Machine Learning, 45, 5–32, 2001

Random Forests

LEO BREIMAN
Statistics Department, University of California, Berkeley, CA 94720

Editor: Robert E. Schapire

Abstract. Random fore t are a combination of tree predictor ch that each tree depend on the al e of a random ector ampled independentl and ith the ame di trib tion for all tree in the fore t. The generali ation error for fore t con erge a. . to a limit a the n mber of tree in the fore t become large. The generali ation error of a fore t of tree cla ifier depend on the trength of the indi id al tree in the fore t and the correlation bet een them. U ing a random election of feat re to plit each node ield error rate that compare fa orabl to Adaboo t (Y. Fre nd & R. Schapire, *Machine Learning*: *Proceedings of the Thirteenth International conference*, ∗∗∗, 148–156), b t are more rob t ith re pect to noi e. Internal e timate monitor error, trength, and correlation and the e are ed to ho the re pon e to increa ing the n mber of feat re ed in the plitting. Internal e timate are al o ed to mea re ariable importance. The e idea are al o applicable to regre ion.

Keywords: cla ification, regre ion, en emble

1. Random forests

1.1. Introduction

Significant impro ement in cla ification acc rac ha e re lted from gro ing an en emble of tree and letting them ote for the mo t pop lar cla . In order to gro the e en emble , often random ector are generated that go ern the gro th of each tree in the en emble. An earl e ample i bagging (Breiman, 1996), here to gro each tree a random election (itho t replacement) i made from the e ample in the training et.

Another e ample i random plit election (Dietterich, 1998) here at each node the plit i elected at random from among the K be t plit . Breiman (1999) generate ne training et b randomi ing the o tp t in the original training et. Another approach i to elect the training et from a random et of eight on the e ample in the training et. Ho (1998) ha ritten a n mber of paper on the random b pace method hich doe a random election of a b et of feat re to e to gro each tree.

In an important paper on ritten character recognition, Amit and Geman (1997) define a large n mber of geometric feat re and earch o er a random election of the e for the be t plit at each node. Thi latter paper ha been in ential in m thinking.

The common element in all of the e proced re i that for the kth tree, a random ector Θ_k i generated, independent of the pa t random ector $\Theta_1, \ldots, \Theta_{k-1}$ b t ith the ame di trib tion; and a tree i gro n ing the training et and Θ_k, re lting in a cla ifier $h(\mathbf{x}, \Theta_k)$ here $\mathbf{x}$ i an inp t ector. For in tance, in bagging the random ector Θ i generated as the counts in N boxes resulting from N darts thrown at random at the boxes, where N is number of examples in the training set. In random split selection Θ consists of a number of independent random integers between 1 and K. The nature and dimensionality of Θ depends on its use in tree construction.

After a large number of trees is generated, they vote for the most popular class. We call these procedures **random forests**.

Definition 1.1. A random forest is a classifier consisting of a collection of tree-structured classifiers $\{h(\mathbf{x}, \Theta_k),\ k = 1, \ldots\}$ where the $\{\Theta_k\}$ are independent identically distributed random vectors and each tree casts a unit vote for the most popular class at input $\mathbf{x}$.

1.2. *Outline of paper*

Section 2 gives some theoretical background for random forests. Use of the Strong Law of Large Numbers shows that they always converge so that overfitting is not a problem. We give a simplified and extended version of the Amit and Geman (1997) analysis to show that the accuracy of a random forest depends on the strength of the individual tree classifiers and a measure of the dependence between them (see Section 2 for definitions).

Section 3 introduces forests using the random selection of features at each node to determine the split. An important question is how many features to select at each node. For guidance, internal estimates of the generalization error, classifier strength and dependence are computed. These are called out-of-bag estimates and are reviewed in Section 4. Sections 5 and 6 give empirical results for two different forms of random features. The first uses random selection from the original inputs; the second uses random linear combinations of inputs. The results compare favorably to Adaboost.

The results turn out to be insensitive to the number of features selected to split each node. Usually, selecting one or two features gives near optimum results. To explore this and relate it to strength and correlation, an empirical study is carried out in Section 7.

Adaboost has no random elements and grows an ensemble of trees by successive reweightings of the training set where the current weights depend on the past history of the ensemble formation. But just as a deterministic random number generator can give a good imitation of randomness, my belief is that in its later stages Adaboost is emulating a random forest. Evidence for this conjecture is given in Section 8.

Important recent problems, i.e., medical diagnosis and document retrieval, often have the property that there are many input variables, often in the hundreds or thousands, with 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. But combining trees grown using random features can produce improved accuracy. In Section 9 we experiment on a simulated data set with 1,000 input variables, 1,000 examples in the training set and a 4,000 example test set. Accuracy comparable to the Bayes rate is achieved.

In many applications, understanding of the mechanism of the random forest "black box" is needed. Section 10 makes a start on this by computing internal estimates of variable importance and binding these together by reuse runs.

Section 11 looks at random forests for regression. A bound for the mean squared generalization error is derived that shows that the decrease in error from the individual trees in the forest depends on the correlation between residuals 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 $h_1(\mathbf{x}), h_2(\mathbf{x}), \ldots, h_K(\mathbf{x})$, and with the training set drawn at random from the distribution of the random vector Y, $\mathbf{X}$, define the margin function as

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

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 exceeds 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 subscripts $\mathbf{X}, Y$ indicate that the probability is 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 structure that:

Theorem 1.2. *As the number of trees increases, for almost surely all sequences* $\Theta_{1,\ldots} PE^*$ *converges to*

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

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 these two gives the foundation for understanding the workings of random forests. We build on the analysis in Amit and Geman (1997).

