

Random Forests were introduced in 2000 by Breiman (Breiman, 2000). This algorithm creates a classification tree forest. A single tree is created by using a bootstrap sample of the training dataset and bagging predictors in order to facilitate more efficient node-splitting. Bagging (Bootstrap Aggregation) is the generation of multiple (random) versions of a predictor in order to combine into an aggregated prediction. Bagging creates the forest prediction by combining numerous trees in order to take advantage of their predictive power (Breiman, 2001). Use of the Strong Law of Large Numbers indicates that the trees always converge (only if the mean is finite [exists]); therefore there is no issue with overfitting the data (Breiman, 2001).

In April of 2001, Breiman introduced code for Random Forest Regression (RFR). This method mirrored the previous classification algorithm (Random Forests); however, it predicted and calculated error using the terminal node means and mean squared error. The thesis of this method is that complex predictors provide a wealth of interpretable scientific information about the data and how it is predicted. In addition, among the current prediction methods, the most complex methods are the most accurate (Breiman, 2000).

Robust Random Forest Regression embeds the methodology of Random Forest Regression (Breiman, 2001) with a major difference ~ the introduction of robust prediction and error statistics using the median and other robust measures for prediction, and the use of mean absolute deviation (MAD) to calculate both the in-node and overall error. By adapting this new strategy, we theoretically lessen the effects of outliers and heteroscedasticity while retaining the beneficial components of RFR as well. RFR currently uses the node mean for prediction and mean squared error (MSE) to derive the in-node and overall error.

Definition: A random forest (regression) is a predictor consisting of a collection of tree-structure predictors $\left\{h(\mathbf{x},\Theta_z)_\psi, z=1...Z\right\}$ in which each tree casts an equal valued vote for the prediction at input vector $\mathbf{x}$ and where Θ_z are independently identically distributed random vectors whose elements are counts on the number of times an input row appears in the bootstrap sample (Breiman, 2001). z is the index for the tree in the forest and Z is the total number of trees in the forest. ψ is a statistic for central tendency, which could be the mean, median, or other measure.

For the z^{th} tree, we generate a random vector Θ_z, independent of the previous random vectors $\Theta_1,...,\Theta_{z-1}$ but with the same distribution; and we grow a tree using the training set and Θ_z, resulting in the predictor $h(\mathbf{x},\Theta_z)_\psi$ where $\mathbf{x}$ is an input vector (Breiman, 2001).

The prediction based on the vectors $\mathbf{x},\Theta_z$ is conducted in two steps. First, at the tree level, the prediction $h(\mathbf{x},\Theta_z)_\psi$ is based on the statistic ψ chosen to determine central tendency of the observations within each terminal node ~ mean for RFR and median, trimean, broadened

median or trimmed mean for RRFR. Second, all the tree predictions are combined to create a forest prediction $\left\{h(\mathbf{x},\Theta_z)_\psi, z=1,...Z\right\}$ using *bagging* (bootstrap-aggregation).

3 ROBUST RANDOM FOREST REGRESSION

Robust Random Forest Regression applies a similar methodology to Random Forest Regression and hence, to the construction of Regression Trees. Each forest contains a user-defined amount of trees that facilitates a better prediction of a response than a single tree would provide, while not overfitting the training dataset in the process.

3.1 RRFR at the Tree Level

Much of the theory related to Regression Trees (Breiman et.al, 1984) applies to RRFR at the tree level with some notable exceptions. In Regression Trees, the predictor space **X** is divided into subsequent children nodes using binary splits until a terminal node threshold is attained. At each terminal node m, the in-node prediction of the response $\hat{y}_m$ remains constant. The prediction method within the node is determined by the user. Notable choices include bootstrapping, tree building stopping rules, and bagging.

3.1.1 Bootstrapping

One major difference between RRFR and Regression Trees is that the predictor space used for each tree in the forest generated by RRFR is a bootstrapped sample of $\mathbf{x}$, denoted by Θ_z. Bootstrapping is a method that randomly samples with replacement the original training dataset while maintaining the original sample number of observations (Figure 2). This means that many of the original training observations may be deleted, while other observations may become duplicates, triplicates, or higher degree replications of the dataset.

The intent of RRFR in tree-building is to have a forest of trees that model observations from the training dataset without undue replication of data. This goal accounts for one major reason why RRFR will not overfit the training dataset: it is highly unlikely that every tree will have the same bootstrapped sample and even more unlikely that these samples are equivalent to the raw training dataset.

```
For a count of 1 to N_Tng (number of training observations)
    Choose a Random number between 1 and N_Tng
    Flag current count as a bootstrapped sample
    Make the random-selected observation y equal to the bootstrapped y

        For a count of 1 to number of predictors
          Make the random-selected observation x_i equal to bootstrapped x_i
        End (Predictor Loop)

End (Observation Loop)
```

Figure 2: Bootstrapping Pseudo-code

From the bootstrapped sample, Θ_z, RRFR needs to decide how to derive a single tree and the related terminal node predictions. Similar to Regression Trees, this is done by answering three important questions: 1) How do we split at every parent (intermediate) node? 2) How do we know when to stop splitting and declare a node terminal? and 3) Within that terminal node, how will we assign the prediction, $\hat{y}_m$?

