



Random Forests

LEO BREIMAN

Statistics Department, University of California, Berkeley, CA 94720

Editor: Robert E. Schapire

Abstract. Random forests are a combination of tree predictors such that each tree depends on the data of a random set of sampled independent and identically distributed observations for all trees in the forest. The generalization error for forests converges a.s. to a limit as the number of trees in the forest becomes large. The generalization error of a forest of tree classifiers depends on the strength of the individual trees in the forest and the correlation between them. Using a random selection of features to split each node yields error rates that compare favorably to Adaboost (Y. Freund & R. Schapire, *Machine Learning: Proceedings of the Thirteenth International conference*, 1999, 148–156), but are more robust to overfitting. Internal estimates of error, strength, and correlation and their use in pruning are described. The ideas are also applicable to regression.

Keywords: classification, regression, ensemble

1. Random forests

1.1. Introduction

Significant improvements in classification accuracy have resulted from growing an ensemble of trees and letting them vote for the most popular class. In order to grow the ensemble, often random subsets are generated that govern the growth of each tree in the ensemble. An early example is bagging (Breiman, 1996), where to grow each tree a random selection (with replacement) is made from the example in the training set.

Another example is random split selection (Dietterich, 1998) where at each node the split is selected at random from among the K best splits. Breiman (1999) generated n training sets by randomly selecting the observations in the original training set. Another approach is to select the training set from a random set of weights on the example in the training set. Ho (1998) has written a number of papers on the random subspace method which does a random selection of a subset of features to use to grow each tree.

In an important paper on written character recognition, Amit and Geman (1997) describe a large number of geometric features and search over a random selection of the features for the best split at each node. This latter paper has been influential in my thinking.

The common element in all of these procedures is that for the k th tree, a random subset Θ_k is generated, independent of the past random subsets $\Theta_1, \dots, \Theta_{k-1}$ by which the observations are distributed; and a tree is grown using the training set and Θ_k , resulting in a classifier $h(\mathbf{x}, \Theta_k)$ where $\mathbf{x}$ is an input vector. For instance, in bagging the random subset Θ is

generated a tree Θ in N bootstraping from N data through a random sample Θ of the bootstrap, where N is number of sample in the training set. In random split election Θ consist of a number of independent random integer between 1 and K . The nature and dimensionality of Θ depend on the tree construction.

After a large number of trees generated, the score for the most popular class. We call the procedure **random forests**.

Definition 1.1. A random forest is a classifier consisting of a collection of tree-structured classifiers $\{h(\mathbf{x}, \Theta_k), k = 1, \dots\}$ where the $\{\Theta_k\}$ are independent identically distributed random vectors and each tree calls a node score for the most popular class in $\mathbf{x}$.

1.2. Outline of paper

Section 2 give some theoretical background for random forest. Use of the Strong Law of Large Number help to show the algorithm converge to the optimal solution in most a problem. We give a simplified and extended version of the Amis and Geman (1997) analysis to show that the accuracy of a random forest depend on the strength of the individual tree classifier and a measure of the dependence between them (see Section 2 for definition).

Section 3 introduce foresting the random election of features at each node to determine the split. An important question is how many features to select at each node. For guidance, internal estimate of the generalization error, classifier strength and dependence are computed. These are called out-of-bag estimate and are related in Section 4. Section 5 and 6 give empirical results for two different form of random features. These results random election from the original input; the second is random linear combination of input. The results compare favorably to Adaboost.

The results may also be interpreted in terms of the number of features selected to split each node. Usually, selecting one or two features give near optimal results. To explore this and relationship to strength and correlation, an empirical study is carried out in Section 7.

Adaboost has no random element and grows an ensemble of weak classifiers by reweighing of the training set where the classifier weights depend on the parity of the ensemble formation. By just a deterministic random number generator can give a good imitation of randomness, my belief is that in the latter stage Adaboost is emulating a random forest. Evidence for this conjecture is given in Section 8.

Important recent problem, i.e., medical diagnosis and document retrieval, often have the proper way where there are many input variable, often in the hundreds or thousands and each one containing only a small amount of information. A single tree classifier will then have accuracy only slightly better than a random choice of class. By combining tree grown using random features can produce improved accuracy. In Section 9 we experiment on a simulated data set with 1,000 input variable, 1,000 samples in the training set and a 4,000 samples test set. Accuracy comparable to the Bagging is achieved.

In many applications, understanding of the mechanism of the random forest black box is needed. Section 10 make a variation of the bootstraping internal estimate of variable importance and binding the together bootstrap results.

Section 11 look at random forests for regression. A bound for the mean squared generalization error is derived, and how it has the decrease in error from the individual trees in the forest depend on the correlation between residual and the mean squared error of the individual trees. Empirical results for regression are in Section 12. Concluding remarks are given in Section 13.

2. Characterizing the accuracy of random forests

2.1. Random forests converge

Given an ensemble of classifiers $h_1(\mathbf{x}), h_2(\mathbf{x}), \dots, h_K(\mathbf{x})$, and let h be the voting or draught random from the distribution of the random vector $Y, \mathbf{X}$, denote the margin function as

$$mg(\mathbf{X}, Y) = \text{avg}_k I(h_k(\mathbf{X}) = Y) - \max_{j \neq Y} \text{avg}_k I(h_k(\mathbf{X}) = j).$$

where $I(\cdot)$ is the indicator function. The margin measures the extent to which the average number of votes at $\mathbf{X}, Y$ for the right class exceed the average vote for any other class. The larger the margin, the more confidence in the classification. The generalization error is given by

$$PE^* = P_{\mathbf{X}, Y}(mg(\mathbf{X}, Y) < 0)$$

where the subscript $\mathbf{X}, Y$ indicates the probability over the $\mathbf{X}, Y$ space.

In random forests, $h_k(\mathbf{X}) = h(\mathbf{X}, \Theta_k)$. For a large number of trees, it follows from the Strong Law of Large Numbers and the tree is correlated:

Theorem 1.2. *As the number of trees increases, for almost surely all sequences $\Theta_1, \dots, PE^*$ converges to*

$$P_{\mathbf{X}, Y}(P_\Theta(h(\mathbf{X}, \Theta) = Y) - \max_{j \neq Y} P_\Theta(h(\mathbf{X}, \Theta) = j) < 0). \quad (1)$$

Proof: see Appendix I. □

This result explains why random forests do not overfit as more trees are added, but produce a limiting value of the generalization error.

2.2. Strength and correlation

For random forests, an upper bound can be derived for the generalization error in terms of two parameters that are measures of how accurate the individual classifiers are and of the dependence between them. The interplay between the two gives the foundation for understanding the working of random forests. We build on the analysis in Amis and Geman (1997).

Definition 2.1. The margin function for a random forest is

$$mr(\mathbf{X}, Y) = P_{\Theta}(h(\mathbf{X}, \Theta) = Y) - \max_{j \neq Y} P_{\Theta}(h(\mathbf{X}, \Theta) = j) \quad (2)$$

and the strength of the set of classifiers $\{h(\mathbf{x}, \Theta)\}$ is

$$s = E_{\mathbf{X}, Y} mr(\mathbf{X}, Y). \quad (3)$$

Assuming $s \geq 0$, Chebyshev's inequality gives

$$PE^* \leq \text{var}(mr)/s^2 \quad (4)$$

A more revealing expression for the variance of mr is derived in the following: Let

$$\hat{j}(\mathbf{X}, Y) = \arg \max_{j \neq Y} P_{\Theta}(h(\mathbf{X}, \Theta) = j)$$

so

$$\begin{aligned} mr(\mathbf{X}, Y) &= P_{\Theta}(h(\mathbf{X}, \Theta) = Y) - P_{\Theta}(h(\mathbf{X}, \Theta) = \hat{j}(\mathbf{X}, Y)) \\ &= E_{\Theta}[I(h(\mathbf{X}, \Theta) = Y) - I(h(\mathbf{X}, \Theta) = \hat{j}(\mathbf{X}, Y))]. \end{aligned}$$

Definition 2.2. The raw margin function is

$$rmg(\Theta, \mathbf{X}, Y) = I(h(\mathbf{X}, \Theta) = Y) - I(h(\mathbf{X}, \Theta) = \hat{j}(\mathbf{X}, Y)).$$