Definition 2.1. The margin f nc ion for a random fore i

$$mr(\mathbf{X}, Y) = P_\Theta(h(\mathbf{X}, \Theta) = Y) - \mathop{\mathrm{ma}}_{j \neq Y} P_\Theta(h(\mathbf{X}, \Theta) = j) \tag{2}$$

and he reng h of he e of cla i er $\{h(\mathbf{x}, \Theta)\}$ i

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

A ming $s \geq 0$, Cheb che ' ineq ali gi e

$$PE^* \leq \ \mathrm{ar}(mr)/s^2 \tag{4}$$

A more re ealing e pre ion for he ariance of mr i deri ed in he follo ing: Le

$$\hat{j}(\mathbf{X}, Y) = \arg \mathop{\mathrm{ma}}_{j \neq Y} P_\Theta(h(\mathbf{X}, \Theta) = j)$$

o

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

Definition 2.2. The ra margin f nc ion i

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

Th , $mr(\mathbf{X}, Y)$ i he e pec a ion of $rmg(\Theta, \mathbf{X}, Y)$ i h re pec o Θ. For an f nc ion f he iden i

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

hold here Θ, Θ' are independen i h he ame di rib ion, impl ing ha

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

U ing (5) gi e

$$\begin{aligned} \mathrm{ar}(mr) &= E_{\Theta,\Theta'}(\mathrm{co}\ _{\mathbf{X},Y} rmg(\Theta, \mathbf{X}, Y) rmg(\Theta', \mathbf{X}, Y)) \\ &= E_{\Theta,\Theta'}(\rho(\Theta, \Theta') sd(\Theta) sd(\Theta')) \end{aligned} \tag{6}$$

here $\rho(\Theta, \Theta')$ i he correla ion be een $rmg(\Theta, \mathbf{X}, Y)$ and $rmg(\Theta', \mathbf{X}, Y)$ holding Θ, Θ' ed and $sd(\Theta)$ i he andard de ia ion of $rmg(\Theta, \mathbf{X}, Y)$ holding Θ ed. Then,

$$\begin{aligned} \mathrm{ar}(mr) &= \bar{\rho}(E_\Theta sd(\Theta))^2 \\ &\leq \bar{\rho} E_\Theta \ \mathrm{ar}(\Theta) \end{aligned} \tag{7}$$

here $\bar{\rho}$ i the mean al e of the correlation; that i ,

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

Write

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

P tting (4), (7), and (8) together ield :

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

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

Altho gh the bo nd i likel to be loo e, it f lfill the ame   gge ti e f nction for random fore t a VC-t pe bo nd do for other t pe of cla ifier . It ho   that the t o ingredient in ol ed in the generali ation error for random fore t are the trength of the indi id al cla ifier in the fore t, and the correlation bet een them in term of the ra margin f nction . The c/ 2 ratio i the correlation di ided b the q are of the trength. In nder tanding the f nctioning of random fore t , thi ratio ill be a helpf l g ide the maller it i , the better.

Definition 2.4. The c/ 2 ratio for a random fore t i defined a

$$c/s2 = \bar{\rho}/s^2.$$

There are implification in the t o cla   i ation. The margin f nction i

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

The req irement that the trength i po iti e (ee (4)) become   imilar to the familiar   eak learning condition $E_{\mathbf{X},Y} P_{\Theta}(h(\mathbf{X}, \Theta) = Y) > .5$. The ra margin f nction i $2I(h(\mathbf{X}, \Theta) = Y) - 1$ and the correlation $\bar{\rho}$ i bet een $I(h(\mathbf{X}, \Theta) = Y)$ and $I(h(\mathbf{X}, \Theta') = Y)$. In partic lar, if the al e for Y are taken to be $+1$ and -1, then

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

 o that $\bar{\rho}$ i the correlation bet een t o different member of the fore t a eraged o er the Θ, Θ' di trib tion.

For more than t o cla e , the mea re of trength defined in (3) depend on the fore t a   ell a the indi id al tree   ince it i the fore t that determine $\hat{j}(\mathbf{X}, Y)$. Another approach is possible. Write

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

Define

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

to be the strength of the set of classifiers $\{h(\mathbf{x},\Theta)\}$ relative to class j. Note that this definition of strength does not depend on the forest. Using Chebyshev's inequality, assuming all $s_{j>0}$ leads to

$$PE^* \leq \sum_j \mathrm{var}\left(P_\Theta(h(\mathbf{X},\Theta)=Y) - P_\Theta(h(\mathbf{X},\Theta)=j)\right)/s_j^2 \tag{9}$$

and using identities similar to those used in deriving (7), the variances in (9) can be expressed in terms of average correlations. I did not use estimates of the quantities in (9) in our empirical study but think they 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 split selection (Dieterrich, 1998) does better than bagging. Breiman's introduction of random noise into the outputs (Breiman, 1998c) also does better. But none of these these three forests do as well as Adaboost (Freund & Schapire, 1996) or other algorithms that work by adaptive reweighting (arcing) of the training set (see Breiman, 1998b; Dieterrich, 1998; Bauer & Kohavi, 1999).

To improve accuracy, the randomness injected has to minimize the correlation $\bar{\rho}$ while maintaining strength. The forests studied here consist of using randomly selected inputs or combinations of inputs at each node to grow each tree. The resulting forests give accuracy that compare favorably with Adaboost. This class of procedures has desirable characteristics:

i Its accuracy is 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 estimates of error, strength, correlation and variable importance.
v It's simple and easily parallelized.

Amit and Geman (1997) grew shallow trees for handwritten character recognition using random selection from a large number of geometrically defined features to define the split at each node. Although my implementation is different and not problem specific, it was their work that provided the start for my ideas.

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

In my 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 trees 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. These estimates are done out-of-bag, which is explained as follows.

Assume a method for constructing a classifier from any training set. Given a specific training set T, form bootstrap training sets T_k, construct classifiers $h(\mathbf{x}, T_k)$ and let these vote to form the bagged predictor. For each $y, \mathbf{x}$ in the training set, aggregate the votes only over those classifiers for which T_k does not containing $y, \mathbf{x}$. 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 Macready (1996), proposed using out-of-bag estimates as an ingredient in estimates of generalization error. Wolpert and Macready worked on regression type problems and proposed a number of methods for estimating the generalization error of bagged predictors. Tibshirani used out-of-bag estimates of variance to estimate generalization error for arbitrary classifiers. The study of error estimates for bagged classifiers in Breiman (1996b), gives empirical evidence to show that the out-of-bag estimate is as accurate as using a test set of the same size as the training set. Therefore, using the out-of-bag error estimate removes the need for a set aside 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 as many classifiers as in the ongoing main combination. Since the error rate decreases as the number of combinations increases, the out-of-bag estimates will tend to overestimate the current error rate. To get unbiased out-of-bag estimates, it is necessary to run past the point where the test set error converges. But unlike cross-validation, where bias is present but its extent unknown, the out-of-bag estimates are unbiased.

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

4. Random forests using random input selection

The simplest random forest with random features is formed by selecting at random, at each node, a small group of input variables to split on. Grow the tree using CART methodology to maximum size and do not prune. Denote this procedure by Forest-RI. The size F of the group is fixed. Two values of F were tried. The first used only one randomly selected

Table 1. Data set summary.