3.1.2 Splitting the Parent Node

The goal of the tree-building process in RRFR is to minimize the prediction error realized at each split. Regression Trees employ a greedy search method to identify the best predictor variable x_i to use as a splitting variable x_i*, and its subsequent value $s*$. RRFR has the same goal. However, in another attempt to grow unique trees for the forest, it *randomly* selects from a subset of all the variables $x_i, i = 1,...,I$. The user is responsible for selection of the size, I_m, of the subset of $x_i, i = 1,...,I$, to try at each node m. At every split iteration, I_m variables are randomly chosen for splitting. Notice that because the selected x_i is random, it is possible for a variable to be chosen multiple times. For each I_m variables, the same splitting criteria apply as in Regression Trees.

For any parent node m that is split into the child nodes m_L and m_R, the split s, of the set of splits, S, attempts to minimize the error by maximizing the decrease in resubstitution error $R(s,m)$ or $\Delta R(s*,m) = \max_{s\in S} \Delta R(s,m)$, which equates to $\underset{s\in S}{Max}\left[R(m) - R(m_L) - R(m_R)\right]$. This best split, $s*$, is found by enumerating and evaluating all the split possibilities for the subset of variables considered at that node. It is complete when the maximizing split value $s*$ is found for x_i*. The split variable x_i* is typically the variable that achieves the best separation of the high response values from the lower ones, hence leading to lower within-node error.

3.1.3 Terminal Nodes and when to stop splitting

When compared to the Regression Tree discussion in (Breiman et.al, 1984), the criterion used to stop splitting and declare a node terminal used in RRFR is simplistic. Both methods follow the rule that for any split of a parent node, m, into child nodes m_L and m_R, the parent node error must be greater than or equal to the sum of the children errors, or $R(m) \geq R(m_L) + R(m_R)$. If any parent node fails this criterion, it is determined to be a terminal node.

Regression trees struggle with the discussion of how to prune the tree in order to minimize error and complexity to develop the best (general) tree model. RRFR simply includes a user input parameter, $N_{m_{threshold}}$, that serves as a threshold for the minimum number of observations in a parent node and assists in the calculation of the total number of nodes in a tree. The use of $N_{m_{threshold}}$ creates two easy ways to determine if a node is terminal. First, after a parent node is split each child node is tested for potential parenthood. If that child node has less than or equal to $N_{m_{threshold}}$ observations within the node, it is now deemed terminal, otherwise it becomes a

parent ready for splitting. Secondly, the maximum number of potential nodes in a tree is determined by the equation:

$$M = 2 * (N_{Tng} / N_{m_{threshold}}) + 1$$

Equation 1: Number of Allowed Nodes in a Tree

where M is the maximum possible nodes in a tree, N_{Tng} is the number of observations used from the training dataset used to create the model, and $N_{m_{threshold}}$ is the threshold for the minimum number of observations in a parent node. The scalar "2" in Equation 1 accounts for the binary splits of parent nodes and the (+1) represents the root node or the first (parent) node of the tree. In the tree creation process, if the sum of the tree nodes is equivalent to M , the algorithm will declare any childless parent nodes as terminal nodes.

3.1.4 Prediction within a Terminal Node

The prediction in a terminal node is simply the prediction statistic value for that specific terminal node. In the Regression Tree literature (Breiman et.al, 1984), two methods of prediction are discussed: mean and median. Random Forest Regression, the foundation on which the RRFR algorithm is built, uses only the mean for prediction. RRFR gives the user additional robust prediction methods to choose from. A brief explanation of these robust methods follows.

3.1.4.1 Median

An extremely simple *L*-estimator that predicts by weighting only the centermost observation or center two observations relative to whether the number of sample observations is odd or even, respectively (Hoaglin et.al., 1983) is

$$\tilde{y} = \begin{cases} Obs_{(N-1)/2+1} \; ; & \text{N odd} \\ \dfrac{Obs_{N/2} + Obs_{N/2+1}}{2} \; ; & \text{N even} \end{cases}$$

Equation 2: Definition of Median

The median, represented as $\tilde{y}$, may also be calculated as a trimmed mean. In this manner, the median is dependent upon the total number of observations, *N*, for a trimming proportion, α , equal to

$$\alpha = 1/2 - 1/2N$$

Equation 3: Trimming Proportion for a Median as a Trimmed Mean (Hoaglin et.al., 1983)

3.1.4.2 Broadened Median

The broadened median is a special *L*-estimator used to preserve the characteristics of a median while also achieving insensitivity to rounding and grouping of observations (Hoaglin et.al., 1983). This predictor not only uses the median as part of its prediction, but also takes into account neighboring observations based on the total number of observations, *N*. These values are then averaged for the overall prediction.

Definition: For *N* odd, the broadened median is the average of the three central order data points for $5 \le N \le 12$; the five central order data points for $N \ge 13$. For *N* even it is the weighted average of the central four order data points for $5 \le N \le 12$ with weights $1/6, 1/3, 1/3, and\ 1/6$; for $N \ge 13$, it is the weighted average of the six central data points with weights $1/5$ for the central four and $1/10$ for the two end data points. NOTE: The broadened median is not defined for $N < 5$; when this occurred, we substituted the median.