Then, $mr(\mathbf{X}, Y)$ is the expectation of $rmg(\Theta, \mathbf{X}, Y)$ with respect to Θ . For an function f the identity

$$[E_{\Theta} f(\Theta)]^2 = E_{\Theta, \Theta'} f(\Theta) f(\Theta')$$

holds where Θ, Θ' are independent with the same distribution, implying that

$$mr(\mathbf{X}, Y)^2 = E_{\Theta, \Theta'} rmg(\Theta, \mathbf{X}, Y) rmg(\Theta', \mathbf{X}, Y) \quad (5)$$

Using (5) gives

$$\begin{aligned} \text{var}(mr) &= E_{\Theta, \Theta'} (\text{cov}_{\mathbf{X}, Y} rmg(\Theta, \mathbf{X}, Y) rmg(\Theta', \mathbf{X}, Y)) \\ &= E_{\Theta, \Theta'} (\rho(\Theta, \Theta') sd(\Theta) sd(\Theta')) \end{aligned} \quad (6)$$

where $\rho(\Theta, \Theta')$ is the correlation between $rmg(\Theta, \mathbf{X}, Y)$ and $rmg(\Theta', \mathbf{X}, Y)$ holding $\mathbf{X}, Y$ fixed and $sd(\Theta)$ is the standard deviation of $rmg(\Theta, \mathbf{X}, Y)$ holding $\mathbf{X}, Y$ fixed. Then,

$$\begin{aligned} \text{var}(mr) &= \bar{\rho} (E_{\Theta} sd(\Theta))^2 \\ &\leq \bar{\rho} E_{\Theta} \text{var}(\Theta) \end{aligned} \quad (7)$$

where $\bar{\rho}$ is the mean value of the correlation; that is,

$$\bar{\rho} = E_{\Theta, \Theta'}(\rho(\Theta, \Theta')sd(\Theta)sd(\Theta'))/E_{\Theta, \Theta'}(sd(\Theta)sd(\Theta'))$$

Write

$$\begin{aligned} E_{\Theta} \text{var}(\Theta) &\leq E_{\Theta}(E_{\mathbf{X}, Y} \text{rmg}(\Theta, \mathbf{X}, Y))^2 - s^2 \\ &\leq 1 - s^2. \end{aligned} \quad (8)$$

Putting (4), (7), and (8) together yield:

Theorem 2.3. *An upper bound for the generalization error is given by*

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

Although the bound is likely to be loose, it illustrates the same qualitative effect for random forests as VC-type bounds do for other types of classifiers. The two ingredients involved in the generalization error for random forests are the strength of the individual classifiers in the forest, and the correlation between them in terms of the random margin function. The $c/2$ ratio of the correlation divided by the square of the strength. In understanding the functioning of random forests, this ratio will be a helpful guide. The smaller it is, the better.

Definition 2.4. The $c/2$ ratio for a random forest is defined as

$$c/s^2 = \bar{\rho}/s^2.$$

There are implications in the above classification. The margin function is

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

The requirement that the strength is positive (see (4)) becomes similar to the familiar weak learning condition $E_{\mathbf{X}, Y} P_{\Theta}(h(\mathbf{X}, \Theta) = Y) > .5$. The random margin function is $2I(h(\mathbf{X}, \Theta) = Y) - 1$ and the correlation $\bar{\rho}$ is between $I(h(\mathbf{X}, \Theta) = Y)$ and $I(h(\mathbf{X}, \Theta') = Y)$. In particular, if the values for Y are taken to be +1 and -1, then

$$\bar{\rho} = E_{\Theta, \Theta'}[\rho(h(\cdot, \Theta), h(\cdot, \Theta'))]$$

so that $\bar{\rho}$ is the correlation between two different members of the forest averaged over the Θ, Θ' distribution.

For more than two classes, the measure of strength defined in (3) depends on the forest as well as the individual tree since in the forest it has determined $\hat{j}(\mathbf{X}, Y)$. Another approach

is possible. Write

$$\begin{aligned} PE^* &= P_{\mathbf{X},Y}(P_{\Theta}(h(\mathbf{X}, \Theta) = Y) - \max_{j \neq Y} P_{\Theta}(h(\mathbf{X}, \Theta) = j) < 0) \\ &\leq \sum_j P_{\mathbf{X},Y}(P_{\Theta}(h(\mathbf{X}, \Theta) = Y) - P_{\Theta}(h(\mathbf{X}, \Theta) = j) < 0). \end{aligned}$$

Define

$$s_j = E_{\mathbf{X},Y}(P_{\Theta}(h(\mathbf{X}, \Theta) = Y) - P_{\Theta}(h(\mathbf{X}, \Theta) = j))$$

so that the strength of the effect of class i over $\{h(\mathbf{x}, \Theta)\}$ relative to class j . Note that this definition of strength does not depend on the forest. Using Chebyshev's inequality, assuming all $s_j > 0$ lead to

$$PE^* \leq \sum_j \text{var}(P_{\Theta}(h(\mathbf{X}, \Theta) = Y) - P_{\Theta}(h(\mathbf{X}, \Theta) = j)) s_j^2 \quad (9)$$

and using identical arguments to those used in deriving (7), the variance in (9) can be expressed in terms of a regression correlation. I did not see a measure of the quality in (9) in our empirical study but I think the result would be interesting in a multiple class problem.

3. Using random features

Some random forests reported in the literature have consistently lower generalization error than others. For instance, random pleth selection (Dietterich, 1998) does better than bagging. Breiman's introduction of random noise into the output (Breiman, 1998c) also does better. But none of these three forests do as well as Adaboost (Freund & Schapire, 1996) or other algorithms that work by adapting reweighing (arcing) of the training set (see Breiman, 1998b; Dietterich, 1998; Barot & Kohavi, 1999).

To improve accuracy, the random noise injected has to minimize the correlation $\bar{\rho}$ while maintaining strength. The forest I tried here consisted of using randomly selected input or combination of inputs at each node to grow each tree. The resulting forest gave accuracy that compared favorably with Adaboost. This class of procedure has desirable characteristics:

- i It is accurate as good as Adaboost and sometimes better.
- ii It's relatively robust to outliers and noise.
- iii It's faster than bagging or boosting.
- iv It gives useful internal estimate of error, strength, correlation and variable importance.
- v It's simple and easily parallelized.

Amis and Geman (1997) grew a hall of trees for handwritten character recognition using random selection from a large number of geometrically defined features on the pixels at each node. Although my implementation is different and not problem specific, it is a heirloom that has provided the basis for my ideas.

3.1. Using out-of-bag estimates to monitor error, strength, and correlation

In many experiments with random forests, bagging is used in tandem with random feature selection. Each new training set is drawn with replacement, from the original training set. Then a tree is grown on the new training set using random feature selection. The tree grown are not pruned.

There are two reasons for using bagging. The first is that the use of bagging seems to enhance accuracy when random features are used. The second is that bagging can be used to give ongoing estimates of the generalization error (PE*) of the combined ensemble of trees, as well as estimates for the strength and correlation. The estimates are done out-of-bag, which is explained as follows.

Assume a method for constructing a classifier from a training set. Given a specific training set T , form bootstrap training set T_k , construct classifier $h(x, T_k)$ and let the ensemble form the bagged predictor. For each y, x in the training set, aggregate the score only for those classifiers for which T_k does not contain y, x . Call this the out-of-bag classifier. Then the out-of-bag estimate for the generalization error is the error rate of the out-of-bag classifier on the training set.

Tibshirani (1996) and Wolpert and Macread (1996), proposed using out-of-bag estimates as an ingredient in estimates of generalization error. Wolpert and Macread worked on regression type problem and proposed a number of methods for estimating the generalization error of bagged predictor. Tibshirani used out-of-bag estimates of variance to estimate generalization error for arbitrary classifier. The use of error estimates for bagged classifier in Breiman (1996b), give empirical evidence to show that the out-of-bag estimates is a accurate as using a test set of the same size as the training set. Therefore, using the out-of-bag error estimates remove the need for a test set.

In each bootstrap training set, about one-third of the instances are left out. Therefore, the out-of-bag estimates are based on combining only about one-third a many classifier as in the ongoing main combination. Since the error rate decrease as the number of combination increase, the out-of-bag estimates will tend to overestimate the current error rate. To get unbiased out-of-bag estimates, it is necessary to correct the point where the test error converge. But unlike cross-validation, where bias is predictable and is well known, the out-of-bag estimates are unbiased.

Strength and correlation can also be estimated using out-of-bag method. This gives internal estimates that are helpful in understanding classifier accuracy and how to improve it. The details are given in Appendix II. Another application is to the measure of variable importance (see Section 10).