Data set	Train size	Test size	Inputs	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
Votes	435	–	16	2
Liver	345	–	6	2
Letters	15000	5000	16	26
Sat-images	4435	2000	36	6
Zip-code	7291	2007	256	10
Waveform	300	3000	21	3
Twonorm	300	3000	20	2
Threenorm	300	3000	20	2
Ringnorm	300	3000	20	2

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

My experiments use 13 smaller sized data sets from the UCI repository, 3 larger sets separated into training and test sets and 4 synthetic data sets. The first 10 sets were selected because I had used them in past research. Table 1 gives a brief summary.

On each of the 13 smaller sized data sets, the following procedure was used: a random 10% of the data was set aside. On the remaining data, random forest was run twice, growing and combining 100 trees, once with $F = 1$, and the second time with $F = \text{int}(\log_2 M + 1)$. The set aside 10% was then put down each forest to get a test set error for both. The test set error selected corresponded to the lower value of the out-of-bag estimate in the two runs. This was repeated 100 times and the test set errors averaged. The same procedure was followed for the Adaboost runs which are based on combining 50 trees.

The use of 100 trees in random forests and 50 for Adaboost comes from two sources. The out-of-bag estimates are based on only about a third as many trees as are in the forest. To get reliable estimates I opted for 100 trees. The second consideration is that growing random forests is many times 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 set errors (%).

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
Liver	30.7	25.1	24.7	40.6
Letters	3.4	3.5	4.7	19.8
Sat-images	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
Twonorm	4.9	3.9	3.9	24.7
Threenorm	18.8	17.5	17.5	38.4
Ringnorm	6.9	4.9	4.9	25.7

In the runs on the larger data sets, the random forest results for the first two data sets were based on combining 100 trees; the zip-code procedure combined 200. For Adaboost, 50 trees were combined for the first three data sets and 100 for zip-code. The synthetic data was described in Breiman (1996) and also used in Schapire et al. (1997). There were 50 runs. In each run, a new training set of size 300 and test set of size 3000 were generated. In random forests 100 trees were combined in each run—50 in Adaboost. The results of these runs are given in Table 2.

The second column are the results selected from the two group sizes by means of lowest out-of-bag error. The third column is the test set error using just one random feature to grow the trees. The fourth column contains the out-of-bag estimates of the generalization error of the individual trees in the forest computed for the best setting (single or selection). This estimate is computed by using the left-out instances as a test set in each tree grown and averaging the result over all trees in the forest.

The error rates using random input selection compare favorably with Adaboost. The comparison might be even more favorable if the search is over more values of F instead of the preset two. But the procedure is not overly sensitive to the value of F. The average absolute difference between the error rate using $F = 1$ and the higher value of F is less than 1%. The difference is most pronounced on the three large data sets.

Table 3. Test set errors (%).

		Forest-RC		
Data set	Adaboost	Selection	Two features	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
Votes	4.8	4.1	4.0	8.6
Liver	30.7	27.3	27.2	40.3
Letters	3.4	3.4	4.1	23.8
Sat-images	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
Twonorm	4.9	3.8	3.9	20.9
Threenorm	18.8	16.8	16.9	34.8
Ringnorm	6.9	4.8	4.6	24.6

8.5%, on the letters data to 3.0%, but the zip-code test set error did not decrease. Acting on an informed hunch, I tried Forest-RI with $F = 25$. The zip-code test set error dropped to 5.8%. These are the lowest test set errors so far achieved on these three data sets by tree ensembles.

5.1. Categorical variables

Some or all of the input variables may be categoricals and since we want to define additive combinations of variables, we need to define how categoricals will be treated so they can be combined with numerical variables. 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 define a substitute variable that is one when the categorical value of the variable is in the subset and zero outside.

Since a categorical variable with I values can be coded into $I - 1$ dummy $0-1$ variables, we make the variable $I - 1$ times as probable as a numeric variable to be selected in node splitting. When many of the variables are categorical, using a low value of F results in low correlation, but also low strength. F must be increased to about two-three times $\text{int}(\log_2 M + 1)$ to get enough strength to provide good test set accuracy.

For instance, on the DNA data set having 60 four-valued categorical values, 2,000 examples in the training set and 1,186 in the test set, using Forest-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 Forest-RI with $F = 12$ gives a test set error of 5.3% (5.8% for Adaboost). Using Forest-RC with combinations of 3 and $F = 8$ gives an error of 5.5%.

One advantage of this approach is that it gets around the difficulty of what to do with categoricals that have many values. 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 any categorical variable involves only the selection 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 sensitivity in the generalization error to the group size F. To conduct an empirical study of the effects of strength and correlation in a variety of data sets, out-of-bag estimates of the strength and correlation, as described in Section 3.1, were used.

We begin by running Forest-RI on the sonar data (60 inputs, 208 examples) using from 1 to 50 inputs. In each iteration, 10% of the data was 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 values of test set error, strength, correlation, etc. recorded. Eighty iterations were done, each time removing a random 10% of the data for use as 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 values of strength and correlation vs. F. The result is fascinating. Past about $F = 4$ 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 stable. Both show the same behavior—a small drop from $F = 1$ out to F about 4–8, and then a general, gradual increase. This increase in error tallies with the beginning of the constancy region for the strength.

Figure 2 has plots for similar runs on the breast data set where features consisting of random combinations of three inputs are used. The number of these features was varied from 1 to 25. Again, the correlation shows a slow rise, while the strength stays virtually constant, so that the minimum error is at $F = 1$. The surprise in these two figures is the relative constancy of the strength. Since the correlations are slowly but steadily increasing, the lowest error occurs when 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 satellite data set. The number of features, each consisting of a random sum of two 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

