2014 International Joint Conference on Neural Networks (IJCNN)
July 6-11, 2014, Beijing, China

Towards Generating Random Forests via Extremely Randomized Trees

Le Zhang, Ye Ren and P. N. Suganthan
Electrical and Electronic Engineering
Nanyang Technological University,
50 Nanyang Ave., Singapore
{lzhang027, re0003ye, epnsugan}@ntu.edu.sg

***Abstract*— The classification error of a specified classifier can be decomposed into bias and variance. Decision tree based classifier has very low bias and extremely high variance. Ensemble methods such as bagging can significantly reduce the variance of such unstable classifiers and thus return an ensemble classifier with promising generalized performance. In this paper, we compare different tree-induction strategies within a uniform ensemble framework. The results on several public datasets show that random partition (cut-point for univariate decision tree or both coefficients and cut-point for multivariate decision tree) without exhaustive search at each node of a decision tree can yield better performance with less computational complexity.**

I. Introduction

Some classifiers are said to be unstable in the sense that small perturbations in their training sets or in construction may result in large changes in constructed predictors. In other words, they usually have extremely high variance. Subset selection methods in regression, decision trees in regression and classification, and neural nets are unstable [1].

The performance of an unstable classifier can be significantly improved by ensemble methods [2], [3], [4], [5], [6], [7] . Ensemble classifiers work by means of firstly generating an ensemble of base classifiers by learning different permutated training sets and then aggregating the outputs of all classifiers to create the final prediction. This "perturb and combine" strategy can significantly reduce the variance of unstable classifiers [1].

Two approaches for constructing classifier ensembles seem to be perceived as "classic" at present, namely bagging and boosting. They have been found to be accurate, computationally feasible across various data domains, and with no clear dominance between them. Bootstrap Aggregation (Bagging) [2] takes bootstrap samples of objects and trains a classifier on each sample set. The outputs of the classifiers are combined by majority voting. Boosting [8], [9], [3] is a family of methods, the most prominent member of which is AdaBoost [3], [8]. The idea is to boost the performance of a "weak" classifier by using it within an ensemble structure. The classifiers in the ensemble is generated in a sequential manner. The current classifier will pay more attention to those data which have been "difficult" for the previous ensemble members. A set of weights is maintained across the objects in the dataset so that objects which were difficult to classify acquire more weight, forcing subsequent classifiers to focus on them.

Comparative studies can be found in [4], [5], [6], [10]. It appears that, on average, AdaBoost is the best method although Random Subspaces and Bagging have their application niches as well. Interestingly, for large ensemble sizes (in the order of thousand classifiers) the significance between the ensemble models almost disappear [4]. Moreover, boosting tend to be more sensitive to outliers and noise.

Random Forests (RF) combine the concepts of bagging and random subspace [7] to build a classification ensemble with a set of decision trees that grow using randomly selected subspaces of data [11]. Moreover, Breiman also stated that Random Forests are similar to Adaboost [11]. RF are well-known ensemble classifiers which have gained popularity in high-dimensional and ill-posed classification and regression tasks, for example on micro-arrays [12], time series [13], or spectral data [14], [15], but also for inference in applications such as image segmentation and object recognition in computer vision [16], [17]. Random forests are comparable in performance to many other non-linear learning algorithms. They often do well with little parameter tuning [18], and are able to identify relevant feature subsets even in the presence of a large number of irrelevant classifiers [19], [20], [21]. More recently, additional properties of the random forest have gained interest, for example in feature selection [21], [22], [23], [24] and the explorative analysis of sample proximities [25].

The main effect of RF and other ensemble classifiers is to reduce variance [1], [26], [27]. It is known that this variance reduction is closely related to the randomness of the algorithm [28]. The randomness comes from various means, such as the perturbation of dataset, the randomness of the algorithm, etc. In this context, we study performance of different decision tree ensembles under the well-known RF framework: we change the test function in each node of the RF to generate different RF models. We also study the relationship among the bias-variance and the randomness of the algorithm.

The rest of this paper is organized as follows: Section II reviews the related work: RF and bias-variance decomposition. Section III elaborates our proposed method. Section IV shows the detail of the experiment environment. Some results are presented and analyzed in Section V. Finally, Section VI concludes the paper.

978-1-4799-1484-5/14/$31.00 ©2014 IEEE

II. Related Work

A. Bias-Variance decomposition

Bias-variance decomposition [29], [1] is a powerful tool from statistical sampling theory for analyzing supervised learning scenarios that have quadratic loss functions. As conventionally formulated, it breaks the expected cost given a fixed target and training set size into the sum of three non-negative quantities: noise, bias and variance. Noise defines the bound on the expected cost of any learning algorithms. It can also be regarded as the expected cost of the Bayes optimal classifiers. This bias (or squared bias) measures how closely the learning algorithm's average guess (over all possible training sets of the given size) matches the target. The variance measures how much the learning algorithm's guess fluctuates from the target for the different training sets of the given size.