4. Random forests using random input selection

The simple random forests with random features is formed by selecting at random, at each node, a small group of input variable to partition. Grow the resulting CART methodology to maximum size and do not prune. Denote this procedure by Forest-RI. The size F of the group is fixed. The total set of F features is varied. The result is only one randomly selected

Table 1. Data set summary.

Data set	Train size	Test size	Input	Classes
Glass	214		9	6
Breast cancer	699		9	2
Diabetes	768		8	2
Sonar	208		60	2
Vowel	990		10	11
Ionosphere	351		34	2
Vehicle	846		18	4
Soybean	685		35	19
German credit	1000		24	2
Image	2310		19	7
Ecoli	336		7	8
Vote	435		16	2
Litter	345		6	2
Letter	15000	5000	16	26
Sat-image	4435	2000	36	6
Zip-code	7291	2007	256	10
Waveform	300	3000	21	3
Twonorm	300	3000	20	2
Threernorm	300	3000	20	2
Ringnorm	300	3000	20	2

variable, i.e., $F = 1$. The second took F to be the smallest integer less than $\log_2 M + 1$, where M is the number of inputs.

More precisely, we used 13 smaller data sets from the UCI repository, 3 larger sets separated into training and testing sets and 4 synthetic data sets. These 10 sets were selected because I had used them in past research. Table 1 gives a brief summary.

On each of the 13 smaller data sets, the following procedure was used: a random 10% of the data was set aside. On the remaining data, random forests were grown, combining 100 trees, once with $F = 1$, and the second time with $F = \min(\log_2 M + 1)$. The set aside 10% was then predicted in each forest to get an estimate of error for both. The testing error selected corresponded to the lower value of the out-of-bag estimate in the two runs. This was repeated 100 times and the testing error averaged. The same procedure followed for the Adaboost runs which are based on combining 50 trees.

The error for 100 trees in random forests and 50 for Adaboost come from previous work. The out-of-bag estimate are based on only about a third of the data are in the forest. To get reliable estimate I opted for 100 trees. The second consideration is that growing random forests is much faster than growing the trees based on all inputs needed in Adaboost. Growing the 100 trees in random forests was considerably quicker than the 50 trees for Adaboost.

Table 2. Test error (%).

Dataset	Adaboost	Selection	Forest-RF (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
Vote	4.8	4.1	4.6	7.4
Litter	30.7	25.1	24.7	40.6
Letter	3.4	3.5	4.7	19.8
Sat-image	8.8	8.6	10.5	17.2
Zip-code	6.2	6.3	7.8	20.6
Waveform	17.8	17.2	17.3	34.0
USonorm	4.9	3.9	3.9	24.7
Threenuorm	18.8	17.5	17.5	38.4
Ringnorm	6.9	4.9	4.9	25.7

In the run on the larger datasets, the random forest results for the regression datasets were based on combining 100 trees; the zip-code procedure combined 200. For Adaboost, 50 trees were combined for the regression datasets and 100 for zip-code. The method is described in Breiman (1996) and adopted in Schapire et al. (1997). There were 50 runs. In each run, a training set of size 300 and a test set of size 3000 were generated. In random forest, 100 trees were combined in each run, 50 in Adaboost. The results of the experiments are given in Table 2.

The second column are the results selected from the regression group by the mean of leave-one-out-of-bag error. The third column is the test error using just one random feature group in the forest. The fourth column contains the leave-one-out-of-bag estimate of the generalization error of the individual trees in the forest computed for the best splitting (single or decision). This estimate is computed by using the left-out instance as a test set in each tree group and averaging the results over all trees in the forest.

The error rate using random input selection compares favorably with Adaboost. The comparison might be even more favorable if the search is over more values of F instead of the pre-specified. But the procedure is not optimal in the sense of F . The average absolute difference between the error rate using $F = 1$ and the higher values of F is less than 1%. The difference is mostly pronounced on the three large datasets.

Table 3. Test set error (%).

Dataset	AdaBoost	Forest-RC		
		Selection	Two feature	One tree
Glass	22.0	24.4	23.5	42.4
Breast cancer	3.2	3.1	2.9	5.8
Diabetes	26.6	23.0	23.1	32.1
Sonar	15.6	13.6	13.8	31.7
Vowel	4.1	3.3	3.3	30.4
Ionosphere	6.4	5.5	5.7	14.2
Vehicle	23.2	23.1	22.8	39.1
German credit	23.5	22.8	23.8	32.6
Image	1.6	1.6	1.8	6.0
Ecoli	14.8	12.9	12.4	25.3
Vowel	4.8	4.1	4.0	8.6
Letter	30.7	27.3	27.2	40.3
Letter	3.4	3.4	4.1	23.8
Satellite-image	8.8	9.1	10.2	17.3
Zip-code	6.2	6.2	7.2	22.7
Waveform	17.8	16.0	16.1	33.2
Two-norm	4.9	3.8	3.9	20.9
Three-norm	18.8	16.8	16.9	34.8
Ring-norm	6.9	4.8	4.6	24.6

8.5%, on the letter dataset 3.0%, but the zip-code test set error did not decrease. Acting on an informed hunch, I tried Forest-RL with $F = 25$. The zip-code test set error dropped to 5.8%. The error of the low-dimensional test set error so far achieved on the other three datasets by the ensemble.

5.1. Categorical variables

Some or all of the input variable may be categorical and include values of discrete addition combination of variable, we need to do not have categorical will be treated as the can be combined with numerical variable. My approach is that each time a categorical variable is selected to split on at a node, to select a random subset of the categories of the variable, and do not a subset of the variable has one when the categorical value of the variable is in the subset and zero otherwise.

Since a categorical variable with I value can be coded into $I - 1$ dimensional $0 - 1$ variable, we make the variable $I - 1$ time a probable a numerical variable to be selected in node splitting. When many of the variable are categorical, using a low value of F results in low correlation, but a low value of F may be increased to about three times in $(\log_2 M + 1)$ to get enough strength to provide good test set accuracy.

For instance, on the DNA data set having 60 forced categorical labels, 2,000 examples in the training set and 1,186 in the test set, using Fore-RI with $F = 20$ gave a test set error rate of 3.6% (4.2% for Adaboost). The soybean data has 685 examples, 35 variables, 19 classes, and 15 categorical variables. Using Fore-RI with $F = 12$ gave a test set error of 5.3% (5.8% for Adaboost). Using Fore-RC with combination of 3 and $F = 8$ gave an error of 5.5%.

One advantage of this approach is that it gets around the difficulty of doing categorical variable handling. In the two-class problem, this can be avoided by using the device proposed in Breiman et al. (1985) which reduces the search for the best categorical split to an $O(I)$ computation. For more classes, the search for the best categorical split is an $O(2^{I-1})$ computation. In the random forest implementation, the computation for an categorical variable involves only the election of a random subset of the categories.

6. Empirical results on strength and correlation

The purpose of this section is to look at the effect of strength and correlation on the generalization error. Another aspect that we wanted to get more understanding of was the lack of evidence in the generalization error of the group size F . To conduct an empirical study of the effect of strength and correlation in a series of data sets, out-of-bag estimate of the strength and correlation, as described in Section 3.1, were used.

We begin by running Fore-RI on the sonar data (60 inputs, 208 examples) using from 1 to 50 inputs. In each iteration, 10% of the data are split off as a test set. Then F , the number of random inputs selected at each node, was varied from 1 to 50. For each value of F , 100 trees were grown to form a random forest and the terminal value of test set error, strength, correlation, etc. recorded. Eight iterations were done, each time removing a random 10% of the data for a test set, and all results averaged over the 80 repetitions. Altogether, 400,000 trees were grown in this experiment.

The top graph of Figure 1 plots the value of strength and correlation vs. F . The results are fascinating. Particularly for $F = 4$ the strength remains constant; adding more inputs does not help. But the correlation continues to increase. The second graph plots the test set error and the out-of-bag estimate of the generalization error against F . The out-of-bag estimates are more variable. Both have the same behavior: a small drop from $F = 1$ to $F = 4$, and then a general, gradual increase. This increase in error is all due to the beginning of the correlation region for the strength.

Figure 2 has plots for similar runs on the breast data set where features consisting of random combination of three inputs are used. The number of the features was varied from 1 to 25. Again, the correlation has a local maximum, while the strength is almost constant, only having the minimum error at $F = 1$. The surprise in the error graphs is the relative constancy of the strength. Since the correlation are quickly and readily increasing, the local error occurs only a few inputs or features are used.