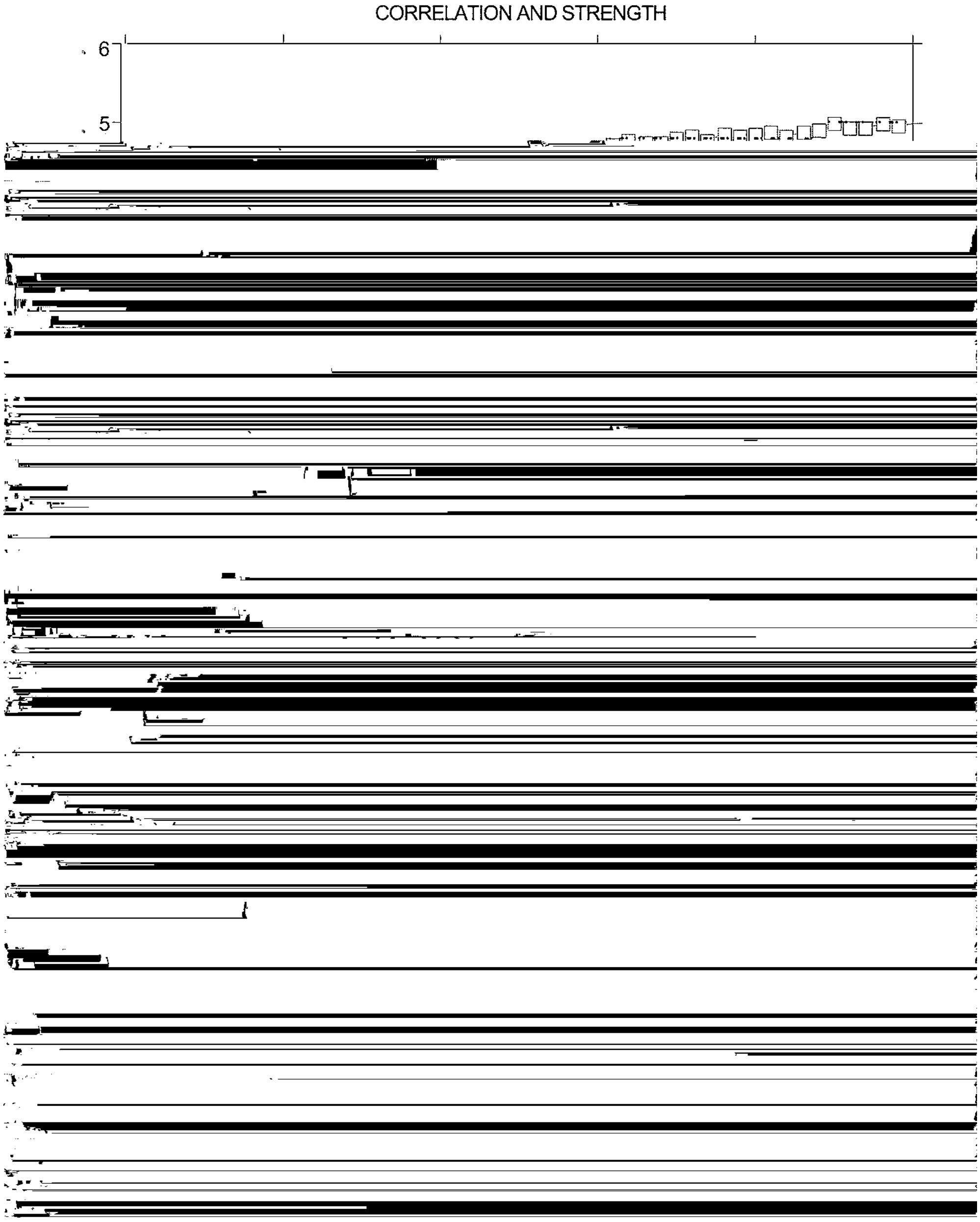

Figure 1. Effec of n mber of inp on onar da a.

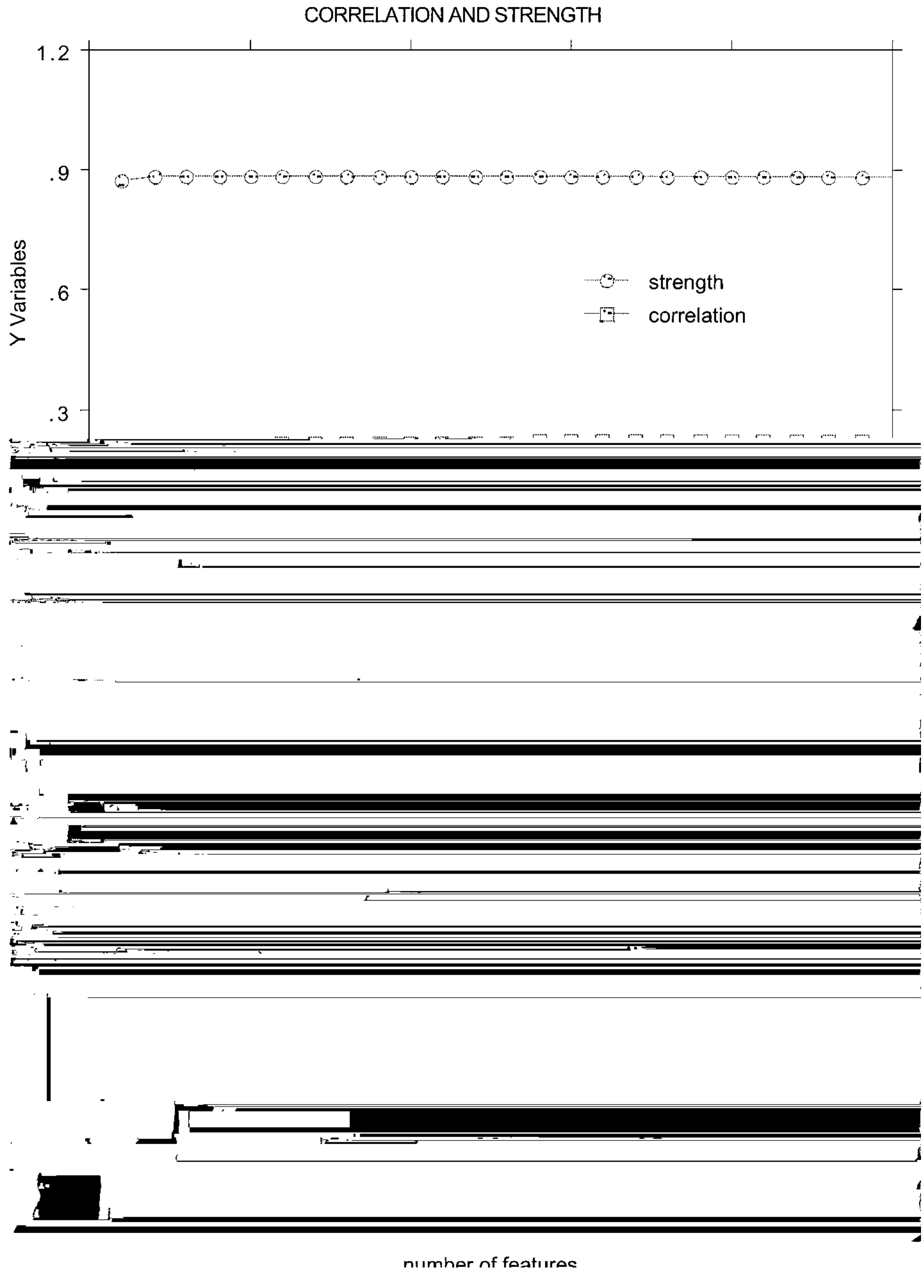

Figure 2. Effect of the number of features on the breast data set.