The bias-variance insight was originally borrowed from the field of regression with squared loss as the loss function [29]. For classification problems, the above decomposition is inappropriate because class labels are categorical, which means it is not proper to transplant the decomposition of error in regression tasks to classification. Fortunately, a number of ways to decompose error into bias and variance terms in classification tasks have been proposed [30], [31], [32], [33], [34]. Each of these definitions is able to provide some valuable insight into different aspects of a learning algorithm's performance.

We consider classification problems and the 0-1 loss function in the Kohavi and Wolpert's work [31]. Let X and Y be the input and output spaces respectively with cardinalities $|X|$ and $|Y|$ and elements x and y respectively. The target f is a conditional probability distribution $P(Y_F = y_F|x)$ where Y_F is a Y-valued random variable. Unless explicitly stated otherwise, we assume that the target is fixed. As an example, if the target is a noise free function from X to Y for any fixed x we have $P(Y_F = y_F|x) = 1$ for one value of y_F and 0 for all others. The hypothesis h generated by a learning algorithm is a similar distribution $P(Y_F = y_F|x)$ where Y_H is a Y-valued random variable. As an example, if the hypothesis is a single-valued function from X to Y, as it is for many classifiers (e.g., decision trees, nearest neighbors), then $P(Y_H = y_H|x) = 1$ for one value of y_H and 0 for all others. Hereafter, we will drop the explicitly delineated random variables from the probabilities when the context is clear. For example, $P(Y_H)$ will be used instead of $P(Y_H = y_H)$. Then for a single test point, it is easy to show:

$$E(C) = 1 - \sum_{y\in Y} P(Y_F = Y_H = y) \tag{1}$$

$$\begin{aligned} E(C) = &-\sum_{y\in Y} P(Y_F = Y_H = y) \\ &+ \sum_{y\in Y} P(Y_F = y)P(Y_H = y) \\ &+ \sum_{y\in Y} [-P(Y_H = y)P(Y_F = y) \\ &+ \frac{1}{2}P(Y_H = y)^2 + \frac{1}{2}P(Y_F = y)^2] \\ &+ [\frac{1}{2} - \frac{1}{2}\sum_{y\in Y} P(Y_H = y)^2] \\ &+ [\frac{1}{2} - \frac{1}{2}\sum_{y\in Y} P(Y_F = y)^2] \end{aligned} \tag{2}$$

Rearranging the terms, we have

$$\begin{aligned} E(C) = &\sum_{y\in Y} [P(Y_F = y)P(Y_H = y) \\ &- P(Y_F = Y_H = y)] \\ &+ \frac{1}{2}\sum_{y\in Y} [P(Y_F = y) - P(Y_H = y)]^2 \\ &+ \frac{1}{2}[1 - \sum_{y\in Y} P(Y_H = y)^2] \\ &+ \frac{1}{2}[1 - \sum_{y\in Y} P(Y_F = y)^2] \end{aligned} \tag{3}$$

Y_F and Y_H are conditionally independent given f and a test point x [31], hence the "covariance" term vanishes. So,

$$E(C) = \sum_{x} P(X)[(bias_x)^2 + \sigma_x^2 + variance_x]; \tag{4}$$

where

$$\begin{aligned} (bias_x)^2 &= \frac{1}{2}\sum_{y\in Y} [P(Y_F = y) - P(Y_H = y)]^2 \\ variance_x &= \frac{1}{2}[1 - \sum_{y\in Y} P(Y_H = y)^2] \\ \sigma_x^2 &= \frac{1}{2}[1 - \sum_{y\in Y} P(Y_H = y)^2] \end{aligned} \tag{5}$$

The $(bias_x)^2$ term measures the squared difference between the target's average output and the algorithm's average output. It is a real valued non-negative quantity and equals zero only if $P(Y_F = y|x) = P(Y_H = y|x)$ for all x and y. The variance term measures the variability (over Y_H)of $P(Y_H = y|x)$. It is a real-valued non-negative quantity and equals zero for an algorithm that always makes the same guess regardless of the training set (e.g. the Bayes optimal classifier). As the algorithm becomes more sensitive to changes in the training set, the variance increases. Moreover, given a distribution over training sets, the variance only measures the sensitivity of the learning algorithm to changes in the training set and is independent of the underlying target. The noise measures the *variance* of the target in that the definitions of variance and noise are identical except for the interchange of Y_F and Y_H. In addition, the noise is independent of the learning algorithm.

B. Random Forests

Ensemble classifiers significantly reduce the variance of the classifier and retain the most part of its bias [11], which means the bias component slightly fluctuate [28]. This fluctuation can be negligible compared with the variance reduction in most cases. Then from the bias-variance decomposition point of view, the ensemble classifier should have much better performance if the base classifier has high variance. So it is easy to understand why decision tree, neural networks naturally work well with ensemble methods [1] while other methods (such as SVM) may need much more complex algorithm to be combined [26].