The broadened median may also be modeled as a trimmed mean that is dependent the number of observations *N* for a trimming proportion α equal to

$$\alpha = \left(0.5 - 1.5/N\right) \quad \text{for } 5 \le N \le 12$$

AND

$$\alpha = \left(0.5 - 2.5/N\right) \quad \text{for } N \ge 13$$

Equation 4: Trimming Proportion for a Broadened Median as a Trimmed Mean (Hoaglin et.al., 1983)

3.1.4.3 Trimean

The trimean is one of the easier *L*-estimators to calculate (Hoaglin et.al., 1983). It has a similar motivation as the broadened median, but is functionally simpler to calculate. The trimean also wishes to preserve the resistance of the median while protecting against any rounding and grouping of observations. The trimean is implemented to include sample information farther away from the center of the ordered distribution. By using the upper and lower fourth observations in the data, it can also protect against errors centered on the median.

Definition: If F_L and F_U are the lower and upper fourths of the data sample and $\tilde{y}$ is the sample median (the 25th, 75th and 50th percentile respectively); then the trimean is calculated by:

$$TRI = 1/4\left(F_L + 2\tilde{y} + F_U\right)$$

Equation 5: Definition of Trimean

The Trimean cannot be expressed as trimmed mean due to the nature of the calculation in Equation 5.

3.1.4.4 Trimmed Mean

The trimmed mean was developed because of the phenomenon that measured data may be "precise (clustered tightly around some value) without being accurate (Hoaglin et.al., 1983)." In this case, accuracy is being expressed as to how well the measurements depict the true characteristics or parameters of the underlying distribution. Therefore, sometimes it is necessary to trim a percentage of the observations used to describe the distribution and focus the prediction on the more central observations of that distribution.

The Trimmed mean is the mean of the set of *N* ordered statistics after trimming a proportion α from both ends of the set. The value $Trimmed(\alpha)$ is returned as the mean of the trimmed set of ordered statistics. In this manner, the 0% trimmed mean is the sample mean and a 50% trimmed mean is approximately the median (Equation 2). The weights given to the ordered statistics and the formula for computing of the trimmed mean (Equation 6) and weights (Equation 7) are given by

$$Trimmed(\alpha) = \frac{1}{n(1-2\alpha)}\left\{(1-r)\left[X_{(g+1)} + X_{(n-g)}\right] + \sum_{i=g+2}^{n-g-1} X_{(i)}\right\}$$

Equation 6: Definition of Trimmed Mean

$$a_i = \left\{\begin{array}{ll} 0; & \text{if } \; i \le g \;\text{ or }\; i \ge (n-g+1) \\ \dfrac{(1-r)}{n(1-2\alpha)}; & \text{if } \; i = g+1 \;\text{ or }\; i = n-g \\ \dfrac{1}{n(1-2\alpha)}; & \text{if } \; (g+2) \le i \le (n-g-1) \end{array}\right\}$$

Equation 7: Trimmed Mean Weights

where $X_{(1)} \le X_{(2)} \le ... \le X_{(n)}$ are the sample ordered statistics of size, *n*, $g = \lfloor \alpha n \rfloor$ (known as a floor function) denoting the greatest integer less than or equal to αn for $0 \le \alpha \le 0.5$, so that $r = \alpha n - \lfloor \alpha n \rfloor = \alpha n - g$ which the fractional part of αn.

3.1.5 Handling Categorical Predictor Variables

RRFR handles categorical predictor variables in the same fashion as Regression Trees and Random Forest Regression. Categorical variables are discrete variables that lack any particular ordering. For example, if variable x_i was used to denote the city categories of Seattle, New York, Boston and Charlottesville, there is no set order to these categorical levels.

In general, each categorical variable has various category levels $x_i \in \{c_1, c_2, ..., c_L\}$ and $L =$ the number of levels. For any categorical variable $x_i \in \{c_1, c_2, ..., c_L\}$ in node m, we define $\bar{y}(c_\ell)$ as the average over all y_n in the node such that the level of x_i is c_ℓ. We order these results in the following manner

$$\bar{y}(c_{\ell_1}) \le \bar{y}(c_{\ell_2}) \le ... \le \bar{y}(c_{\ell_L})$$

Equation 8: Categorical Variable Ordering to Aid in Splitting (Breiman et. al., 1984)

The best split on x_i in node m is one of the potential $L-1$ splits. This methodology mirrors both Regression Trees and Random Forest Regression and is denoted as

$$x_i \in \{c_{\ell_1}, c_{\ell_2}, ..., c_{\ell_{L-1},}\}$$

Equation 9: Best Categorical Split Result (Breiman et. al., 1984)

It is also noted that one benefit of this methodology is that is reduces the search for the best split from $2^{L-1}-1$ to $L-1$ categorical levels (Breiman et. al., 1984). However, this approach does not consider all possible splits on this variable.

3.1.6 Cross Validation

Unlike Regression Trees, yet similar to Random Forest Regression, RRFR does not use cross validation for pruning or as a measure to protect against overfitting the training dataset. Cross validation is not implemented because there are other methodologies in-place that serve the same purpose. In particular, RFR and RRFR do not prune their trees; they simply set thresholds on how many observations are required for a parent node to split $N_{m_{threshold}}$ and the maximum number of nodes per tree (Equation 1). These two methods stop tree growth before growing to $T_{\max}$, thereby not requiring the tree pruning.

In addition, cross validation is unnecessary for overfit protection because we build trees from a bootstrapped training sample. As discussed in section 3.1.1, there is an extremely low probability that each bootstrapped sample will be equivalent to the original training dataset.