Since the larger data sets seemed to have a different behavior than the smaller, we ran a similar experiment on the australian data set. The number of features, each consisting of a random subset of inputs, was varied from 1 to 25, and for each, 100 classifiers were combined. The results are shown in Figure 3. The results differ from those on the smaller



Figure 1. Effect of number of input on output.

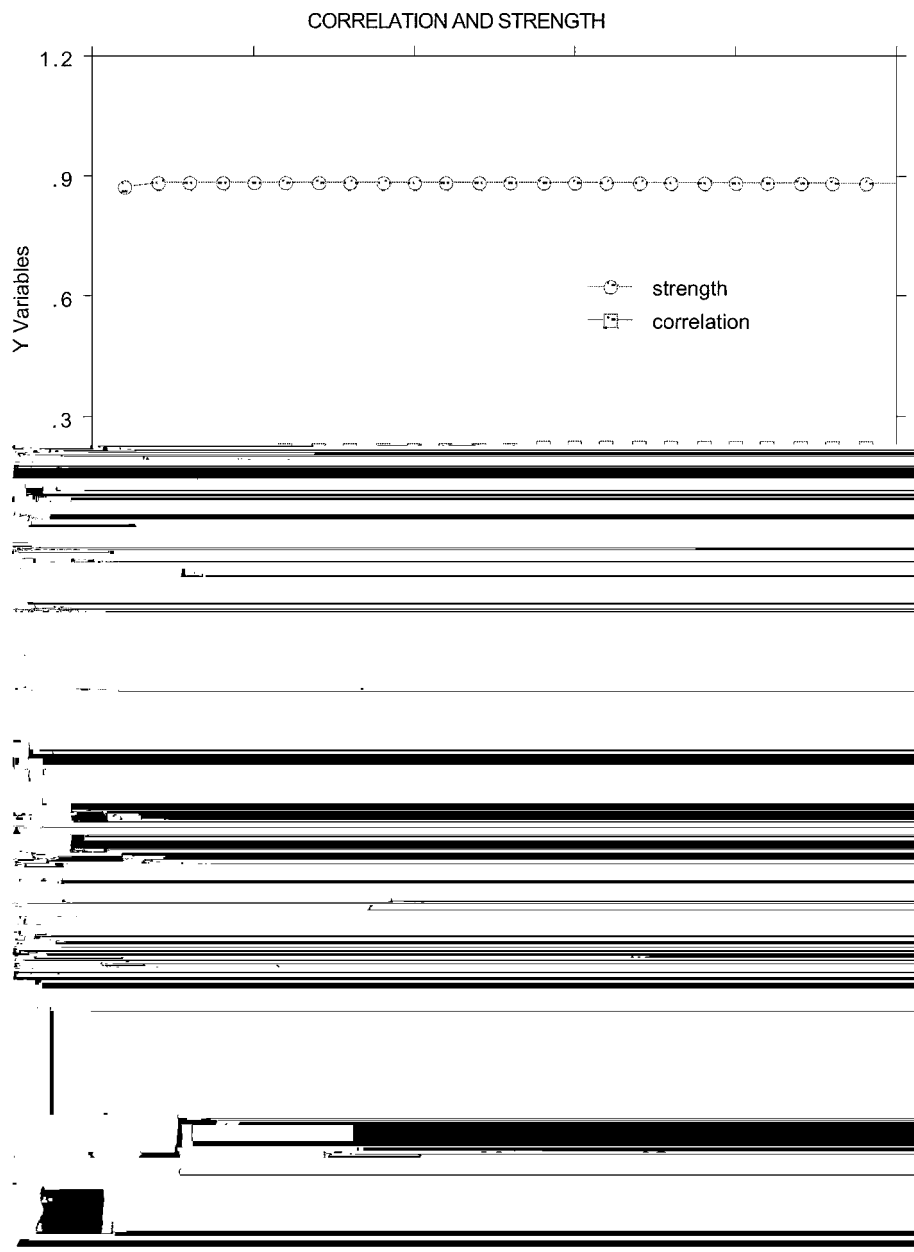


Figure 2. Effect of the number of features on the breast cancer data set.

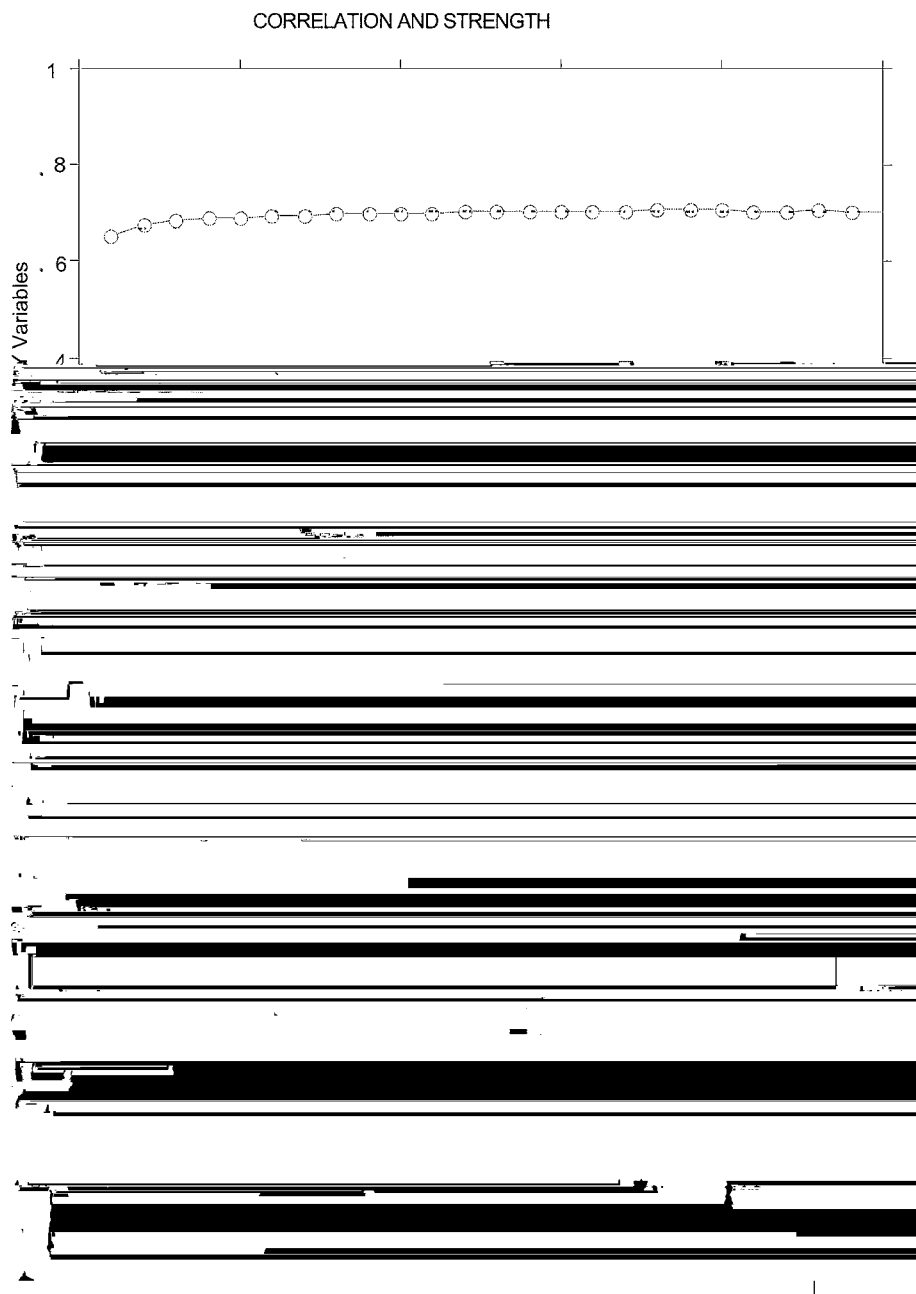


Figure 3. Effect of number of feature on elliptic data.

data set. Both the correlation and strength have a small bias toward increase. The error rate has a slight decrease. We conjecture that with larger and more complex data sets, the strength continues to increase longer before it plateaus.