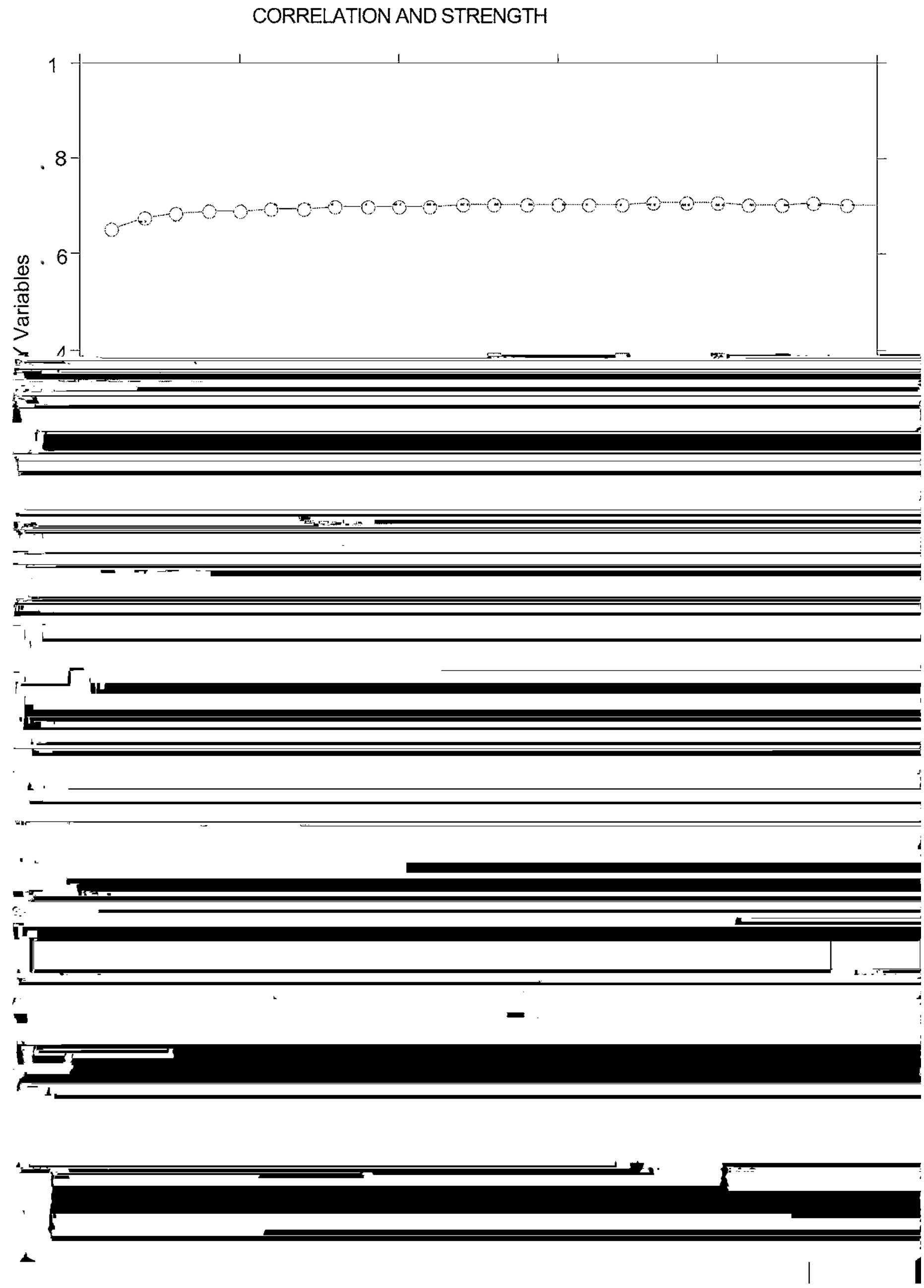

Figure 3. Effect of number of features on satellite data.

data sets. Both the correlation and strength show a small but steady increase. The error rates show a slight decrease. We conjecture that with larger and more complex data sets, the strength continues to increase longer before it plateaus out.

Our results indicate that better (lower generalization error) random forests have lower correlation between classifiers and higher strength. The randomness used in tree construction has to aim for low correlation $\bar{\rho}$ while maintaining reasonable strength. This conclusion has been hinted at in previous work. Dietterich (1998) has measures of 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 decouple the next classifier from the current one. Amit et al. (1999) give an analysis to show that 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 use 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 sum-one weights on the training set is defined. Denote these weights by $\mathbf{w}(1), \mathbf{w}(2), \ldots, \mathbf{w}(K)$. Corresponding to these weights are probabilities $p(1), p(2), \ldots, p(K)$ whose sum is one. Draw from the integers $1, \ldots, K$ according to these 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 selects the weights on the training set for input to the next classifier based on the misclassifications in the previous classifiers. In our experiment, random forests were produced as follows: Adaboost was run 75 times on a data set producing sets of non-negative sum-one weights $\mathbf{w}(1), \mathbf{w}(2), \ldots, \mathbf{w}(50)$ (the first 25 were discarded). The probability for the kth set of weights is set proportional to $Q(\mathbf{w}_k) = \log[(1 - \mathit{error}(k))/\mathit{error}(k)]$ where $\mathit{error}(k)$ is the $\mathbf{w}(k)$ weighted training set error of the kth classifier. Then the forest is run 250 times.

This was repeated 100 times on a few data sets, each time leaving out 10% as a test set and then averaging the test set errors. 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 kth classifier by $h(\mathbf{x}, \mathbf{w}_k)$. The vote of the kth classifier is weighted by $Q(w_k)$ so the normalized vote for class j at $\mathbf{x}$ equals

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

For any 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 selected at random from the distribution $Q\pi$.