RF combine the concept of bagging and random subspaces to further enlarge the variance of the base classifier. RF build a classification ensemble with a set of decision trees that grow using randomly selected subspaces of data. The RF work as follows:

· Training phase:

Given:

$X := N \times m$ is the training dataset, where N is the number of the training data, m is the dimension of each data.

$Y := N \times 1$ is the labels of the training set.

L is the ensemble size, which means the number of trees in the forests.

T_i refers to each random tree in the RF, $i = 1...L$.

m is the number of features randomly selected to split in each non-leaf node.

For $i = 1...L$:

1) Generate the training set for T_i by sampling N times from all Navailable training cases with replacement.
2) At each node the best split is calculated using the m randomly chosen features in the training set for T_i.
3) Go to Step 2 until T_i is fully grown without being pruned.

· Classification phase:

For a given sample, it is pushed down each tree in the forests and each tree in the forests will give one vote on the predicted label of this sample. In this case, the predicted label of this sample is determined as the one which has the most votes in the forests.

Each tree classifier in the RF ensemble is trained on the bootstrap set of the original training set. At each node of the tree classifier, m features from M are randomly selected. Then one feature from the m features is selected to perform a partition along this feature axis according to some impurity criteria (e.g. information-gain, gini-impurity, etc.) [35]. This kind of decision tree can also be called as univariate decision tree [36] since the test within each node is performed by only one feature.

III. Proposed method

Several research work has shown ensemble method can significantly reduce the variance of the tree classifiers while maintaining the most part of its bias [11], [26], [27]. Recently, Pierre et al. [28] showed that if stronger randomization is involved, the ensemble would work better. Their algorithm works by replacing the deterministic best-split test among m features with a random test within each node. That is, they conduct a random split with each feature among the m randomly selected features. For all experiments, the impurity criteria is only used to select one best split among those m random splits.

In this paper, we propose to use even stronger randomization strategies by extending the work of Pierre to multivariate (or oblique) [36] decision trees. At each node of tree classifier, we propose to conduct a test by $f(x) = \sum_{i=1}^{m} w_i * x_i$, where x_i are the randomly selected features and w_i are their coefficients. We propose two methods. First one works by generating m different coefficients at each node and use impurity criteria to find the optimized the cut-point $f(x)$ for each trial. The second method works by randomly generating the cut-point $f(x)$ as well as all the coefficients for each trial. For both methods, we need the impurity criteria to find the best split among the m trials. Hereafter, we name Pierre's work as "Extreme RF" (E-RF)and the first method as "Oblique RF" (O-RF) and the second method as "Extreme Oblique RF" (EO-RF).

Decision tree (regardless of univariate or multi-variate) has very low bias and extremely high variance. Since the coefficient of each feature is randomly generated, the variance of this kind of oblique decision tree will tend to increase. The rationale behind the proposed method here is that the explicit randomization of the coefficient and attribute combined with ensemble averaging should be able to reduce variance more strongly than the weaker randomization schemes. In the next section, we will evaluate the performance of the proposed method and the method of Pierre [28] and Breiman [11] with several benchmark datasets.

IV. Experients

This section compares the performance of the proposed method (O-RF, EO-RF) and the method of Pierre (E-RF) and Breiman (RF) on real-world benchmark classification datasets. The information of those datasets used in this paper is summarized in Table I.

The simulation of different algorithms on all datasets are carried out in Matlab R2010b with Intel (R) Core(TM) i5, 3.20-GHz CPU and 4-GB RAM. Actually there are 2 parameters here for all versions of Random Forests. The first one L, controls the size of the ensemble. The second one m, controls the randomization of the algorithm, which stands for the number of features randomly selected to conduct the test. For all experiments, the gini-impurity [35] is employed as the impurity criterion for all tree classifiers in each node to select the best split.

Considering the computational complexity, we set the ensemble size L to be 100 for all experiments. For all datasets,

TABLE I

SPECIFICATION OF CLASSIFICATION PROBLEMS

Datasets	Samples	Features	Classes
Banknote	1372	4	2
W-Breast	699	9	2
Iris	150	4	3
P-Relax	182	12	2
Ringnorm	7400	20	2
Twonorm	7400	20	2
Vowel	990	10	11
Waveform	5000	21	3

The"W-Breast" stands for "Breast-cancer wisconsin" and "P-Relax" represents "Planning Relax".

the input features are normalized in the range of [-1, +1] to avoid the dominance of some of the features. In the first set of experiment, we set the other parameter with the default value ($m = round(\sqrt{M})$, where M is the number of features of the training data).

