DD2434 Project Group 25
Replicating Results from the *Random Forests* paper by L.Breiman

Oktay Bahceci Hui Pang Sugandh Sinha Diego Yus

January 16, 2017

Abstract

Random forest is a classical ensemble method for classification and regression. It is a strong classifier consisting of a set of relatively weak decision trees. Each tree is sampled from i.i.d. random variables. The output can be majority vote for classification. We re-implemented this method and reproduced part of the result from the paper. We use the random selection of features to determine how many features to select. We also compare our results on strength, correlation, and their relation to error rates. It can be proved that even with some noise and bias, our implementation is successful.

1 Introduction

Random forest belongs to a family of learning algorithms that are called *Ensemble Methods*. Ensemble methods work by essentially constructing a set of (weak) classifiers and then taking a majority vote or mode of the output of those classifiers for classification task, or taking the average for regression. A lot of interest has been shown in Ensemble methods because of their ability to generate classifiers, from a set of classifiers, that perform better than any single classifier belonging to that set.

Two of the main techniques in Ensemble methods, which are also called *meta-algorithms*, are *Boosting* and *Bagging*. In Boosting, we are given a classifier, we run it multiple times on the data and in every iteration of the run, we weight each training sample based on the amount of its misclassification. Boosting helps in decreasing the bias. In the paper, the author is also comparing his results with Adaboost algorithm which is a boosting algorithm. In Bagging, short for Bootstrap Aggregation, is a way to manipulate training set. In bagging, we generate from our original dataset multiple subsets by sampling with replacement to train multiple classifiers and then combine their results. Bagging helps in reducing variance.

Random forest which was originally proposed by author of paper, in simple terms, can be said to have an extra layer of randomness on top of bagging. The "random" in Random forest comes because of two reasons: 1. Randomness involved in generating bootstrap replicas of the data. 2. At each node, the split is done by selecting the best, from a set of randomly selected predictors.

Although this strategy may sound quite unreasonable, it outperforms many other traditional classifiers such as *Support Vector Machine* and also works well against overfitting. The Random Forest algorithm can be used for both Classification and Regression.

2 Background

2.1 Decision Tree Learning

A *Decision Tree* is a data mining technique designed to create a model that is capable of predicting the outcome value of a target variable based on the value of all other input variables. Decision trees are most commonly used in decision making and is a *predictive model* in the field of machine learning. Decision tree learning is applied and used in areas such as statistics, data mining and machine learning. It is a tree-like model of decisions with corresponding possible consequences and chances of outcomes of events or costs. The tree is built using nodes, edges and leafs. Each node in the decision tree corresponds to an input variable where the edges between the nodes represent possible values of the input variables. The leaf nodes corresponds to the deduced target variable value based on the path from the root of the tree to a specific leaf.

There are multiple learning algorithms to *learn* a tree. It is commonly trained by splitting the source set into subsets based on an attribute value test. This process is repeated on each derived subset recursively. One of the most commonly used learning algorithms is namely the top-down induction recursive partitioning algorithm, a greedy algorithm. This algorithm makes optimal choices at each recursive step building trees.

Another common algorithm used to generate a decision tree from a dataset is the Iterative Dchotomiser 3 (ID3) algorithm. This algorithm is based on practices in information theory, namely *information gain* and *entropy*, the unpredictability or impurity of the content.

Yet another algorithm available for Decision trees is CART, which is also used by the author to form decision trees in the paper. In this algorithm, a binary tree is formed by recursively using binary partition. The advantage of CART over logistic regression or SVM is that it can accommodate different types of input data like age or gender.

2.2 Tree Bagging

In case of decision trees, the *bagging* algorithm work as follows:

- A random sample of size **N** with replacement from data is taken (bootstrap sample).
- Construct a decision tree normally without pruning the tree.
- A class is assigned to each leaf node and the class attached to each case coupled with the predictor values for each observation is also kept.
- Repeat the above steps for a considerable amount of iterations, until k steps.
- Finally, each observation is assigned to a final category by a majority vote over the set of trees.