Our results indicate that better (lower generalization error) random forests have lower correlation between classifiers and higher strength. The randomness induced in tree construction has no aim for lower correlation while maintaining reasonable strength. This conclusion has been hinted at in previous work. Dietterich (1998) has measured the dispersion of an ensemble and notes that more accurate ensembles have larger dispersion. Freund (personal communication) believes that one reason why Adaboost works so well is that at each step it tries to decompose the new classifier from the current one. Amis et al. (1999) give an analysis of how the Adaboost algorithm is aimed at keeping the covariance between classifiers small.

7. Conjecture: Adaboost is a random forest

Various classifiers can be modified to be both a training set and a set of weights on the training set. Consider the following random forest: a large collection of K different sets of non-negative m -one weights on the training set is defined. Denote the i th weight by $\mathbf{w}(1), \mathbf{w}(2), \dots, \mathbf{w}(K)$. Corresponding to the i th weight are probabilities $p(1), p(2), \dots, p(K)$ whose sum is one. Draw from the integer $1, \dots, K$ according to the probabilities. The outcome is Θ . If $\Theta = k$ grow the classifier $h(\mathbf{x}, \Theta)$ using the training set with weights $\mathbf{w}(k)$.

In its original version, Adaboost (Freund & Schapire, 1996) is a deterministic algorithm that elects the weights on the training set for input to the new classifier based on the misclassification in the previous classifier. In our experiment, random forests were produced as follows: Adaboost was run 75 times on a data set producing sets of non-negative m -one weights $\mathbf{w}(1), \mathbf{w}(2), \dots, \mathbf{w}(50)$ (the first 25 were discarded). The probability for the k th set of weights is proportional to $Q(\mathbf{w}_k) = \log[(1 - \text{error}(k))/\text{error}(k)]$ where $\text{error}(k)$ is the $\mathbf{w}(k)$ weighted training set error of the k th classifier. Then the forest is run 250 times.

This was repeated 100 times on a fixed data set, each time leaving out 10% a new set and then averaging the new set error. On each data set, the Adaboost error rate was very close to the random forest error rate. A typical result is on the Wisconsin Breast Cancer data where Adaboost produced an average of 2.91% error and the random forest produced 2.94%.

In the Adaboost algorithm, $\mathbf{w}(k+1) = \phi(\mathbf{w}(k))$ where ϕ is a function determined by the base classifier. Denote the k th classifier by $h(\mathbf{x}, \mathbf{w}_k)$. The score of the k th classifier is weighted by $Q(\mathbf{w}_k)$ on the normalized score for classifier j at $\mathbf{x}$ equal

$$\sum_k I(h(\mathbf{x}, \mathbf{w}_k) = j) Q(\mathbf{w}_k) / \sum_k Q(\mathbf{w}_k). \quad (10)$$

For an function f defined on the weight space, define the operator $\mathbf{T}f(\mathbf{w}) = f(\phi(\mathbf{w}))$. We conjecture that $\mathbf{T}$ is ergodic with invariant measure $\pi(d\mathbf{w})$. Then (10) will converge to $E_{Q\pi}[I(h(\mathbf{x}, \mathbf{w}) = j)]$ where the distribution $Q\pi(d\mathbf{w}) = Q(\mathbf{w})\pi(d\mathbf{w}) / \int Q(\mathbf{v})\pi(d\mathbf{v})$. If this conjecture is true, then Adaboost is equivalent to a random forest where the weights on the training set are elected at random from the distribution $Q\pi$.

It is important to be plain that Adaboost does not mean that more trees are added to the ensemble, an experimental fact that has been proving. There is some experimental evidence that Adaboost may be more effective if more data and more time (Grove & Schürman, 1998), but the experiments were done using a very simple base classifier and may not carry over to the case of more sophisticated base classifiers. More experience running Adaboost on more than 1,000 trees on a number of datasets is what the experimental error converges to an empirical one.

The truth of this conjecture does not solve the problem of how Adaboost selects the favorable distribution on the weightspace that it does. Note that the distribution of the weights will depend on the training set. In the usual random forest, the distribution of the random vector does not depend on the training set.

8. The effects of output noise

Dieyerich (1998) has shown that when a fraction of the output labels in the training set are randomly altered, the accuracy of Adaboost degenerates, while bagging and random partition election are more immune to the noise. Since some noise in the output is often present, robustness to the noise is a desirable property. Following Dieyerich (1998) the following experiments were done which changed about one in ten class labels (injecting 5% noise).

For each dataset in the experiments, 10% of the random partitions were made on the remaining training set. The results on the training set are shown in Table 4. The second row is on a noise injection of the training set. The noise injection is given by changing, at random, 5% of the class labels into an alternative class label chosen uniformly from the other labels.

This is repeated 50 times using Adaboost (deterministic version), Forest-RI and Forest-RC. The results are averaged over the 50 repetitions and the percent increase in the noise is computed. In both random forests, the increase in the number of features required to reach the error in the Section 5 and 6 experiments. Because of the length of the results, only the 9 most significant results are shown. Table 4 gives the increase in error rate due to the noise.

Table 4. Increase in error rate due to noise (%).

Dataset	Adaboost	Forest-RI	Forest-RC
Glass	1.6	.4	-.4
Breast cancer	43.2	1.8	11.1
Diabetes	6.8	1.7	2.8
Sonar	15.1	-6.6	4.2
Ionosphere	27.7	3.8	5.7
Soybean	26.9	3.2	8.5
Ecoli	7.5	7.9	7.8
Vowel	48.9	6.3	4.6
Litter	10.3	-.2	4.8

Adaboos performance markedly with 5% noise, while the random forest procedure generally has small change. The effect on Adaboos performance of data dependent, with the two most difficult data sets, glass and ecoli, along with diabetes, less affected by the noise. The Adaboos algorithm iteratively increases the weight on the instance most recently misclassified. In instance having incorrect class label will persist in being misclassified. Then, Adaboos will concentrate increasing weight on the erroneous instance and become warped. The random forest procedure does not concentrate weight on any subset of the instance and the noise effect is smaller.

9. Data with many weak inputs

Data sets with many weak inputs are becoming more common, i.e. in medical diagnosis, document retrieval, etc. The common characteristic is no single input or small group of inputs can distinguish between the classes. This type of data is difficult for the neural classifier, neural network and tree.

To see if there is a possibility that Forest-R1 method can work, the following 10 classes, 1,000 binary input data, was generated: (randomly a uniform random number, selected and each time it appear)

```

do j=1,10
do k=1,1000
p(j,k)=.2*rand+.01
end do
end do

do j=1,10
do i=1, nint(400*rand) !nint=nearest integer
k=nint(1000*rand)
p(j,k)=p(j,k)+.4*rand
end do
end do

do n=1,N
j=nint(10*rand)
do m=1,1000
if (rand<p(j,m)) then
(m,n)=1
else
(m,n)=0
end if
(n)=j ! (n) is the class label of the nth example
end do
end do

```

This code generates a set of probabilities $\{p(j, m)\}$ where j is the class label and m is the input number. Then the inputs for a class j example are a string of M binary variables such that the m th variable has probability $p(j, m)$ of being one.

For the training set, $N = 1,000$. A 4,000 example set is constructed using the same $\{p(j, k)\}$. Examination of the code shows that each class has higher underlying probability at certain locations. But the total over all classes of the probability is about 2,000, so there is significant overlap. A missing one knows all of the $\{p(j, k)\}$, the Bayes error rate for the particular $\{p(j, m)\}$ computed is only 1.0%.

Since the inputs are independent of each other, the Naïve Bayes classifier, which estimates the $\{p(j, k)\}$ from the training data is supposedly optimal and has an error rate of 6.2%. This is not an endorsement of Naïve Bayes, since it would be easy to create a dependence between the inputs which would increase the Naïve Bayes error rate. I varied the inputs which became the Bayes error rate and the Naïve Bayes error rate are easy to compare.

I varied the number of Forest-RF with $F = 1$. It converged very slowly and by 2,500 iterations, when it was stopped, it had still not converged. The test set error was 10.7%. The strength was .069 and the correlation .012 with a $c/2$ ratio of 2.5. Even though the strength was low, the almost zero correlation means that the weak learners were adding small increments of accuracy as the iteration proceeded.

Clearly, the weak learners increased an increase in strength while keeping the correlation low. Forest-RF was run again using $F = \ln(\log_2 M + 1) = 10$. The results were encouraging. It converged after 2,000 iterations. The test set error was 3.0%. The strength was .22, the correlation .045 and $c/2 = .91$. Going further, Forest-RF was run with $F = 25$ and stopped after 2,000 iterations. The test set error was 2.8%. Strength was .28, correlation .065 and $c/2 = .83$.