The classification accuracies of each dataset are presented in Table II. For each data set and ensemble method, 10 3-fold cross validations were performed. The accuracies were averaged over all the 30 testing accuracies per method and data set. The boldface indicates the best result. Table III shows a summary of the comparisons among the methods. For each of these datasets, a paired t-test ($\alpha = 0.05$) is used to determine the significance of the differences between each method. The entry α_{ij} displays the number of times when the method of the row (i) has a better result than the method of the column(j). The number in the parentheses shows in how many of these differences have been statistically significant. The biases and variances from each dataset and each algorithm are presented in Table IV.

Note that theoretically, the prediction error of a classifier should also be decomposed into three terms (irreducible error, squared bias and variance). However, it is usually difficult to estimate irreducible error in real-world learning tasks of which the true underlying class distribution is unknown and there are generally too few instances at any given point in the instance space to reliably estimate the class distribution at that point. In the commonly used methods, the irreducible error is generally aggregated into both bias and variance or only the bias term due to the fact that irreducible error is invariant across learning algorithms for a single learning task and hence not a significant factor in comparative evaluations.

TABLE III

T-TEST OF RESULTS

Method	RF	O-RF	E-RF	EO-RF
RF	-	0(0)	1(0)	0(0)
O-RF	8(8)	-	5(3)	1(1)
E-RF	7(6)	3(2)	-	2(2)
EO-RF	8(8)	7(3)	6(5)	-

The entry α_{ij} displays the number of times when the method of the row i has a better result than the method of the column j. The number in the parentheses shows in how many of these differences have been statistically significant.

In the second set of experiments, we investigate how the randomization influence the performance of the ensemble by analyzing the effect of parameter m. As we know, the parameter m denotes the number of features randomly selected at each node. It may be chosen from the interval $[1, ...M]$, where M is the number of the features of a particular dataset. For a given problem, the smaller m is, the stronger the randomization of the tree classifier. In the extreme case, for $m = 1$, the attributes selection are most likely to be different. While for $m = M$, the choice of features is not explicitly randomized at all.

In order to check how this parameter influence the performance, we have conducted a systematic experiment for all our datasets by varying the parameter over its range. For each dataset, Fig. 4 shows the evolution of the classification accuracy and bias- variance with respect to different m for "waveform" dataset.

V. DISCUSSION ON THE RESULTS

The first set of experiments check the performance of different Random Forests with default parameter. From Table II we can see in most cases, O-RF, E-RF and EO-RF all outperforms RF, which indicates that involving stronger randomization improves the performance of the Random Forest. Fig. 1 gives a graphical overview of the results in Table II. For each dataset, each bar graph from left to right stands for the accuracy for the base classifier of RF, RF, base classifier of O-RF, O-RF, base classifier of E-RF, E-RF, base classifier of EO-RF, EO-RF respectively. Form Fig. 1 we can see that the base classifier of E-RF, which selects best splits among m randomly univariate splits within each node, performs the worst among all other base classifiers. The reason is quite straightforward. For the real-life problem, especially when the optimal decision boundary is complex, recursively and randomly drawing a threshold to split may not generate a good approximation of the decision boundary of a given problem. On the other hand, exhaustive search for a optimal threshold or use of a multivariate split can achieve a good approximation of the optimal decision boundary. However, even with a higher bias, the variance of the base classifier of E-RF is usually much larger than other base classifiers, which can be evidenced by Figs. 2 and 3. So E-RF can achieve comparable performance as O-RF and EO-RF since ensemble methods can benefit significantly from such high variance base classifiers.

We investigate two oblique RF (O-RF, EO-RF)in this study. For both methods, the binary test : $f(x) = \sum_{i=1}^{m} w_i * x_i$ (where x_i is the randomly selected features and w_i is their coefficients) is conducted at each node. The only difference lies in the choice of the threshold, which is quite similar as the difference of RF and E-RF. For O-RF, the threshold, $f(x)$ is found by exhaustive search to minimize the gini-impurity. From EO-RF, the threshold is also randomly generated. Form Table II and Fig. 1, it is obvious that optimizing this threshold cannot lead to a better performance in most cases. From Figs. 2 and 3, we can see that optimizing this threshold only improves the bias of the classifier. However, on the