Usage of Bagging also coins a new term called *Out-of-Bag Observations* which are simply the observations that were not used in the bootstrap sample. Prediction error is based only on the Out-of-bag observations. There is empirical evidence ([1]) that using out-of-bag estimates is as accurate as using a test set of the same size as the training set, so there is no need for a set aside test. The out-of-bag estimates will then combine less classifiers than the actual combination of them (the whole forest). Since the classification accuracy increases when more trees are added, the out-of-bag estimates will always overestimate the error rates.

2.3 Random Forest

Like mentioned earlier in the Introduction section, Random Forest works in a very similar fashion as Bagging. Almost the same steps as for tree bagging are followed, but the trees are grown in a different way. For each node and for each individual tree, a random sample of the predictors is taken without replacement and the node is split according to the best split using these predictors.

Three parameters that affect the performance of Random Forest are: 1. Number of trees; 2. Number of predictors that are sampled; 3. Node Size.

With Random Forests, we can achieve both a decrease in variance and bias. In Random Forests, because of the combination of bootstrap samples and random pick of predictors during split, we get a decrease in variance. As for the decrease in bias because of the possibility of using large number of predictors and also, since random forest can use large number of trees, all the predictors, even the ones that are not dominant, have the possibility of participating in some split.

In the paper, [4], the generalization error of the forests is shown to converge because of the Law of Large Numbers, so that overfitting does not occur as for single decision trees. To explain the derivations, we will focus on the two-classes problem, since it is easier to provide intuition for functions.

The generalization error of a forest can be expressed as

$$PE* = P_{\mathbf{X},Y}(mr(\mathbf{X},Y) < 0) \tag{1}$$

where for the two classes problem

$$mr(\mathbf{X},Y)) = 2P_{\Theta}(h_k(\mathbf{X},\Theta) = Y) - 1 \tag{2}$$

Here, $h_k(\mathbf{X}, \Theta)$ is an individual classifier (tree) of the forest and $P_\Theta()$ expresses the probability over all different trees. The margin function illustrates how good the classification of a sample is in terms of the numbers of trees that correctly classified it.
In the paper, it is shown that the generalization error will have an upper bound given by:

$$PE* \leq \bar{\rho}(1 - s^2)/s^2 \tag{3}$$

where s (strength) and $\bar{\rho}$ (mean correlation) are two key parameters to understand the behavior of the forest. The strength is defined as:

$$s = E_{\mathbf{X},Y}[mr(\mathbf{X}, Y)] \tag{4}$$

and can be understood as the difference between correct and incorrect classified samples averaged for all samples and all trees. It expresses how good (*strong*) an individual classifier (tree) of the forest is in classifying. The correlation ρ is defined as:

$$\rho = cor(rmg(\Theta, \mathbf{X}, Y), rmg(\Theta', \mathbf{X}, Y)) \tag{5}$$

where $rmg(\Theta, \mathbf{X}, Y)$ is the raw margin function given by:

$$rmg(\Theta, \mathbf{X}, Y) = 2I(h(\mathbf{X}, \Theta) = Y) - 1 \tag{6}$$

Basically, the raw margin function yields a 1 when the sample is correctly classified and -1 when it is incorrectly classified. The correlation just expresses how similar the classification behavior of two individual classifiers (trees) is, regardless of their accuracy. The mean correlation is simply the mean value of the correlation for all the pairs of trees. It will express how similar (*correlate*) the trees in the forest (on average) are to each other in their classifying behavior.
These two magnitudes will be very important when assessing the accuracy of a classifier. An increase in strength will improve accuracy of the trees (decrease bias) and decrease the generalization error, while an increase in correlation will increase the generalization error, because if the trees are more similar, the reduction in variance due to averaging will be lower. Their dependence on F will be discussed in *Section 4: Results*.

3 Re-Implementation of Method

For the implementation of the Random Forests, we follow the guidelines mentioned by the author in the paper. Bagging (sampling with replacement) is used in tandem with random feature selection for each node in the tree. For the growing of the trees, the CART (Classification and Regression Trees) methodology is used, which can be summarized in the following points:

1. At each node, the parent node is split into two children and subsequently in a recursive procedure. 2. The best split is selected among the possible splits based on the attribute's values. 3. There is no pruning, the trees are grown until the maximum size when no further gain can be achieved.