It's interesting that Forest-RF could produce error rates not far above the Bayes error rate. The individual classifiers are weak. For $F = 1$, the average tree error rate is 80%; for $F = 10$, it is 65%; and for $F = 25$, it is 60%. Forest seems to have the ability to work with the weak classifiers as long as their correlation is low. A comparison using Adaboost was tried, but I can't get Adaboost to run on this data because the base classifiers are too weak.

10. Exploring the random forest mechanism

A forest of trees is impenetrable as far as simple interpretation of its mechanism goes. In some applications, analysis of medical experiments for example, it is critical to understand the interaction of variables that is producing the predictive accuracy. A solution to this problem is made by using internal out-of-bag estimates, and examination by rerunning only selected variables.

Suppose there are M input variables. After each tree is constructed, the values of the m th variable in the out-of-bag example are randomly permuted and the out-of-bag data is run down the corresponding tree. The classification given for each x_n has a ratio of bagged. This is repeated for $m = 1, 2, \dots, M$. At the end of the run, the plurality of out-of-bag classifications for x_n with the m th variable noted is compared with the true class label of x_n to give a misclassification rate.

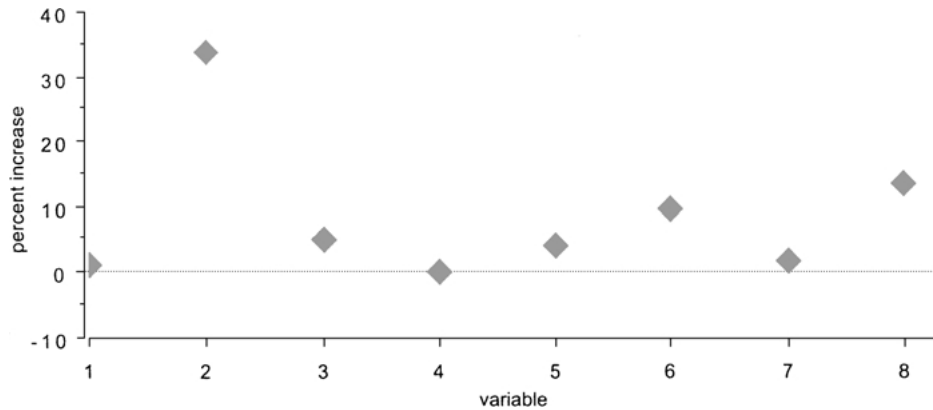


Figure 4. Measure of variable importance: diabete data.

The only place the percent increase in misclassification rate is compared to the out-of-bag rate (with all variable intact). We get the estimate by using a single run of a forest with 1,000 trees and no variable. The procedure is illustrated by the sample.

In the diabete data set, using only single variable with $F = 1$, the rise in error due to the omitting of variable is given in Figure 4.

The second variable appears to be far more important followed by variable 8 and variable 6. Running the random forest in 100 repetitions using only variable 2 and leaving out 10% each time of mean variable error gave an error of 29.7%, compared with 23.1% using all variable. But when variable 8 is added, the error fell only to 29.4%. When variable 6 is added to variable 2, the error fell to 26.4%.

The reason that variable 6 seems important, is no help once variable 2 is entered is a characteristic of highly dependent variable affect prediction error in random forest. Say there are two variable x_1 and x_2 which are identical and carry identical information. Because each gets picked with about the same frequency in a random forest, omitting each separately will result in the same increase in error rate. But once x_1 is entered as a predictive variable, adding x_2 in addition will not produce an decrease in error rate. In the diabete data set, the 8th variable carries some of the same information as the second. So it does not add predictive accuracy when combined with the second.

The relative magnitude of rise in error rate are fairly variable with respect to the input features used. The experiment about the repeated combination of three input variables with $F = 2$. The results are in Figure 5.

Another interesting sample is the voting data. This has 435 examples corresponding to 435 Congressmen and 16 variables recording their preference on 16 issues. The class variable is Republican or Democrat. To see which issues were most important, we again ran the omitting variable program generating 1,000 trees. The lowest error rate on the original data was given using single input variables with $F = 5$, of the parameters used in the run. The results are in Figure 6.

Variable 4 and only the error triple if variable 4 is omitted. We reran this data set using only variable 4. The variable error is 4.3%, about the same as if all variable were used. The

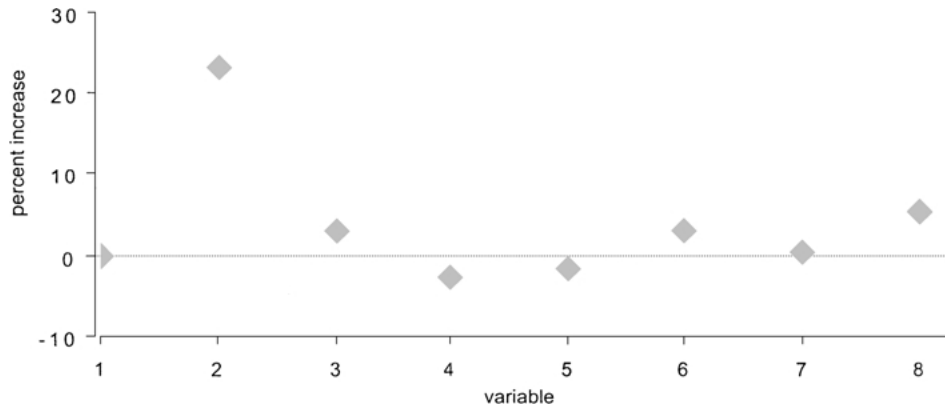


Figure 5. Measure of variable importance-diabetic data.

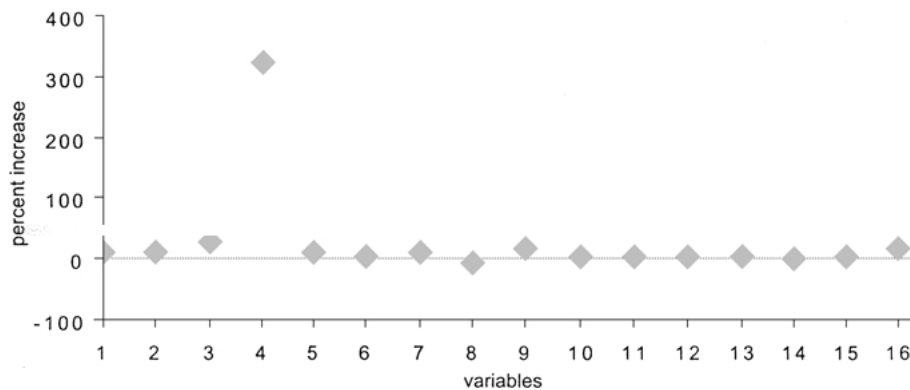


Figure 6. Measure of variable importance -cancer data.

one on 4 separate Republican from Democrat, almost as well as the one on 4 combined with the one on all other 15 issues.

The approach given in this section is only a beginning. More research will be necessary to understand how to give a more complete picture.

11. Random forests for regression

Random forests for regression are formed by growing trees depending on a random vector Θ which has the tree predictor $h(\mathbf{x}, \Theta)$ take on numerical value as opposed to class label. The output values are numerical and we assume that the training data is independent drawn from the distribution of the random vector $Y, \mathbf{X}$. The mean-squared generalization error for an numerical predictor $h(\mathbf{x})$ is

$$E_{\mathbf{X}, Y}(Y - h(\mathbf{X}))^2 \quad (11)$$

The random forest predictor is formed by taking the average over k of the trees $\{h(\mathbf{x}, \Theta_k)\}$. Similarly, the classification can be, the following hold :

Theorem 11.1. *As the number of trees in the forest goes to infinity, almost surely,*

$$E_{\mathbf{X},Y}(Y - \text{avg}_k h(\mathbf{X}, \Theta_k))^2 \rightarrow E_{\mathbf{X},Y}(Y - E_{\Theta} h(\mathbf{X}, \Theta))^2. \quad (12)$$

Proof: see Appendix I. □

Denote the right hand side of (12) as $PE^*(\text{forest})$. The generalization error of the forest. Define the average generalization error of a tree as :

$$PE^*(\text{tree}) = E_{\Theta} E_{\mathbf{X},Y}(Y - h(\mathbf{X}, \Theta))^2$$

Theorem 11.2. *Assume that for all Θ , $EY = E_{\mathbf{X}}h(\mathbf{X}, \Theta)$. Then*

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

where $\bar{\rho}$ is the weighted correlation between the residuals $Y - h(\mathbf{X}, \Theta)$ and $Y - h(\mathbf{X}, \Theta')$ where Θ, Θ' are independent.