TABLE II

Classification Accuracy and Standard Deviation of each Algorithm

Datasets	RF	O-RF	E-RF	EO-RF
W-Breast	93.88 ± 1.67	94.62 ± 1.50	91.07 ± 3.70	94.94 ± 1.67
	96.43 ± 01.11	97.05 ± 0.95	96.36 ± 1.04	**97.08 ± 1.02**
Banknote	97.79 ± 1.21	97.98 ± 1.26	95.65 ± 1.69	98.71 ± 0.91
	99.18 ± 0.39	99.79 ± 0.20	99.90 ± 0.18	**99.99 ± 0.16**
Iris	93.20 ± 3.48	91.40 ± 4.91	88.53 ± 6.63	90.47 ± 5.39
	94.80 ± 2.42	95.27 ± 1.98	95.07 ± 2.16	**95.33 ± 2.10**
P-Relax	58.46 ± 6.36	60.71 ± 7.05	65.11 ± 6.32	61.04 ± 6.37
	69.45 ± 4.27	71.43 ± 4.48	**72.47 ± 4.91**	71.87 ± 24.70
Ringnorm	86.94 ± 0.98	86.44 ± 0.98	77.44 ± 1.70	84.08 ± 1.09
	95.50 ± 0.51	96.79 ± 0.34	**97.88 ± 0.26**	97.28 ± 0.31
Twonorm	83.94 ± 0.89	91.84 ± 0.57	81.02 ± 0.12	91.65 ± 0.54
	96.86 ± 0.40	97.62 ± 0.30	97.20 ± 0.30	**97.65 ± 0.29**
Vowel	70.75 ± 3.98	64.58 ± 3.34	55.66 ± 04.02	62.65 ± 3.53
	92.35 ± 1.65	94.19 ± 1.51	93.65 ± 1.68	**94.75 ± 1.66**
Waveform	73.68 ± 1.30	70.19 ± 1.40	63.69 ± 2.00	67.12 ± 1.64
	84.71 ± 0.81	**85.69 ± 0.80**	85.00 ± 0.80	85.48 ± 0.71

$\mu \pm \sigma$ of each algorithm. The first line in each entry of the table stands for the performance of the base classifier of the ensemble and the second line stands for the performance of the ensemble

TABLE IV

Bias and Variance of each Algorithm

Datasets	RF	O-RF	E-RF	EO-RF
W-Breast	(3.24, 2.88)	(2.67, 2.71)	(3.62, 5.31)	(2.85, 2.21)
	(2.92, 0.65)	(2.56, 0.40)	(3.13, 0.51)	(2.49, 0.43)
Banknote	(0.76, 1.45)	(0.47, 1.55)	(0.81, 3.54)	(0.23, 1.05)
	(0.71, 0.11)	(0.11, 0.10)	(0.06, 0.04)	(0.00, 0.01)
Iris	(4.56, 2.24)	(4.55, 4.05)	(4.66, 6.81)	(4.11, 5.42)
	(4.49, 0.70)	(4.19, 0.55)	(4.49, 0.45)	(4.33, 0.34)
P-Relax	(25.52, 16.03)	(23.28, 16.00)	(21.88, 13.01)	(23.79, 15.17)
	(26.65, 3.90)	(26.75, 1.82)	(26.60, 0.92)	(26.85, 1.29)
Ringnorm	(4.37, 8.69)	(4.19, 9.38)	(7.06, 15.50)	(4.88, 11.04)
	(3.29, 1.21)	(2.57, 0.64)	(1.39, 0.74)	(2.08, 0.64)
Twonorm	(5.60, 10.46)	(2.98, 5.19)	(6.81, 12.17)	(2.97, 5.38)
	(2.12, 1.03)	(1.97, 0.41)	(1.88, 0.92)	(1.94, 0.42)
Vowel	(9.53, 19.72)	(12.05, 23.37)	(16.46, 27.88)	(12.71, 24.65)
	(3.65, 4.00)	(2.14, 3.67)	(2.30, 4.05)	(1.65, 3.60)
Waveform	(12.49, 13.83)	(13.41, 16.40)	(16.53, 19.78)	(14.55, 18.33)
	(12.22, 3.07)	(11.21, 3.10)	(10.50, 4.50)	(10.86, 3.66)

The first number in the bracket stands for the bias and the second number represents the variance. The first bracket in each entry of the table stands for the bias-variance for the base classifier and the second stands for the bias-variance for the ensemble

other hand, randomly generating the threshold is advantageous to get a higher variance. As we mentioned before, stronger randomization combined with ensemble averaging should be able to reduce variance more strongly than the weaker randomization schemes. Hence, the performance of EO-RF is better than O-RF in most cases. Moreover, randomly generating a threshold without exhaustive search can reduce the computational complexity significantly, especially for large datasets.

We also find that the variance of multivariate decision tree is smaller than the base classifier of E-RF. The reason may be that multivariate decision needs fewer nodes to approximate the optimal decision boundary than the univariate one. In other words, for a given problem, multivariate decision tree is smaller than the univariate one. The variance of the decision tree grows as the depth of the tree increases [36]. In order to confirm our conjecture, we designed another experiment to test the average tree nodes of each algorithm. The results are presented in Table V.