All specific parameter values such as the number of trees used, and the number of runs for averaging are used as described in the paper.

3.1 Dataset

Regarding datasets, some of the datasets proposed by the author are not complete, i.e., some values are missing and marked with *?*, denoting an unknown value. To solve this issue, since there is no solution proposed in the paper, we follow the simple way mentioned by the author, Official Website for Random Forests: *If the mth variable is not categorical, the method computes the median of all values of this variable in class j, then it uses this value to replace all missing values of the mth variable in class j. If the mth variable is categorical, the replacement is the most frequent non-missing value in class j.*

The only exception was made in the *House votes* dataset, where the large number of missing values made us decide for setting a new categorical value of "missing" and by mapping the non-numerical features into numerical to support our implementation.

4 Reproduction of Results

Among the many sections of the paper, we decided to focus on reproduction of sections 4 and 6, because we thought the core understanding of Random Forests (for categorical output) can be found there.

4.1 Section 4: Random forests using random input selection

In Section 4 the simple random forest with random features is constructed, by selecting a random subset of the variables (attributes) at each node and choosing the best variable to split among them.

The size of this subset is denoted F and two values are tested: $F = 1$ (so the attribute selected at random is the one used for splitting) and $F = floor(log_2(M) + 1))$ where M is the total number of attributes. The results of the paper are shown in Table 1b and our results are shown in Table 1a.

Dataset	Selection	Single Input	One tree
Glass	31.5	34.6	40.0
Breast Cancer	3.4	2.7	7.1
Diabetes	27.7	32.1	33.3
Sonar	22.0	22.3	30.4
Vehicle	43.9	44.4	45.0
Votes	4.1	5.4	7.1

(a) Our results

Data set	Adaboost	Selection	Forest-RI single input	One tree
Glass	22.0	20.6	21.2	36.9
Breast cancer	3.2	2.9	2.7	6.3
Diabetes	26.6	24.2	24.3	33.1
Sonar	15.6	15.9	18.0	31.7
Vowel	4.1	3.4	3.3	30.4
Ionosphere	6.4	7.1	7.5	12.7
Vehicle	23.2	25.8	26.4	33.1
German credit	23.5	24.4	26.2	33.3
Image	1.6	2.1	2.7	6.4
Ecoli	14.8	12.8	13.0	24.5
Votes	4.8	4.1	4.6	7.4

(b) Paper results

Table 1: Out-of-bag, test set and individual tree out-of-bag errors for different datasets

Note: The Adaboost values were not reproduced but directly referred from the paper

We didn't reproduce all the datasets in the paper because the different file formats made the process rather tedious and much of the dataset would have to have been reformatted. Instead, we chose some representative datasets to run Random Forests on: Votes and Breast Cancer are categorical datasets, while the others have numerical values. We also selected different datasets for trying binary classification (Breast Cancer, Diabetes, Sonar and Votes) and others for multiple-class classification (Vehicle, Glass).

We can see that the results for the categorical datasets are very similar to the values stated in the paper, while the numerical datasets have a difference between 5 and 10% error compared to the values mentioned in the paper, for the three errors that were computed. We believe that this difference comes from the splitting rules for each node. These rules are not explicitly mentioned in the paper, but we suppose according to some official documentations of popular machine learning packages[1] that different thresholds were tested for each of the attributes belonging to the subset in order to determine not only the best attribute, but the best threshold in this attribute.

Nevertheless, as not mentioned in the paper, we chose the threshold as the middle value of the domain for that attribute. If we had had the information of what the author meant by *best split*, we could have certainly increased the gain in some of the splits done in the nodes and reduced the error.

However, this is not the case for categorical variables. When splitting at a categorical variable in a node, all the possible subsets of the categories of the variable (generally few) are checked and the threshold is set at the best. Also, when comparing accuracy to Adaboost (main goal of the paper), we can see that for categorical values, we have a similar or even performance while the numerical values, as mentioned, perform badly.

In both tables 1a and 1b, we can appreciate that test set error and out-of-bag estimate of the generalization error for the whole forest are very similar, while out-of-bag estimate of generalization error for of individual trees is larger, as we could have expected.