Proof:

$$\begin{aligned} PE^*(\text{forest}) &= E_{\mathbf{X},Y}[E_{\Theta}(Y - h(\mathbf{X}, \Theta))]^2 \\ &= E_{\Theta} E_{\Theta'} E_{\mathbf{X},Y}(Y - h(\mathbf{X}, \Theta))(Y - h(\mathbf{X}, \Theta')) \end{aligned} \quad (13)$$

The term on the right in (13) is a covariance and can be written as :

$$E_{\Theta} E_{\Theta'}(\rho(\Theta, \Theta') sd(\Theta) sd(\Theta'))$$

where $sd(\Theta) = \sqrt{E_{\mathbf{X},Y}(Y - h(\mathbf{X}, \Theta))^2}$. Define the weighted correlation as :

$$\bar{\rho} = E_{\Theta} E_{\Theta'}(\rho(\Theta, \Theta') sd(\Theta) sd(\Theta')) / (E_{\Theta} sd(\Theta))^2 \quad (14)$$

Then

$$PE^*(\text{forest}) = \bar{\rho} (E_{\Theta} sd(\Theta))^2 \leq \bar{\rho} PE^*(\text{tree}). \quad \square$$

Theorem (11.2) pinpoints the requirement for accurate regression forest, i.e. correlation between residual and local error tree. The random forest decrease the average error of the tree employed by the factor $\bar{\rho}$. The randomization employed need to aim at local correlation.

12. Empirical results in regression

In regression forests the random features selection on top of bagging. Therefore, we can use the monitoring provided by out-of-bag estimation to give an estimate of $PE^*(\text{forest})$, $PE^*(\text{tree})$ and $\bar{\rho}$. These are derived similarly to the estimation in classification. Throughout, features are formed by a random linear mix of the input variables. We comment later on how many of the features are used to determine the split at each node. The more features used, the lower $PE^*(\text{tree})$ but the higher $\bar{\rho}$. In our empirical study the following datasets are used, see Table 5.

Of the datasets, the Boston Housing, Abalone and Serum are available at the UCI repository. The Robot Arm dataset is provided by Michael Jordan. The last three datasets are synthetic. The original in Friedman (1991) and are also described in Breiman (1998b). These are the same datasets used to compare adaptive bagging to bagging (see Breiman, 1999), except that one synthetic dataset (Peak20), which was found anomalous by other researchers and myself, is eliminated.

The first three datasets listed are moderate in size and the test error is estimated by leaving out a random 10% of the instances, training on the remaining 90% and using the left-out 10% as a test set. This is repeated 100 times and the test error averaged. The abalone dataset is larger with 4,177 instances and 8 input variables. It originally came with 25% of the instances as a test set. We ran this dataset leaving out a random selected 25% of the instances as a test set, repeated this 10 times and averaged.

Table 6 gives the test mean-squared error for bagging, adaptive bagging and the random forest. The errors are all running 25 features, each a random linear combination of the randomly selected inputs, to split each node, each feature a random combination of the inputs. All random splits are enforced, combined 100 times. In all datasets, the results don't split if the node size is < 5 is enforced.

An interesting difference between regression and classification is that the correlation increase quite slowly as the number of features increased. The major effect is the decrease in $PE^*(\text{tree})$. Therefore, a relatively large number of features are required to reduce $PE^*(\text{tree})$ and get near optimal test error.

Table 5. Dataset summary.

Dataset	Nr. input	#Training	#Test
Boston Housing	12	506	10%
One	8	330	10%
Serum	4	167	10%
Abalone	8	4177	25%
Robot Arm	12	15,000	5000
Friedman#1	10	200	2000
Friedman#2	4	200	2000
Friedman#3	4	200	2000

Table 6. Mean-squared test error.

Data set	Bagging	Adapt. bag	Forest
Boston Housing	11.4	9.7	10.2
One	17.8	17.8	16.3
Servo $\times 10 - 2$	24.5	25.1	24.6
Abalone	4.9	4.9	4.6
Robot Arm $\times 10 - 2$	4.7	2.8	4.2
Friedman #1	6.3	4.1	5.7
Friedman #2 $\times 10 + 3$	21.5	21.5	19.6
Friedman #3 $\times 10 - 3$	24.8	24.8	21.6

The results shown in Table 6 are mixed. Random forest-random features is always better than bagging. In data sets for which adaptive bagging gives a sharp decrease in error, the decrease produced by forest are not pronounced. In data sets in which adaptive bagging gives no improvement over bagging, forest produce improvement.

For the same number of inputs combined, over a wide range, the error does not change much with the number of features. If the number is too small, $PE^*(tree)$ becomes too large and the error goes up. If the number is too large, the correlation goes up and the error again increases. The in-between range is still large. In this range, as the number of features goes up, the correlation increases, but $PE^*(tree)$ compensates by decreasing.

Table 7 gives the test error, the out-of-bag error estimate, and the OB estimate for $PE^*(tree)$ and the correlation.

As expected, the OB Error estimates are consistently high. It is likely in the robot arm data, but I believe that this is an artifact caused by overparameterizing and the test set where the test set makes a slightly higher error rate than the training set.

As an experiment, I turned off the bagging and replaced it by randomizing outputs (Breiman, 1998b). In this procedure, mean-zero Gaussian noise is added to each of the outputs. The standard deviation of the noise is equal to the standard deviation of the

Table 7. Error and OB estimate.

Data Set	Test error	OB error	$PE^*(tree)$	Cor.
Boston Housing	10.2	11.6	26.3	.45
One	16.3	17.6	32.5	.55
Servo $\times 10 - 2$	24.6	27.9	56.4	.56
Abalone	4.6	4.6	8.3	.56
Robot Arm $\times 10 - 2$	4.2	3.7	9.1	.41
Friedman #1	5.7	6.3	15.3	.41
Friedman #2 $\times 10 + 3$	19.6	20.4	40.7	.51
Friedman #3 $\times 10 - 3$	21.6	22.9	48.3	.49

Table 8. Mean-squared error.

Data set	With bagging	With Noise
Boston Housing	10.2	9.1
Online	17.8	16.3
Servo $\times 10 - 2$	24.6	23.2
Abalone	4.6	4.7
Robot Arm $\times 10 - 2$	4.2	3.9
Friedman #1	5.7	5.1
Friedman #2 $\times 10 + 3$	19.6	20.4
Friedman #3 $\times 10 - 3$	21.6	19.8

output. Similar to the bagging experiment, tree construction is done using 25 features, each a random linear combination of 40 randomly selected inputs, to split each node. The results are given in Table 8.

The error rate on the test data sets are the lowest of the data sets. Overall, adding output noise to the random feature selection performs better than bagging. This is still a rare case of the flexibility of the random forest using various combinations of random noise can be added to each data set to be used.

13. Remarks and conclusions

Random forests are an effective tool in prediction. Because of the Law of Large Number, they do not overfit. Injecting the right kind of randomness makes them accurate classifiers and regressors. Furthermore, the framework in terms of strength of the individual predictors and their correlation gives insight into the ability of the random forest to predict. Using out-of-bag estimation makes concrete the otherwise theoretical value of strength and correlation.

For all while, the conventional thinking is that forests could not compete with arcing type algorithms in terms of accuracy. Our results dispel this belief, by leading to interesting questions. Boosting and arcing algorithms have the ability to reduce bias as well as variance (Schapire et al., 1998). The adaptive bagging algorithm in regression (Breiman, 1999) is designed to reduce bias and operate effectively in classification as well as in regression. But, like arcing, it also changes the training set as it progresses.

Forests give rise to competition with boosting and adaptive bagging, and do not progress to change the training set. Their accuracy indicates that they are also good classifiers. The mechanism for this is not obvious. Random forests may also be identified as a Bayesian procedure. Although I do not have this as a first line of exploration, if it could explain the bias reduction, I might become more of a Bayesian.

Random inputs and random feature products produce good results in classification, less so in regression. The only type of randomness used in this study is bagging and random features. It may well be that other types of injected randomness give better results. For instance, one of the referees has suggested use of random Boolean combination of features.