In oder to investigate the effect of the parameter, m, we have designed another experiment by varying this parameter over its range. The result for "waveform" dataset are presented in Fig. 4. For other datasets, the results are quite similar. From the results, we can see that for very small m (especially in the extreme case, m=1), the bias and the variance of the base classifiers are larger than those with larger m. For all values of m, the bias of the ensemble and the base classifiers are almost equal and the variance of the ensemble are quite smaller than that of the base classifiers. In general, the performances of the ensemble are quite stable when m lies in the middle of its range [18].

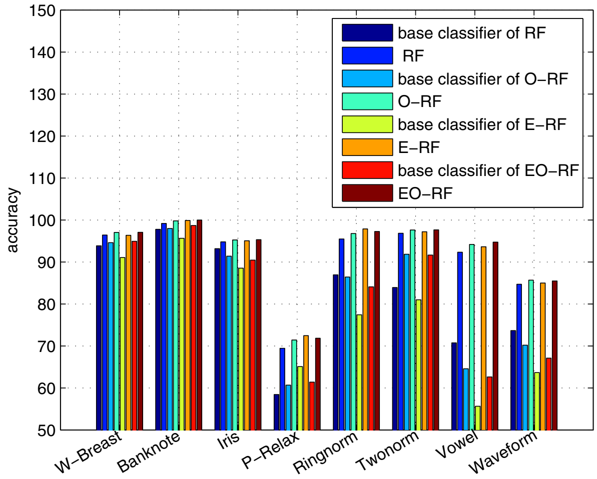

Fig. 1. The Accuracy of each Method. For each dataset, each bar graph from left to right stands for the accuracy for the base classifier of RF, RF, base classifier of O-RF, O-RF, base classifier of E-RF, E-RF, base classifier of EO-RF, EO-RF, respectively.

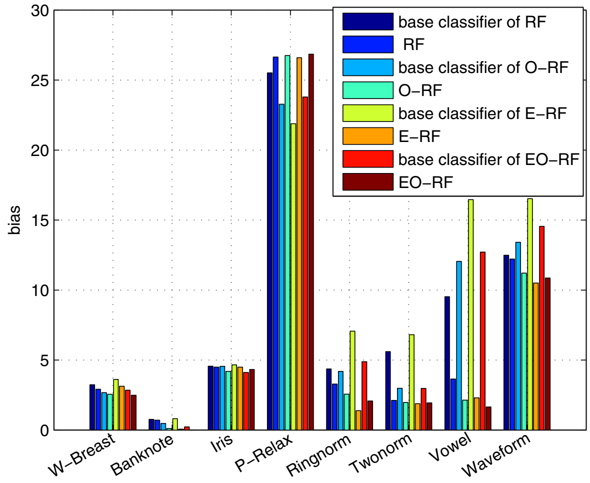

Fig. 2. The Bias of each Method

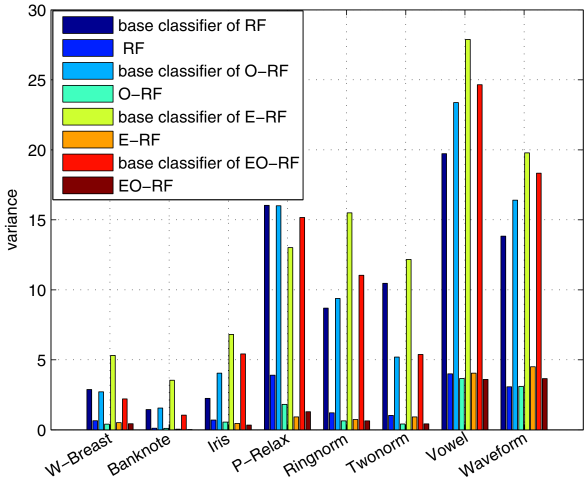

Fig. 3. The Variance of each Method

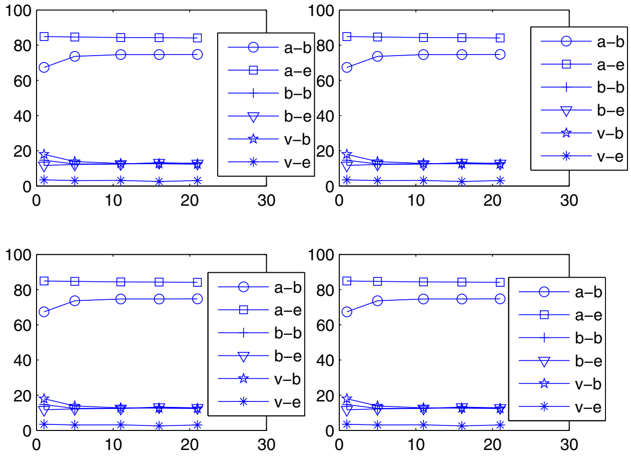

Fig. 4. Accuracy of each method with Different m Parameter for "waveform" Dataset. $a-b$, $a-e$, $b-b$, $b-e$, $v-b$, $v-e$ stands for the accuracy of the base classifier, accuracy of the ensemble, bias of the base classifier, bias of the ensemble, variance of the base classifier, variance of the ensemble respectively. The first row stands for RF and O-RF (from left to right) and the second row stands for the E-RF and EO-RF (from left to right).