3.2 Robust Properties of RRFR

This section provides support to the claim that robust measures are better suited to model data with extreme outliers. Random Forest Regression lacks robustness due to its use of the mean for prediction of a terminal node when an unbounded outlier is present. In contrast, Robust Random Forest Regression, using the median for prediction in a terminal node, is less influenced by an unbounded outlier.

Suppose that the training dataset contains an extreme unbounded outlier, y_r, then RFR, where $\psi = \bar{y}$, has node estimates that are not robust i.e. they are not bounded in the presence of this outlier as shown in theorem 1.

Theorem 1: For an observation y_r in the bootstrap sample, there exists a node m in the random forest with predictor $\left\{h(\mathbf{x},\Theta_z)_{\bar{y}}, z=1,...,Z\right\}$ such that $h_m(\mathbf{x},\Theta_z)_{\bar{y}} \to \infty$ as $y_r \to \infty$.

Proof. At each tree in the random forest with predictor $\left\{h(\mathbf{x},\Theta_z)_{\bar{y}}, z=1,...,Z\right\}$ for which the y_r is in the bootstrapped sample there exists a terminal node, m that contains y_r. The existence of this node follows from construction since this data point must reach a terminal node. Let the predictor at this node be $h_m(\mathbf{x},\Theta_z)_{\bar{y}}$. Then $h_m(\mathbf{x},\Theta_z)_{\bar{y}} = \frac{\sum_{1}^{N_m}\mathbf{y}}{N_m}$. But $y_r \to \infty$ so $h_m(\mathbf{x},\Theta_z)_{\bar{y}} \to \infty$.

QED

An important issue is existence of this extreme unbounded outlier in the bootstrapped training dataset. The probability a single outlier will be included in the bootstrap sample of a single tree is

$$P(y_r \mid T_z) = 1 - \left[\left(\frac{N-1}{N}\right)\right]^N \geq 0.63$$

Equation 10: Probability of single observation is bootstrapped into single tree

where y_r is a single instance of an unbounded outlier, T_z is a single tree and $N \geq 1$ is the number of observations in the modeled dataset

As we increase the number of trees to build a forest, the probability that a single outlier is included in the bootstrap sample of a forest is

$$P(y_r \mid T_Z) = 1 - \left[\left(\frac{N-1}{N}\right)\right]^{N*Z} \approx 1$$

Equation 11 : Probability of single observation is bootstrapped into a forest

where y_r is a single instance of an unbounded outlier, T_Z is a forest of size $Z \geq 5$ trees and $N \geq 1$ is the number of observations in the modeled dataset.

Since we are building forests, we can see that y_r will impact the predictions based on the high probability of inclusion in the bootstrapping process.

In the presence of an unbounded outlier, y_r, RRFR, where $\psi = \tilde{y}$, is robust i.e. the predictor remains bounded.

Theorem 2: Assume y_r is in a single training set sample, then for all nodes, $N_m \geq 3$, in the forest $h_m(\mathbf{x},\Theta_z)_{\tilde{y}} < \infty$.

Proof. For any terminal node m, where $N_m \geq 3$, containing y_r, $h_m(\mathbf{x},\Theta_z)_{\tilde{y}} < \infty$ since $h_m(\mathbf{x},\Theta_z)_{\tilde{y}} = \underset{j\in J_m}{median}\{y_j\}$ where J_m = the set of indices of y_i in node m, $i = 1,...,I$. **QED**

Theorem 2 holds when only one instance of y_r is in the node. However, the theorem fails, $h_m(\mathbf{x},\Theta_z)_{\tilde{y}} \to \infty$, when more than one instance of y_r is bootstrapped and inhabits a node with $N_m \leq 4$. In the case when two instances of y_r are in the bootstrap and hence the node, the probability is $P(y_r : 2 \mid T_z) = 1/N^2$. To protect against this, we can set $N_m \geq 5$ and/or choose a very large training dataset. The probability of three y_r in the bootstrap further diminishes to $P(y_r : 3 \mid T_z) = 1/N^3$ which is very small for a large dataset.

3.3 RRFR at the Forest Level

Random Forest Regression grows trees based on a random vector Θ such that the predictor $h(\mathbf{x},\Theta)_\psi$ takes on numerical values. These numerical output values are assumed to be drawn from a training dataset that is independently drawn from the distribution of the random vector $[Y : \mathbf{X}]$. The mean absolute deviation generalization error for any numerical predictor for the random variable $\mathbf{X}$

$$E_{Y\mathbf{X}} \left| Y - h(\mathbf{X},\Theta_z) \right|$$

Equation 12: Mean absolute deviation generalization error

The prediction for a random forest is calculated by taking the average over z of the trees $h(\mathbf{x},\Theta)_\psi$. We now use robust statistics e.g. median and MAD which provides similar results to Breiman's Theorem 11.1 (Breiman, 2001). Therefore we have the following:

Theorem 3 As the number of trees or the number of nodes in the forest grows to infinity, almost surely,

$$E_{Y\mathbf{X}} \left| Y - avg_z h(\mathbf{X},\Theta_z)_\psi \right| \to E_{Y\mathbf{X}} \left| Y - E_\Theta h(\mathbf{X},\Theta)_\psi \right|$$

Equation 13: Almost sure convergence

The proof that the above theorem holds follows from the application of the Law of Large Numbers and the Central Limit Theorem as was done in (Breiman, 2001).