An almost obvious question is whether gain in accuracy can be gotten by combining random features with bootstrapping. For the larger data sets, it seems that significant lower error rates are possible. On some runs, the good error rate was 5.1% on the zip-code data, 2.2% on the letter data and 7.9% on the australian data. The improvement was also on the smaller data sets. More work is needed on this; but it does suggest that different injection of randomness can produce better results.

A recent paper (Breiman, 2000) has suggested an interesting approach for the classification problem, random forests are equivalent to a kernel acting on the hypermargin. Arguments are given that randomness (low correlation) enforces the minimality of the kernel, while strength enhances a desirable kernel as a bracketed bounded operator. Hopefully, this sheds light on the dual role of correlation and strength. The theoretical framework given by Kleinberg (2000) for Stochastic Discrimination may also help understanding.

Appendix I: Almost sure convergence

Proof of theorem 1.2: If we take the hypothesis that there is a set of probability zero C on the sequence space $\Theta_1, \Theta_2, \dots$ such that $\Theta \in C$, for all $\mathbf{x}$,

$$\frac{1}{N} \sum_{n=1}^N I(h(\Theta_n, \mathbf{x}) = j) \rightarrow P_\Theta(h(\Theta, \mathbf{x}) = j).$$

For a fixed training set and fixed Θ , the set of all $\mathbf{x}$ such that $h(\Theta, \mathbf{x}) = j$ is a union of hyper-rectangle. For all $h(\Theta, \mathbf{x})$ there is only a finite number K of such union of hyper-rectangle, denoted by $S_1, \dots, S_K$. Define $\varphi(\Theta) = k$ if $\{\mathbf{x} : h(\Theta, \mathbf{x}) = j\} = S_k$. Let N_k be the number of time that $\varphi(\Theta_n) = k$ in the first N trials. Then

$$\frac{1}{N} \sum_{n=1}^N I(h(\Theta_n, \mathbf{x}) = j) = \frac{1}{N} \sum_k N_k I(\mathbf{x} \in S_k)$$

By the Law of Large Number,

$$N_k = \frac{1}{N} \sum_{n=1}^N I(\varphi(\Theta_n) = k)$$

converges a.s. to $P_\Theta(\varphi(\Theta) = k)$. Taking union of all the sets on which convergence does not occur for some value of k gives a set C of zero probability such that $\Theta \in C$,

$$\frac{1}{N} \sum_{n=1}^N I(h(\Theta_n, \mathbf{x}) = j) \rightarrow \sum_k P_\Theta(\varphi(\Theta) = k) I(\mathbf{x} \in S_k).$$

The right hand side is $P_\Theta(h(\Theta, \mathbf{x}) = j)$. □

Proof of theorem 9.1: There are at most ℓ hyper-rectangle $R_1, \dots, R_K$, such that if $\bar{y}_k$ is the average of the training ℓ -values for all training input vectors in R_k then $h(\Theta, \mathbf{x})$ has one of the values $I(\mathbf{x} \in S_k)\bar{y}_k$. The rest of the proof parallel that of Theorem 1.2. $\square$

Appendix II: Out-of bag estimates for strength and correlation

At the end of a combination run, let

$$Q(\mathbf{x}, j) = \sum_k I(h(\mathbf{x}, \Theta_k) = j; (y, \mathbf{x}) \notin T_{k,B}) / \sum_k I((y, \mathbf{x}) \notin T_{k,B}).$$

Then, $Q(\mathbf{x}, j)$ is the out-of-bag proportion of observations $\mathbf{x}$ for class j , and is an estimate for $P_\Theta(h(\mathbf{x}, \Theta) = j)$. From Definition 2.1 the strength is the expectation of

$$P_\Theta(h(\mathbf{x}, \Theta) = y) - \max_{j \neq y} P_\Theta(h(\mathbf{x}, \Theta) = j)$$

Substituting $Q(\mathbf{x}, j)$, $Q(\mathbf{x}, y)$ for $P_\Theta(h(\mathbf{x}, \Theta) = j)$, $P_\Theta(h(\mathbf{x}, \Theta) = y)$ in this latter expression and taking the average over the training ℓ gives the strength estimate.

From Eq. (7),

$$\bar{\rho} = \text{ar}(mr) / (E_\Theta sd(\Theta))^2.$$

The variance of mr is

$$E_{\mathbf{X}, Y} [P_\Theta(h(\mathbf{x}, \Theta) = y) - \max_{j \neq y} P_\Theta(h(\mathbf{x}, \Theta) = j)]^2 - s^2 \quad (\text{A1})$$

where s is the strength. Replacing the term in (A1) by the average over the training ℓ of

$$(Q(\mathbf{x}, y) - \max_{j \neq y} Q(\mathbf{x}, j))^2$$

and substituting the out-of-bag estimate of s gives the estimate of $\text{ar}(mr)$. The standard deviation is given by

$$sd(\Theta) = [p_1 + p_2 + (p_1 - p_2)^2]^{1/2} \quad (\text{A2})$$

where

$$\begin{aligned} p_1 &= E_{\mathbf{X}, Y}(h(\mathbf{X}, \Theta) = Y) \\ p_2 &= E_{\mathbf{X}, Y}(h(\mathbf{X}, \Theta) = \hat{j}(\mathbf{X}, Y)) \end{aligned}$$

After the k th class is constructed, $Q(\mathbf{x}, j)$ is computed, and then to compute $\hat{j}(\mathbf{x}, y)$ for each example in the training set. Then, let p_1 be the average over all $(y, \mathbf{x})$ in the training set of p_1 in the k th bagged training set of $I(h(\mathbf{x}, \Theta_k) = y)$. Then p_2 is the similar average of $I(h(\mathbf{x}, \Theta_k) = \hat{j}(\mathbf{x}, y))$. So by the law of large numbers (A2) the variance of $sd(\Theta_k)$. The average of the $sd(\Theta_k)$ over all k is the variance of $sd(\Theta)$.

References

- Amis, Y. & Geman, D. (1997). Shape quantization and recognition with randomized trees. *Neural Computation*, 9, 1545–1588.
- Amis, Y., Blanchard, G., & Wilder, K. (1999). Multiple randomized classifiers: MRCL Technical Report, Department of Statistics, University of Chicago.
- Baer, E. & Kohavi, R. (1999). An empirical comparison of voting classifier algorithms. *Machine Learning*, 36(1/2), 105–139.
- Breiman, L. (1996a). Bagging predictor. *Machine Learning* 26(2), 123–140.
- Breiman, L. (1996b). Out-of-bag estimation, ftp://pds.berkeley.edu/pub/breiman/OOBestimate.p
- Breiman, L. (1998a). Arcing classifier (discussion paper). *Annals of Statistics*, 26, 801–824.
- Breiman, L. (1998b). Randomizing output to increase prediction accuracy. Technical Report 518, March 1, 1998, Statistics Department, UCB (in press, Machine Learning).
- Breiman, L. 1999. Using adaptive bagging to debias regression. Technical Report 547, Statistics Department, UCB.
- Breiman, L. 2000. Some interesting theory for prediction ensemble. Technical Report 579, Statistics Department, UCB.
- Dietterich, T. (1998). An experimental comparison of three methods for combining ensemble of decision trees: Bagging, boosting and randomization, *Machine Learning*, 1–22.
- Freund, Y. & Schapire, R. (1996). Experimental results on a new boosting algorithm, *Machine Learning: Proceedings of the Thirteenth International Conference*, 148–156.
- Grove, A. & Schervish, D. (1998). Boosting in the limit: Maximizing the margin of learned ensemble. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence (AAAI-98)*.
- Ho, T. K. (1998). The random subspace method for combining decision forests. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 20(8), 832–844.
- Kleinberg, E. (2000). On the algorithmic implementation of stochastic discrimination. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 22(5), 473–490.
- Schapire, R., Freund, Y., Barlett, P., & Lee, W. (1998). Boosting the margin: A new explanation for the effectiveness of voting method. *Annals of Statistics*, 26(5), 1651–1686.
- Tibshirani, R. (1996). Bias, variance, and prediction error for classifier ensemble. Technical Report, Statistics Department, University of Toronto.
- Wolpert, D. H. & Macread, W. G. (1997). An efficient method to estimate Bagging's generalization error (in press, Machine Learning).

Received November 30, 1999

Revised April 11, 2001

Accepted April 11, 2001

Final manuscript April 11, 2001