TABLE V

THE AVERAGE NUMBER OF NODES FOR EACH TREE CLASSIFIER IN EACH ENSEMBLE METHOD

Datasets	RF	O-RF	E-RF	EO-RF
W-Breast	20.63	16.92	56.25	52.87
Iris	5.66	7.97	29.48	18.52
P-Relax	22.43	23.30	75.45	50.68
Ringnorm	299.93	341	1543.3	1486.7
Twonorm	367.11	226.04	1399.6	917.11
Vowel	118.56	136.31	406.28	276.27
Waveform	917.11	467.10	1278.9	1432.4

VI. CONCLUSIONS

In this paper, we have studied the performance of different decision tree ensembles under the well-known Random Forests framework. We have changed the test function in each node of the Random Forests to generate different RF models. We have also studied the relationship among the bias- variance and the randomness of the algorithm. A number of observations has been made from the experiments as follows:

1) The classification error of a specific classifier can be decomposed into bias and variance. The bias of the ensemble are almost equal to the bias of its base classifier. For decision tree, the variance of the ensemble can be significantly reduced.
2) In most cases, multivariate decision tree needs fewer nodes (or smaller tree size) to approximate the decision boundary than the univariate decision tree.
3) Optimizing the threshold of each binary test at each node can reduce both the bias and variance, but it is not a good startegy for ensemble methods.
4) Involving stronger randomization by randomly generating the threshold (or cut-point) of the split in RF can yield much larger variance and slightly larger bias. With significant variance reduction of ensemble, those randomization method can improve RF.
5) Besides saving computational time, involving stronger randomization in multivariate decision tree by randomly generating the coefficient of each feature to conduct a linear combination of features ($f(x) = \sum_{i=1}^{m} w_i * x_i$, where x_i are the randomly selected features and w_i are their coefficients) for the test at each node can yield higher variance and comparable bias. With significant variance reduction by the ensemble, randomization methods can improve the Random Forests. Moreover, randomly generating the threshold $f(x)$ can further enlarge the variance with slightly larger bias. Hence, it is unwise to optimize the threshold $f(x)$ by exhaustive search.
6) The parameter m (which stands for the number of features randomly selected at each node) controls the randomization of the algorithm. For the base classifiers, larger m leads to better performance with lower bias and lower variance. For the ensemble, the performance is quite stable when m lies in the middle of its range.

REFERENCES

[1] L. Breiman, “Bias, variance, and arcing classifiers,” 1996.

[2] ——, “Bagging predictors,” *Machine learning*, vol. 24, no. 2, pp. 123–140, 1996.

[3] Y. Freund, R. E. Schapire *et al.*, “Experiments with a new boosting algorithm,” in *ICML*, vol. 96, 1996, pp. 148–156.

[4] R. E. Banfield, L. O. Hall, K. W. Bowyer, D. Bhadoria, W. P. Kegelmeyer, and S. Eschrich, “A comparison of ensemble creation techniques,” in *Multiple classifier systems*. Springer, 2004, pp. 223–232.

[5] E. Bauer and R. Kohavi, “An empirical comparison of voting classification algorithms: Bagging, boosting, and variants,” *Machine learning*, vol. 36, no. 1-2, pp. 105–139, 1999.

[6] T. G. Dietterich, “Ensemble methods in machine learning,” in *Multiple classifier systems*. Springer, 2000, pp. 1–15.

[7] T. K. Ho, “The random subspace method for constructing decision forests,” *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 20, no. 8, pp. 832–844, 1998.

[8] Y. Freund and R. E. Schapire, “A desicion-theoretic generalization of on-line learning and an application to boosting,” in *Computational learning theory*. Springer, 1995, pp. 23–37.

[9] R. E. Schapire and Y. Freund, “Boosting the margin: a new explanation for the effectiveness of voting methods,” *The Annals of Statistics*, vol. 26, pp. 322–330, 1998.

[10] T. K. Ho, “A data complexity analysis of comparative advantages of decision forest constructors,” *Pattern Analysis & Applications*, vol. 5, no. 2, pp. 102–112, 2002.

[11] L. Breiman, “Random forests,” *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[12] H. Jiang, Y. Deng, H.-S. Chen, L. Tao, Q. Sha, J. Chen, C.-J. Tsai, and S. Zhang, “Joint analysis of two microarray gene-expression data sets to select lung adenocarcinoma marker genes,” *BMC bioinformatics*, vol. 5, no. 1, p. 81, 2004.

[13] K.-Q. Shen, C.-J. Ong, X.-P. Li, Z. Hui, and E. P. Wilder-Smith, “A feature selection method for multilevel mental fatigue eeg classification,” *Biomedical Engineering, IEEE Transactions on*, vol. 54, no. 7, pp. 1231–1237, 2007.