If k = number of terminal nodes (leafs) and z = number of trees built

1) As $z \to \infty$ or $k \to \infty$, every $h(\mathbf{x},\Theta_z)_\psi$ is included in the analysis and,
2) As $z \to \infty$ or $k \to \infty$, there is no bias, i.e. no $h(\mathbf{x},\Theta_z)_\psi$ is likely to be included a larger proportion of the time than the average observation or $P(h(\mathbf{x},\Theta_z)_\psi) \in SampleSpace) = 1/n$.

The idea of selecting predictor variables at random may at first seem counterintuitive. Randomly selecting predictor variables decreases correlation (Equation 15) between trees, while maintaining unpruned trees helps the model retain some predictor strength (Equation 14). This balance is what assists Random Forest models to perform very well.

The underlying theory of the balance of correlation and strength in RRFR is directly relevant to RFR. In presentation, we maintain similar notation, only altering to maintain consistency in this study. Validation and proofs of this theory are also found in (Breiman, 2001).

Definition: The margin function of a random forest is

$$mr(\mathbf{X},Y) = P_{\Theta}(h(\mathbf{X},\Theta)_{\psi}) = Y) - \max_{j \neq Y} P_{\Theta}(h(\mathbf{X},\Theta)_{\psi}) = j)$$

and the strength of a set of predictors $\{h(\mathbf{x},\Theta)_{\psi}\}$ is

$$SP = E_{\mathbf{XY}} mr(\mathbf{X},Y)$$

Equation 14: Margin Function and Strength of Predictors

If we define the right-hand-side of Equation 13 as the forest generalization error $PE^*(forest)$, the average generalization of a tree is $PE^*(tree) = E_{\Theta}E_{XY}\left|Y - h(\mathbf{X},\Theta)_{\psi}\right|$. Similarly to Theorem 11.2 (Breiman, 2001), we may assume that for all Θ, $EY = E_{XY}h(\mathbf{X},\Theta)_{\psi}$ then

$$PE^*(forest) \leq \bar{\rho} PE^*(tree)$$

Equation 15: Relationship between Forest and Tree Generalization Error

where the $\bar{\rho}$ is the weighted correlation between the residuals $Y - h(\mathbf{X},\Theta)_{\psi}$ and $Y - h(\mathbf{X},\Theta')_{\psi}$ with Θ and Θ' being independent. Therefore, the average error of the trees decreases by the factor of $\bar{\rho}$ as we build the random forest.

3.3.1 How do Forests Predict and Calculate Error?

The key question when dealing with forests rather than trees is how to combine the trees to arrive at a solution (Equation 13). After each tree, *T*, of the forest is built, the forest makes predictions on each individual raw training dataset observation relying on a technique called *bagging* (Breiman, 1994; Breiman, 1998). The predictions are based on the *out-of-bag* predictions from the tree only. The *out-of-bag* predictions are those predictions derived from non-bootstrapped observations which built that particular tree. On average, each bootstrapped sample is about $1/3$ *out-of-bag* observations (Breiman, 2003). For subsequent trees, the predictions are a rolling average for each particular *out-of-bag* observation value which is then compared to the raw training dataset to calculate error.

For a count of 1 to N_{Tng} (number of training observations)

    If that sample observation was NOT bootstrapped then
    Calculate forest estimate based on average of *out-of-bag* tree estimates
    Maintain count for number of trees used in the average
    End (Calculation If)

  Calculate the current error of the estimates

  End (Observation Loop)

Calculate overall error of estimates

Figure 3: *Bagging* Pseudo-Code

After the first tree, there are many observations in which the forest does not predict because for these observations a bootstrapped observation was used. Now, because there are only a few points used in prediction, the error starts off very high, drops quickly as the number of observations increases and later stabilizes. This also identifies why the *out-of-bag* (training) error is typically greater than or equal to the test error because training error is based only on the *out-of-bag* samples that are compared to the overall training dataset statistics whereas the test dataset is compared to the current forest.

In order to derive the test dataset estimate of error, pluralize the prediction over all the trees for the *out-of-bag* observations to calculate a test dataset estimate of $\hat{y}_n$ for y_n. Averaging the loss over all the test dataset observations provides the *out-of-bag* test dataset estimate of prediction error (Breiman, 2001).

3.3.2 Variable Importance

A frequent question in modeling is what predictor variables are more important to a model and to what degree are they important. Several modeling programs (Breiman, 1984; Breiman, 2001; Friedman, 1991; Friedman, 1999; Marketminer Corporation, 1996) incorporate such a measure in one manner or another to provide potentially valuable information to the model developer. The challenge to derive this relative ranking measure is to how to rank order the predictor variables that follow the first best predictor.

RFFR calculates variable importance at the forest level rather than for a single tree as in of the i^{th} prediction variable, $i = 1,...,I$. After a tree construction, the values of i^{th} variable in the out-of-bag examples are randomly permuted. The out-of-bag data is then dropped down that tree and the predictions for those out-of-bag x_n are stored. We repeat this for all predictor variables, $i = 1,...,I$. When all I variables are complete with their corresponding permutated runs, we compute a new estimate of error by comparing the true value of x_n versus the forest prediction based on the permutation of the predictor variables.