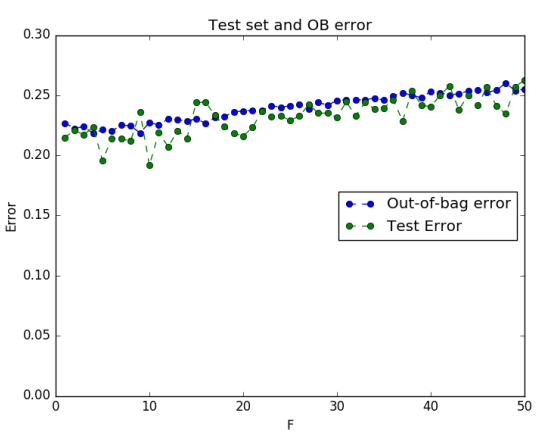
4.2 Section 6: Empirical results on strength and correlation

We worked on the sonar data set with the Forest-RI algorithm that we re-implemented as mentioned in previous section, in order to reproduce the empirical results on strength, correlation, and their relation to the generalization error. Further, we intended to reproduce the lack of sensitivity in the generalization error to the group size F.

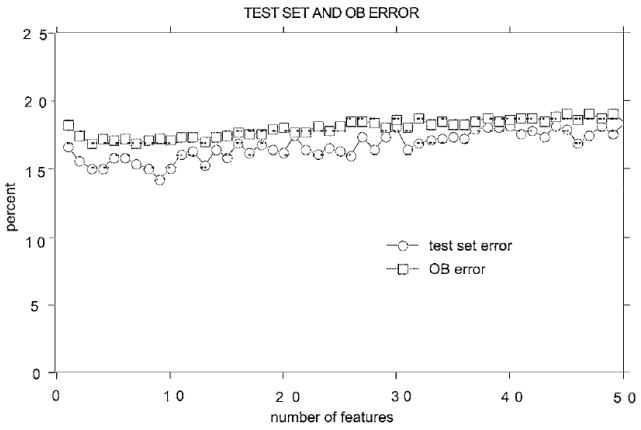
Just as described in Section 6, we ran Forest-RI on the sonar data varying F from 1 to 50 inputs, grew 100 trees, and averaged terminal results for 80 iterations.
According to our understanding of the data sets during re-implementation, the sonar data set is chosen because of its many attributes (60 inputs, second-most in the original paper [4] *Table 1*) and few number of classes (binary).

Therefore, the data set suits our intention to investigate the sensitivity (or lack of it) of our random forest to the change of F, without the need for generating more features with linear combination of inputs. We plot our reproduced results next to the screenshot of results from the paper (Figs. 1 and 2), similarly w.r.t. the increase of F. In doing so we intend to both prove the validity of our re-implementation, and try to explain the difference by linking the plot of strength and correlation to the plot of error rates.