[14] V. Svetnik, A. Liaw, C. Tong, J. C. Culberson, R. P. Sheridan, and B. P. Feuston, “Random forest: a classification and regression tool for compound classification and qsar modeling,” *Journal of chemical information and computer sciences*, vol. 43, no. 6, pp. 1947–1958, 2003.

[15] M. Pal, “Random forest classifier for remote sensing classification,” *International Journal of Remote Sensing*, vol. 26, no. 1, pp. 217–222, 2005.

[16] A. Criminisi, J. Shotton, and S. Bucciarelli, “Decision forests with long-range spatial context for organ localization in ct volumes,” in *MICCAI Workshop on Probabilistic Models for Medical Image Analysis*, 2009.

[17] B. Yao, A. Khosla, and L. Fei-Fei, “Combining randomization and discrimination for fine-grained image categorization,” in *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*. IEEE, 2011, pp. 1577–1584.

[18] T. Hastie, R. Tibshirani, and J. J. H. Friedman, *The elements of statistical learning*. Springer New York, 2001, vol. 1.

[19] G. Biau, L. Devroye, and G. Lugosi, “Consistency of random forests and other averaging classifiers,” *The Journal of Machine Learning Research*, vol. 9, pp. 2015–2033, 2008.

[20] Y. Lin and Y. Jeon, “Random forests and adaptive nearest neighbors,” *Journal of the American Statistical Association*, vol. 101, no. 474, pp. 578–590, 2006.

[21] B. H. Menze, W. Petrich, and F. A. Hamprecht, “Multivariate feature selection and hierarchical classification for infrared spectroscopy: serum-based detection of bovine spongiform encephalopathy,” *Analytical and bioanalytical chemistry*, vol. 387, no. 5, pp. 1801–1807, 2007.

[22] Y. Saeys, T. Abeel, and Y. Van de Peer, “Robust feature selection using ensemble feature selection techniques,” in *Machine Learning and Knowledge Discovery in Databases*. Springer, 2008, pp. 313–325.

[23] B. H. Menze, B. M. Kelm, R. Masuch, U. Himmelreich, P. Bachert, W. Petrich, and F. A. Hamprecht, “A comparison of random forest and its gini importance with standard chemometric methods for the feature selection and classification of spectral data,” *BMC bioinformatics*, vol. 10, no. 1, p. 213, 2009.

[24] E. Tuv, A. Borisov, G. Runger, and K. Torkkola, “Feature selection with ensembles, artificial variables, and redundancy elimination,” *The Journal of Machine Learning Research*, vol. 10, pp. 1341–1366, 2009.

[25] X. Su, C.-L. Tsai, H. Wang, D. M. Nickerson, and B. Li, “Subgroup analysis via recursive partitioning,” *The Journal of Machine Learning Research*, vol. 10, pp. 141–158, 2009.

[26] G. Valentini and T. G. Dietterich, “Bias-variance analysis of support vector machines for the development of svm-based ensemble methods,” *The Journal of Machine Learning Research*, vol. 5, pp. 725–775, 2004.

[27] C.-X. Zhang and J.-S. Zhang, “Rotboost: A technique for combining rotation forest and adaboost,” *Pattern Recognition Letters*, vol. 29, no. 10, pp. 1524–1536, 2008.

[28] P. Geurts, D. Ernst, and L. Wehenkel, “Extremely randomized trees,” *Machine learning*, vol. 63, no. 1, pp. 3–42, 2006.

[29] S. Geman, E. Bienenstock, and R. Doursat, “Neural networks and the bias/variance dilemma,” *Neural computation*, vol. 4, no. 1, pp. 1–58, 1992.

[30] E. B. Kong and T. G. Dietterich, “Error-correcting output coding corrects bias and variance.” in *ICML*, 1995, pp. 313–321.

[31] R. Kohavi, D. H. Wolpert *et al.*, “Bias plus variance decomposition for zero-one loss functions,” in *ICML*, 1996, pp. 275–283.

[32] J. H. Friedman, “On bias, variance, 0/1łloss, and the curse-of-dimensionality,” *Data mining and knowledge discovery*, vol. 1, no. 1, pp. 55–77, 1997.

[33] L. Breiman, “Arcing classifier (with discussion and a rejoinder by the author),” *The annals of statistics*, vol. 26, no. 3, pp. 801–849, 1998.

[34] G. M. James, “Variance and bias for general loss functions,” *Machine Learning*, vol. 51, no. 2, pp. 115–135, 2003.

[35] L. Breiman, “Classification and regression trees,” 1984.

[36] K. V. S. Murthy and S. L. Salzberg, “On growing better decision trees from data,” Ph.D. dissertation, Citeseer, 1995.