The *margin* (Equation 14) is then calculated by comparing the unpermuted error rate with the permuted error rate. If the *margin* is positive, meaning the permuted error rate exceeded the

unpermuted error rate, that value is the variable importance for the i^{th} prediction variable. Breiman describes this as "the average lowering of the margin across all cases when the i^{th} variable is randomly permuted (Breiman, 2003)." Neither RFR nor RRFR report negative values for variable importance.

4 EMPIRICAL TESTING

4.1 How does an extreme outlier affect RFR and RRFR?

We created a dataset with a very extreme outlier. The observation is an outlier due to an enormously large response term in comparison to the other responses. The outlier response is on the order of 10^{38} compared to values of 10^{3}. When the dataset is examined using Cook's D (Figure 4) and DFfits (Figure 5) the outlier is clearly identified.

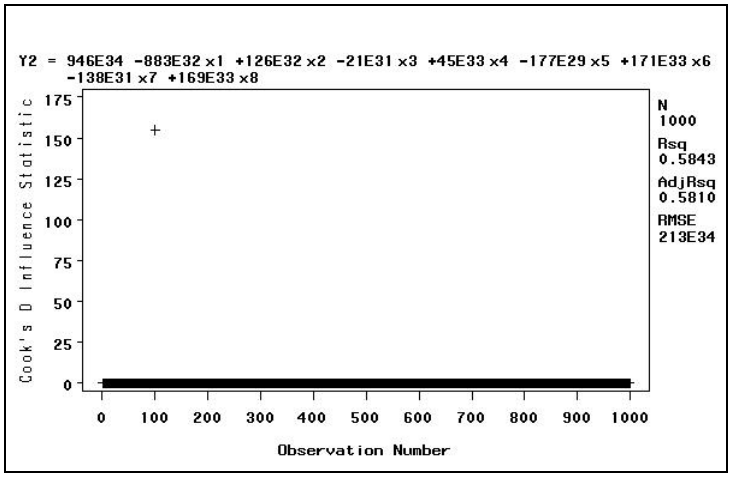

Figure 4: Cook's D for Extreme Outlier Case

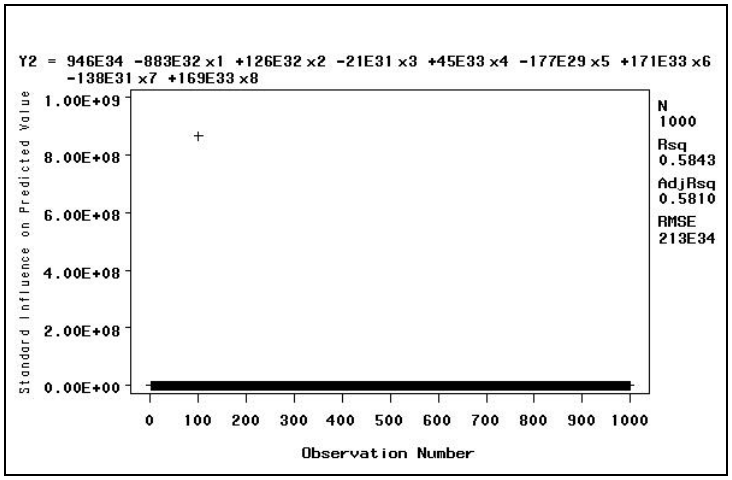

Figure 5: DFfits for Extreme Outlier Case

We ran an experiment using like parameters for RFR and RRFR (median). We ran the model in two steps. First, we ran ten iterations of the forest with the outlier present (1000 observations). Second, we ran ten iterations of the forest without the outlier (999 observations). After the predictions for all the runs were collected, we compared the predictions for like observations (deleting the outlier prediction because it had no mate for comparison). From these 999 differences, we found some interesting results.

Table 1 and Table 2 show the results from running the experiment. Similarly to DFfits we wish to consider the influence of the n^{th} case on the n^{th} fitted value or a scaled measure of change in the predicted value for the n^{th} observation. Therefore we calculate the sum of the change in the predictions, using the equation $\Delta h(\mathbf{x},\Theta)_{\psi} = \left| h(\mathbf{x},\Theta)_{\psi} - h(\mathbf{x}_{(r)},\Theta)_{\psi} \right|$ where $h(\mathbf{x},\Theta)_{\psi}$ is the dataset including the outlier and $h(\mathbf{x}_{(r)},\Theta)_{\psi}$ is the dataset with the outlier removed. This result is shown in the ΔPred columns.

The influence columns depict how many times the outlier influenced the prediction. Influence was easy to detect since the prediction values were much greater than the true observations or $>>10^3$.

RFR Extreme Outlier Experiment					
Run	ΔPred	Influence	Run	ΔPred	Influence
1	9.03E+37	879	6	1.17E+38	886
2	1.21E+38	879	7	8.86E+37	879
3	1.02E+38	884	8	9.93E+37	882
4	1.17E+38	885	9	1.08E+38	889
5	1.16E+38	892	10	1.04E+38	884
Average ΔPred		1.06E+38	Average Influence		883.90

Table 1: RFR Extreme Outlier Prediction Comparison

Table 1 shows the results from the RFR runs. The most interesting aspect of this table is the influence column. On average, the outlier influenced the predictions 85% (883.90 of 999) of the time. The sum of the average change in prediction for the ten runs was 1.06×10^{38}.