Its truth would also explain why Adaboost does not overfit as more trees are added to the ensemble—an experimental fact that has been puzzling. There is some experimental evidence that Adaboost may overfit if run thousands of times (Grove & Schuurmans, 1998), but these runs were done using a very simple base classifier and may not carry over to the use of trees as the base classifiers. My experience running Adaboost out to 1,0000 trees on a number of data sets is that the test set error converges to an asymptotic value.

The truth of this conjecture does not solve the problem of how Adaboost selects the favorable distributions on the weight space 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 vectors does not depend on the training set.

8. The effects of output noise

Dietterich (1998) showed that when a fraction of the output labels in the training set are randomly altered, the accuracy of Adaboost degenerates, while bagging and random split selection are more immune to the noise. Since some noise in the outputs is often present, robustness with respect to noise is a desirable property. Following Dietterich (1998) the following experiment was done which changed about one in twenty class labels (injecting 5% noise).

For each data set in the experiment, 10% at random is split off as a test set. Two runs are made on the remaining training set. The first run is on the training set as is. The second run is on a noisy version of the training set. The noisy version is gotten by changing, at random, 5% of the class labels into an alternate class label chosen uniformly from the other labels.

This is repeated 50 times using Adaboost (deterministic version), Forest-RI and Forest-RC. The test set results are averaged over the 50 repetitions and the percent increase due to the noise computed. In both random forests, we used the number of features giving the lowest test set error in the Section 5 and 6 experiments. Because of the lengths of the runs, only the 9 smallest data sets are used. Table 4 gives the increases in error rates due to the noise.

Table 4. Increases in error rates due to noise (%).

Data set	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
Votes	48.9	6.3	4.6
Liver	10.3	−.2	4.8

Adaboo t deteriorate markedl ith 5% noi e, hile the random fore t proced re generall ho mall change . The effect on Adaboo t i c rio l data et dependent, ith the t o m lticla data et , gla and ecoli, along ith diabete , lea t effected b the noi e. The Adaboo t algorithm iterati el increa e the eight on the in tance mo t recentl mi cla ified. In tance ha ing incorrect cla label ill per i t in being mi cla ified. Then, Adaboo t ill concentrate increa ing eight on the e noi in tance and become arped. The random fore t proced re do not concentrate eight on an b et of the in tance and the noi e effect i maller.

9. Data with many weak inputs

Data et ith man eak inp t are becoming more common, i.e. in medical diagno i , doc ment retrie al, etc. The common characteri tic i no ingle inp t or mall gro p of inp t can di ting i h bet een the cla e . Thi t pe of data i difficult for the al cla ifier , ne ral net and tree .

To ee if there i a po ibilit that Fore t-RI method can ork, the follo ing 10 cla , 1,000 binar inp t data, a generated: (rnd i a niform random n mber, elected ane each time it appear)

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

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

do n=1,N
j=nint(10*rnd)
do m=1,1000
if (rnd<p(j,m) )then
  (m,n)=1
el e
  (m,n)=0
end if
  (n)=j       !  (n) i  the cla   label of the nth e ample
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 with the mth variable having probability p(j, m) of being one.

For the training set, N = 1,000. A 4,000 example test set was also 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 these locations is about 2,000, so there is significant overlap. Assuming one knows all of the {p (j, k)}, the Bayes error rate for the particular {p (j, m)} computed in our run is 1.0%.

Since the inputs are independent of each other, the Naive 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 Naive Bayes, since it would be easy to create a dependence between the inputs which would increase the Naive Bayes error rate. I stayed with this example because the Bayes error rate and the Naive Bayes error rate are easy to compute.

I started with a run of Forest-RI 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/s^2 ratio of 2.5. Even though the strength was low, the almost zero correlation meant that we were adding small increments of accuracy as the iterations proceeded.

Clearly, what was desired was an increase in strength while keeping the correlation low. Forest-RI was run again using $F = \text{int}(\log_2 M + 1) = 10$. 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 $c/s^2 = .91$. Going with the trend, Forest-RI was run with $F = 25$ and stopped after 2,000 iterations. The test set error was 2.8%. Strength was .28, correlation .065 and $c/s^2 = .83$.

It's interesting that Forest-RI 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%. Forests seem to have the ability to work with very 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 interpretations of its mechanism go. In some applications, analysis of medical experiments for example, it is critical to understand the interaction of variables that is providing the predictive accuracy. A start on this problem is made by using internal out-of-bag estimates, and verification by reruns using only selected variables.

Suppose there are M input variables. After each tree is constructed, the values of the mth variable in the out-of-bag examples are randomly permuted and the out-of-bag data is run down the corresponding tree. The classification given for each $\mathbf{x}_n$ that is out of bag is saved. This is repeated for $m = 1, 2, \ldots, M$. At the end of the run, the plurality of out-of-bag class votes for $\mathbf{x}_n$ with the mth variable noised up is compared with the true class label of $\mathbf{x}_n$ to give a misclassification rate.

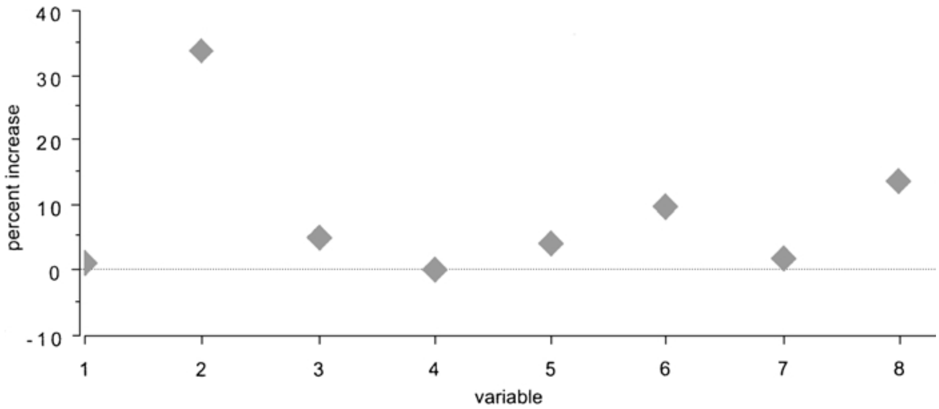

Figure 4. Measure of variable importance, diabetes data.

The output is the percent increase in misclassification rate as compared to the out-of-bag rate (with all variables intact). We get these estimates by using a single run of a forest with 1,000 trees and no test set. The procedure is illustrated by examples.

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

The second variable appears by far the most 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 to measure test set error gave an error of 29.7%, compared with 23.1% using all variables. But when variable 8 is added, the error falls only to 29.4%. When variable 6 is added to variable 2, the error falls to 26.4%.