[1] `https://www.stat.berkeley.edu/~breiman/RandomForests/cc_home.htm`

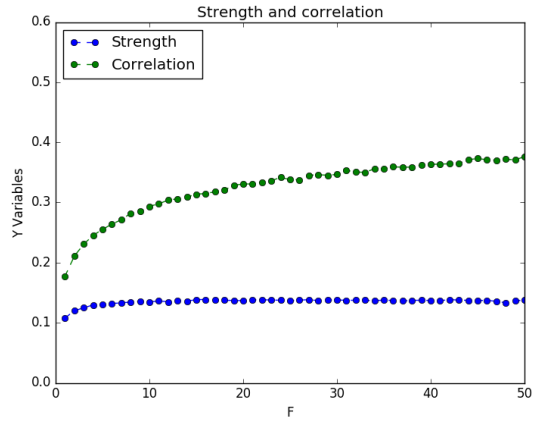


(a) Our reproduction

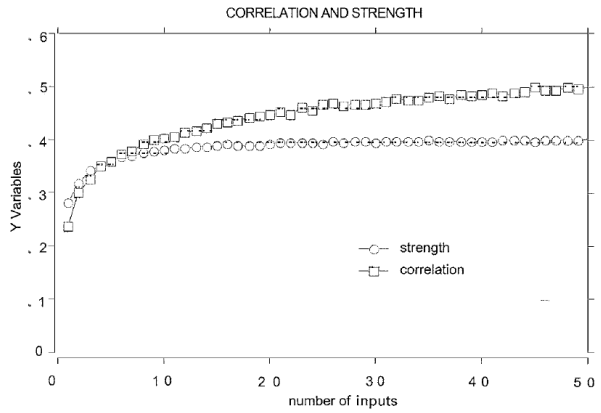


(b) Original results from paper

Figure 2: Comparison of our re-implementation and original results of error rates(test set error and OB error) w.r.t. group size F on the sonar data set. We can see that while we succeed in reproducing similar tendencies for both, our re-implementation got higher error rates for both.



(a) Our reproduction



(b) Original results from paper

Figure 1: Comparison of our re-implementation and original results of out-of-bag estimates for strength and correlation w.r.t. group size F on the sonar data set. We can see that while we succeed in reproducing very similar strength, the correlation has a constant positive bias from the original result.

As we can see in Figure 2, both Figure 2a and 2b have exactly the same shape, having only a difference in the absolute value for the error, ours classifying worse. We believe the source of this error is the split method mentioned before in Section 4.1.
Both curves reproduce the same behavior; as papers says: "both errors (out-of-bag and test set) show the same behavior -a small drop from F = 1 out to F about 4-8, and then a general, gradual increase". Similarly, the out-of-bag error is more stable than the test set error and we can see that, for both our reproduction and the original results, the OB error is similar to test set error for cross validation, which guarantees convergence, as is mentioned in the same author's another paper[1]. We can appreciate as well that out-of-bag errors overestimate the actual error, as discussed in Section 2.

In Figure 1, again both curves, 1a and 1b have the same shape, with the strength becoming constant after a point around $F = 8$ and the correlation continuously increasing over all the F range. Again, there is a difference in the absolute values. Strength of our RF is lower, probably because the classification is worse, and therefore the margin function yields lower values. Correlation values are similar because the (mean) correlation is not influenced by how good the classifiers are, but for the similarity among them, for which the source of error mentioned before does not have a high influence. As expected, the increase in the error tallies with the constant region of the strength.

Nevertheless, the correct behavior of the curves show that the algorithm works correctly, and value difference is due to a methodological error, but not a conceptual one.

Finally, we would like to add a short explanation about the dependence of strength and correlation on F. Regarding strength, the increase in F would allow larger subsets of attributes at each node and therefore better possible splits, so the accuracy would increase. After some point though, adding more inputs does not help and the strength remains constant. For correlation, with low F values, the subsets are small, so there are few possible attributes to split at the nodes and due to randomness, the trees are quite dissimilar. As F increases, the number of possible attributes to split at each node grows, and trees tend to choose same attributes more often, so the correlation increases as well.

5 Discussion

So what is good about Random Forest? From our experience with it, for data sets, it works for both categorical and continuous responses. Also, it yields favorable results compared with other state-of-the-art methods, such as Adaboost mentioned by the author. Besides, with out-of-bag estimates that perform as good as test error, as is shown in 2, there is no need for a separate validation data set, therefore cut off some computational burden.

There is still some aspects left for improvement though. For well-known efforts on it, there are ExtraTrees[2], that introduces more randomness by selecting random points for splitting, and can be computationally faster, or manipulate the description space by building subsets with Principle Component Analysis, as Rotation Forest[3] does. However, most of them take the number of trees and group sizes as *a priori*. We may introduce more dynamic ways to optimize the selection of those parameters, by monitoring the strength and correlation and therefore the upper bound of generalization error.

References

[1] Leo Breiman. Out-of-bag estimation. Technical report, Citeseer, 1996.

[2] Pierre Geurts, Damien Ernst, and Louis Wehenkel. Extremely randomized trees. *Machine learning*, 63(1):3–42, 2006.

[3] Juan José Rodriguez, Ludmila I Kuncheva, and Carlos J Alonso. Rotation forest: A new classifier ensemble method. *IEEE transactions on pattern analysis and machine intelligence*, 28(10):1619–1630, 2006.

[4] Leo Breiman Statistics and Leo Breiman. Random forests. In *Machine Learning*, pages 5–32, 2001.