RRFR Extreme Outlier Experiment					
Run	ΔPred	Influence	Run	ΔPred	Influence
1	5.34E+37	30	6	7.99E+37	30
2	6.75E+37	28	7	6.51E+37	30
3	7.80E+37	35	8	7.40E+37	33
4	8.87E+37	39	9	9.34E+37	40
5	8.10E+37	27	10	9.43E+37	39
Average ΔPred		7.75E+37	Average Influence		33.10

Table 2: RRFR Extreme Outlier Prediction Comparison

Table 2 shows the results for the RRFR runs. The influence column here is not much of a factor. The outlier worked into the prediction an average of 3% (30.1 of 999) of the time. The sum of the average change in prediction for the ten runs was 7.75×10^{37}.

The results from this experiment empirically support the theoretical argument posed in section 3.2. The average change in prediction for the RRFR code is 2.88×10^{37} less than the RFR code.

An interesting outcome from this experiment was that the RRFR algorithm was influenced far less than the RFR code; however, the average change was still very large. From classical robust statistics, we would assume that the median and MAD used in RRFR would ignore the extreme outlier and entirely reject its influence. This did not happen in this experiment primarily because classical robust statistics are based on parametric models and RRFR is a nonparametric algorithm that utilizes bootstrap-aggregation or *bagging*.

Whenever the outlier is included in the bootstrapped training sample, there is a chance that the outlier will influence a prediction. The probability of outlier prediction influence increases as multiple copies of the outlier are included in the bootstrap. For RFR, if the outlier is present in a terminal node, the prediction will always be influenced based on the calculation of the mean. An outlier will influence the prediction of RRFR (median), when the terminal node's conditions are either 1) the outlier is the only observation in the terminal node, 2) there are only two observations in the terminal node and the outlier is one of them, or 3) there are multiple copies of the outlier which places it at the center of the distribution in the terminal node.

Influence from an outlier is far more extreme in RRFR than RFR. RFR uses the mean for predictions, so the outlier will always effect the prediction if present in a terminal node; however, its influence is somewhat muffled from the calculation. In contrast RRFR uses the median, so the exact value of the central-most observation(s) is used to determine the prediction. If the terminal node meets one of the three conditions mentioned above, the prediction is (essentially) the response value of the outlier. Both RFR and RRFR further subdue the prediction influence of the outlier through the aggregation step of *bagging*, but not generally enough to quell all influence.

From this experiment, even though RFR had more instances of influence, the effect per prediction was only on the order of 10^{34} or 10^{35} while the outliers influence on RRFR's predictions was on the order of 10^{36}. This is the main reason why even though RRFR has less instances of influence, the average change in prediction is still substantially large.

4.2 How does heteroscedasticity affect RFR and RRFR?

We created a dataset with a mixed Gaussian distribution response. We used a mean value of 10,000 for both distributions; however, distribution 1 (blue in Figure 6) had 100 observations and a standard deviation of 1,000 while distribution 2 (pink in Figure 6) had 900 observations and a standard deviation of 10. This dataset was designed to be heteroscedastic (displaying non-constant variance) in nature.

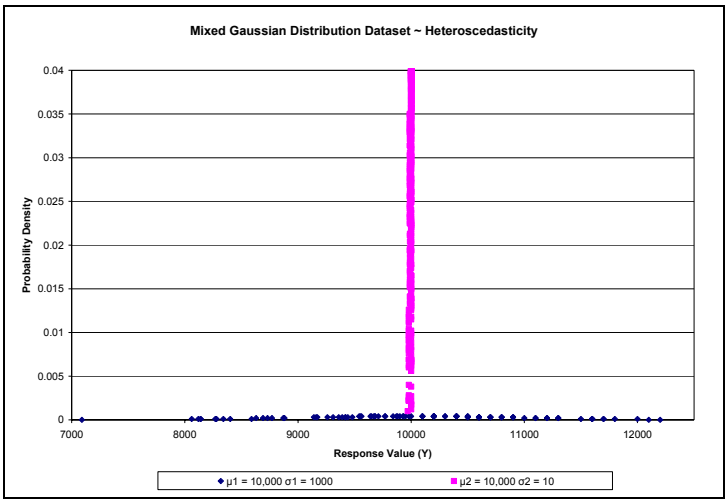

Figure 6: Mixed Gaussian Distribution Dataset ~ Heteroscedasticity Experiment

We ran an experiment using like parameters for RFR and RRFR (median). We ran the model in two steps. First, we ran ten iterations of the forest with both response distributions present (1000 observations). Second, we ran ten iterations of the forest without distribution 1 (900 observations). After the predictions for all the runs were collected, we compared the predictions for like observations (deleting the distribution 1 predictions because they had no mates for comparison). From these 900 differences, we were able to provide further support for Theorem 2.

Table 3 and Table 4 show the results from running the experiment. Similar to the calculation used in section 4.1, we calculate the sum of the change in the predictions using the equation $\Delta h(\mathbf{x},\Theta)'_{\psi} = \left| h(\mathbf{x},\Theta)_{\psi} - h((\mathbf{x}_{(\mathbf{x}')}),\Theta)_{\psi} \right|$, where $h(\mathbf{x},\Theta)_{\psi}$ includes all elements of the dataset and $h((\mathbf{x}_{(\mathbf{x}')}),\Theta)_{\psi}$ is the dataset with those high variance data (blue in Figure 6) removed. This result is shown in the ΔPred columns.