The reason that variable 6 seems important, yet is no help once variable 2 is entered is a characteristic of how dependent variables affect prediction error in random forests. Say there are two variables x_1 and x_2 which are identical and carry significant predictive information. Because each gets picked with about the same frequency in a random forest, noising each separately will result in the same increase in error rate. But once x_1 is entered as a predictive variable, using x_2 in addition will not produce any decrease in error rate. In the diabetes 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 magnitudes of rises in error rates are fairly stable with respect to the input features used. The experiment above was repeated using combinations of three inputs with $F = 2$. The results are in Figure 5.

Another interesting example is the voting data. This has 435 examples corresponding to 435 Congressmen and 16 variables reflecting their yes-no votes on 16 issues. The class variable is Republican or Democrat. To see which issues were most important, we again ran the noising variables program generating 1,000 trees. The lowest error rate on the original data was gotten using single inputs with $F = 5$, so these parameters were used in the run. The results are in Figure 6.

Variable 4 stands out, the error triples if variable 4 is noised. We reran this data set using only variable 4. The test set error is 4.3%, about the same as if all variables were used. The

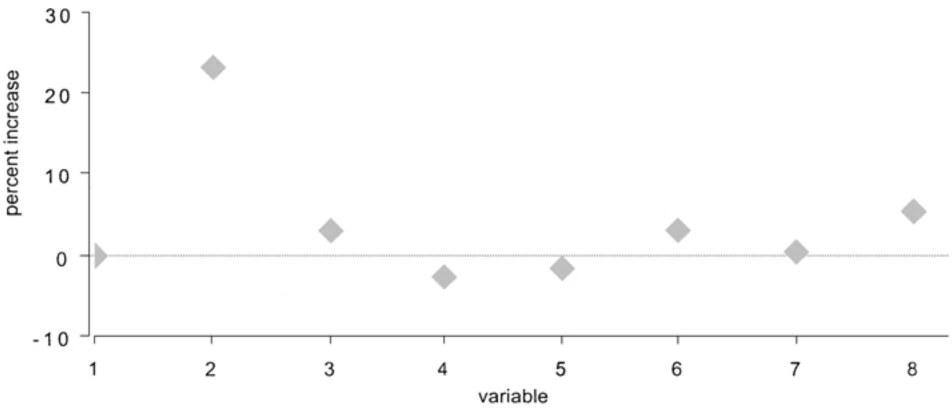

Figure 5. Mea re of ariable importance-diabete data.

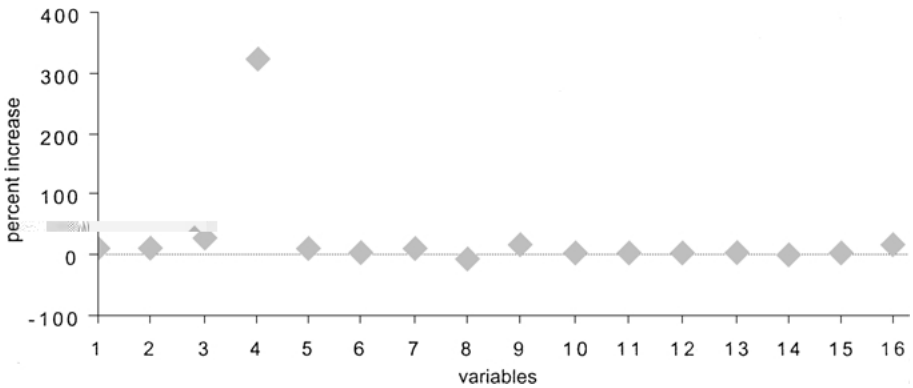

Figure 6. Mea re of ariable importance, ote data.

ote on 4 eparate Rep blican from Democrat almo t a ell a the ote on 4 combined ith the ote on all other 15 i e .

The approach gi en in thi ection i onl a beginning. More re earch ill be nece ar to nder tand ho to gi e a more complete pict re.

11. Random forests for regression

Random fore t for regre ion are formed b gro ing tree depending on a random ector Θ ch that the tree predictor $h(\mathbf{x}, \Theta)$ take on n merical al e a oppo ed to cla label . The o tp t al e are n merical and e a me that the training et i independentl dra n from the di trib tion of the random ector $Y, \mathbf{X}$. The mean- q ared generali ation error for an n merical predictor $h(\mathbf{x})$ i

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

The random fore predic or i formed b aking he a erage o er k of he ree $\{h(\mathbf{x}, \Theta_k)\}$. Similarl o he cla i ca ion ca e, he follo ing hold :

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

$$E_{\mathbf{X},Y}(Y - av_k h(\mathbf{X}, \Theta_k))^2 \rightarrow E_{\mathbf{X},Y}(Y - E_\Theta h(\mathbf{X}, \Theta))^2. \tag{12}$$

Proof: ee Appendi I. □

Deno e he righ hand ide of (12) a PE*(fore), he generali a ion error of he fore . De ne he a erage generali a ion error of a ree a :

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

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

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

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

Proof:

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

The erm on he righ in (13) i a co ariance and can be ri en a :

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

here $sd(\Theta) = \sqrt{E_{\mathbf{X},Y}(Y - h(\mathbf{X}, \Theta))^2}$. De ne he eigh ed correla ion a :

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

Then

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

Theorem (11.2) pinpoin he req iremen for acc ra e regre ion fore lo correla- ion be een re id al and lo error ree . The random fore decrea e he a erage error of he ree emplo ed b he fac or $\bar{\rho}$. The randomi a ion emplo ed need o aim a lo correla ion.

12. Empirical results in regression

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

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

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

Table 6 gives the test set mean-squared error for bagging, adaptive bagging and the random forest. These were all run using 25 features, each a random linear combination of two randomly selected inputs, to split each node, each feature a random combination of two inputs. All runs with all data sets, combined 100 trees. In all data sets, the rule don't split if the node size is <5 was enforced.

An interesting difference between regression and classification is that the correlation increases quite slowly as the number of features used increases. The major effect is the decrease in PE*(tree). Therefore, a relatively large number of features are required to reduce PE*(tree) and get near optimal test set error.

Table 5. Data set summary.

Data set	Nr. inputs	#Training	#Test
Boston Housing	12	506	10%
Ozone	8	330	10%
Servo	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 set error.

Data set	Bagging	Adapt. bag	Forest
Boston Housing	11.4	9.7	10.2
Ozone	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 sharp decreases in error, the decreases produced by forests are not as pronounced. In data sets in which adaptive bagging gives no improvements over bagging, forests produce improvements.