RFR Heteroscedasticity Experiment			
Run	ΔPred	Run	ΔPred
1	4.95E+04	6	5.00E+04
2	4.97E+04	7	5.06E+04
3	4.97E+04	8	5.10E+04
4	4.81E+04	9	4.68E+04
5	4.96E+04	10	4.90E+04
Average ΔPred		4.94E+04	

Table 3: RFR Heteroscedasticity Prediction Comparison

RRFR Heteroscedasticity Experiment			
Run	ΔPred	Run	ΔPred
1	3.01E+04	6	2.99E+04
2	2.94E+04	7	3.09E+04
3	3.16E+04	8	2.85E+04
4	2.87E+04	9	2.85E+04
5	2.98E+04	10	2.89E+04
Average ΔPred		2.96E+04	

Table 4: RRFR Heteroscedasticity Prediction Comparison

Table 3 and Table 4 display the results for this experiment for RFR and RRFR, respectively. The sum of the average change in prediction for the ten runs for RFR was 4.94×10^4, while RRFR's sum of the change was 2.96×10^4. Thus, the average change in prediction for the RRFR code is 1.98×10^4 less than the RFR code.

As with the result from section 4.1, the reason that the difference between the RRFR and RFR is not more pronounced is because of *bagging*. The same three influence conditions for RRFR apply here except that the probability of abnormal data influence is greater because instead of a single observation (unbounded outlier) affecting the prediction, the data has many observations (with high variance) that could adversely influence the prediction.

5 CONCLUSION

The introduction of robust prediction and error statistics using the median and other robust measures for prediction, and the use of mean absolute deviation to calculate both the in-node and overall error of RRFR makes this algorithm very attractive when unbounded outliers and heteroscedastic conditions arise in datasets. By adapting this new strategy, we theoretically lessen the effects of outliers and heteroscedasticity while retaining the beneficial elements of the Random Forest Regression algorithm.

Future work includes an in-depth examination of the performance of RRFR, in comparison to RFR, using many of the datasets from (Breiman, 2001) and a few others. In addition, we will explore the general performance of these algorithms based on different model parameter inputs. We will further explain any strengths and vulnerabilities of these algorithms and provide insight into possible methods of improvement.

This exploratory study has shown that even with the excellent prediction performance of the original Random Forest Regression that this algorithm still has some shortfalls to overcome with respect to relatively dirty datasets that include extreme outliers, strange observations or heteroscedastic properties. In this vein, we are confident that the initial results from the performance of RRFR provide an excellent foundation for an improved predictive algorithm.

6 REFERENCES

- Breiman, L., Friedman, J., Olshen, R. A. & Stone C. J. 1984. Classification and Regression Trees. Wadsworth and Brooks/Cole, Monterey, CA.
- Breiman, L. 1994. Bagging Predictors., Statistics Department, University of California Technical Report No. 471.
- Breiman, L. 1998. Out-of-bag Estimation. Statistics Department, University of California, ftp://ftp.stat.berkeley.edu/pub/users/breiman/.
- Breiman, L. 2000. Understanding Complex Predictors, Statistics Department, University of California. ftp://ftp.stat.berkeley.edu/pub/users/breiman/.
- Breiman, L. 2001. Random Forests, Statistics Department, University of California, ftp://ftp.stat.berkeley.edu/pub/users/breiman/.
- Breiman, L. 2003. RF/tools: A Set of Two-eyed Algorithms. An invited presentation given at SIAM Workshop. ftp://ftp.stat.berkeley.edu/pub/users/breiman/.

- Chaudhuri, P., Huang, M, Loh, W., & Yao, R. 2001. Piecewise-Polynomial Regression Trees, Indian Statistical Institute, National Cheng Kung University, and University of Wisconsin.
- Friedman, J. 1991. Multivariate Adaptive Regression Splines, The Annals of Statistics, 19, 1, pp. 1-67.
- Friedman, J. 1999. Stochastic Gradient Boosting, Department of Statistics & Stanford Linear Accelerator Center, Stanford University.
- Hampel, F., Ronchetti, E, Rousseeuw, P., & Stahel, W. 1986. Robust Statistics: The Approach Based on Influence Functions, John Wiley and Sons, New York.
- Härdle, W. 1990. Applied Nonparametric Regression, MIT Press.
- Hastie, T., Tibishirani, R., & Friedman, J. 2002. The Elements of Statistical Learning: Data Mining, Inference, and Prediction. Springer, New York.
- Hoaglin, D. C., Mosteller, F., and Tukey, J. W. (eds.). 1983. Understanding Robust and Exploratory Data Analysis, New York: John Wiley & Sons.
- Loh, Wen-Yin, [2002] Regression Trees with Unbiased Variable Selection and Interaction Detection, Statistica Sinica, vol 12, pp. 361-386.
- Marketminer Corporation. 1996. ModelQuest™ User's Manual, Charlottesville, Virginia.
- Torgo, L. 1999. Inductive Learning of Tree-Based Regression Models. PhD Dissertation, Departmento de Ciéncia de Computadores Faculdade de Ciéncias de Universidade do Porto.