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 used is too small, PE*(tree) becomes too large and the error goes up. If the number used is too large, the correlation goes up and the error again increases. The in-between range is usually 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 set errors, the out-of-bag error estimates, and the OB estimates for PE*(tree) and the correlation.

As expected, the OB Error estimates are consistently high. It is low in the robot arm data, but I believe that this is an artifact caused by separate training and test sets, where the test set may have 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 set equal to the standard deviation of the

Table 7. Error and OB estimates.

Data Set	Test error	OB error	PE*(tree)	Cor.
Boston Housing	10.2	11.6	26.3	.45
Ozone	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 test set error.

Data set	With bagging	With Noise
Boston Housing	10.2	9.1
Ozone	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

outputs. Similar to the bagging experiments, tree construction was done using 25 features, each a random linear combination of two randomly selected inputs, to split each node. The results are given in Table 8.

The error rates on the first two data sets are the lowest to date. Overall, adding output noise works with random feature selection better than bagging. This is illustrative of the flexibility of the random forest setting—various combinations of randomness can be added to see what works the best.

13. Remarks and conclusions

Random forests are an effective tool in prediction. Because of the Law of Large Numbers 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 correlations gives insight into the ability of the random forest to predict. Using out-of-bag estimation makes concrete the otherwise theoretical values of strength and correlation.

For a while, the conventional thinking was that forests could not compete with arcing type algorithms in terms of accuracy. Our results dispel this belief, but lead 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) was designed to reduce bias and operates effectively in classification as well as in regression. But, like arcing, it also changes the training set as it progresses.

Forests give results competitive with boosting and adaptive bagging, yet do not progressively change the training set. Their accuracy indicates that they act to reduce bias. The mechanism for this is not obvious. Random forests may also be viewed as a Bayesian procedure. Although I doubt that this is a fruitful line of exploration, if it could explain the bias reduction, I might become more of a Bayesian.

Random inputs and random features produce good results in classification—less so in regression. The only types 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 combinations of features.

An almost obvious question is whether gains in accuracy can be gotten by combining random features with boosting. For the larger data sets, it seems that significantly lower error rates are possible. On some runs, we got errors as low as 5.1% on the zip-code data, 2.2% on the letter data and 7.9% on the satellite data. The improvement was less on the smaller data sets. More work is needed on this; but it does suggest that different injections of randomness can produce better results.

A recent paper (Breiman, 2000) shows that in distribution space for two class problems, random forests are equivalent to a kernel acting on the true margin. Arguments are given that randomness (low correlation) enforces the symmetry of the kernel while strength enhances a desirable skewness at abrupt curved boundaries. 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: It suffices to show that there is a set of probability zero C on the sequence space $\Theta_1, \Theta_2, \ldots$ such that outside of C, for all $\mathbf{x}$,

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

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-rectangles. For all $h(\Theta, \mathbf{x})$ there is only a finite number K of such unions of hyper-rectangles, denoted by $S_1, \ldots, S_K$. Define $\varphi(\Theta) = k$ if $\{\mathbf{x} : h(\Theta, \mathbf{x}) = j\} = S_k$. Let N_k be the number of times 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 Numbers,

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

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

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

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

Proof of theorem 9.1: There are a finite set of hyper-rectangles $R_1, \ldots, R_K$, such that if $\bar{y}_k$ is the average of the training set y-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 parallels that of Theorem 1.2. □

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

At the end of a combination run, let

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

Thus, $Q(\mathbf{x}, j)$ is the out-of-bag proportion of votes cast at $\mathbf{x}$ 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

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

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 set gives the strength estimate.

From Eq. (7),

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

The variance of mr is

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

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

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

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

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

where

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

After the kth classifier is constructed, $Q(\mathbf{x}, j)$ is computed, and used to compute $\hat{j}(\mathbf{x}, y)$ for every example in the training set. Then, let p_1 be the average over all $(y, \mathbf{x})$ in the training set but not in the kth bagged training set of $I(h(\mathbf{x}, \Theta_k) = y)$. Then p_2 is the similar average of $I(h(\mathbf{x}, \Theta_k) = \hat{j}(\mathbf{x}, y))$. Substitute these estimates into (A2) to get an estimate of $sd(\Theta_k)$. Average the $sd(\Theta_k)$ over all k to get the final estimate of $sd(\Theta)$.

References

Amit, Y. & Geman, D. (1997). Shape quantization and recognition with randomized trees. *Neural Computation, 9*, 1545–1588.

Amit, Y., Blanchard, G., & Wilder, K. (1999). Multiple randomized classifiers: MRCL Technical Report, Department of Statistics, University of Chicago.

Bauer, E. & Kohavi, R. (1999). An empirical comparison of voting classification algorithms. *Machine Learning, 36*(1/2), 105–139.

Breiman, L. (1996a). Bagging predictors. *Machine Learning 26*(2), 123–140.

Breiman, L. (1996b). Out-of-bag estimation, ftp.stat.berkeley.edu/pub/users/breiman/OOBestimation.ps

Breiman, L. (1998a). Arcing classifiers (discussion paper). *Annals of Statistics, 26*, 801–824.

Breiman. L. (1998b). Randomizing outputs to increase prediction accuracy. Technical Report 518, May 1, 1998, Statistics Department, UCB (in press, Machine Learning).

Breiman, L. 1999. Using adaptive bagging to debias regressions. Technical Report 547, Statistics Dept. UCB.

Breiman, L. 2000. Some infinity theory for predictor ensembles. Technical Report 579, Statistics Dept. UCB.

Dietterich, T. (1998). An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting and randomization, *Machine Learning*, 1–22.

Freund, Y. & Schapire, R. (1996). Experiments with a new boosting algorithm, *Machine Learning*: *Proceedings of the Thirteenth International Conference*, 148–156.

Grove, A. & Schuurmans, D. (1998). Boosting in the limit: Maximizing the margin of learned ensembles. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence (AAAI-98).*

Ho, T. K. (1998). The random subspace method for constructing 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., Bartlett, P., & Lee, W. (1998). Boosting the margin: A new explanation for the effectiveness of voting methods. *Annals of Statistics, 26*(5), 1651–1686.

Tibshirani, R. (1996). Bias, variance, and prediction error for classification rules. Technical Report, Statistics Department, University of Toronto.

Wolpert, D. H. & Macready, 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