Multidisciplinary | Rapid Review | Open Access Journal

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2017.DOI

Optimal trees selection for classification via out-of-bag assessment and sub-bagging

ZARDAD KHAN[1,2],NAZ GUL[1], NOSHEEN FAIZ[1,2], ASMA GUL[3], WERNER ADLER[4], BERTHOLD LAUSEN[2,4]

[1]Department of Statistics, Abdul Wali Khan University Mardan, 23200 Pakistan
[2]Department of Mathematical Sciences, University of Essex, UK
[3]Department of Statistics, Shaheed Benazir Bhutto Women University Peshawar, Pakistan
[4]Department of Biometry and Epidemiology, University of Erlangen-Nuremberg, Germany

Corresponding authors: Berthold Lausen (e-mail: blausen@essex.ac.uk) & Zardad Khan (e-mail:zardadkhan@awkum.edu.pk)

"We acknowledge support from grant number ES/L011859/1, from The Business and Local Government Data Research Centre, funded by the Economic and Social Research Council to provide researchers and analysts with secure data services."

ABSTRACT The effect of training data size on machine learning methods has been well investigated over the past two decades. The predictive performance of tree based machine learning methods, in general, improves with a decreasing rate as the size of training data increases. We investigate this in optimal trees ensemble (OTE) where the method fails to learn from some of the training observations due to internal validation. Modified tree selection methods are thus proposed for OTE to cater for the loss of training observations in internal validation. In the first method, corresponding out-of-bag (OOB) observations are used in both individual and collective performance assessment for each tree. Trees are ranked based on their individual performance on the OOB observations. A certain number of top ranked trees is selected and starting from the most accurate tree, subsequent trees are added one by one and their impact is recorded by using the OOB observations left out from the bootstrap sample taken for the tree being added. A tree is selected if it improves predictive accuracy of the ensemble. In the second approach, trees are grown on random subsets, taken without replacement-known as sub-bagging, of the training data instead of bootstrap samples (taken with replacement). The remaining observations from each sample are used in both individual and collective assessments for each corresponding tree similar to the first method. Analysis on 21 benchmark datasets and simulations studies show improved performance of the modified methods in comparison to OTE and other state-of-the-art methods.

INDEX TERMS Tree selection, Classification, Ensemble learning, Out-of-bag sample, Random forest, Sub-bagging

I. INTRODUCTION

Ensemble techniques help to improve machine learning results by integrating multiple models. Using ensemble methods allows to produce better predictions compared to a single base model. There is a huge literature on ensemble methods which is fast growing [1]–[5]. One of the most widely used ensemble method is random forest [6] that combines classification and regression trees [7], [8] as the base model. Classification and regression tree, the building block of many tree based ensemble methods, including random forest, depends both on the quality and quantity of training data [9]. A tree grown with more meaningful information (data points) will give better results than the one built otherwise [9].

The efficacy of combining a large number of individual classifiers, also called base learners, has been well studied [10]–[16]. The main advantage of combining the results of many variants of the same classifier is that it leads to a reduction in the generalization error of the resultant ensemble classifier [11]–[13], [17], [18]. The reason behind this is that the variants of the same classifier have different inductive biases. This kind of diversity results in a reduction of variance-error without increasing the bias-error [19]–[21]. Following this, Breiman [6] argued that diverse and individually strong classifiers will result in an efficient ensemble, while propos-

ing his famous random forest method. Breiman achieved this by selecting $p < d$ features at each node while growing trees on bootstrap samples. The random forest algorithm has been extensively used in solving various classification and regression problems related to medicine [22], banking and finance [23], engineering [24], etc. and has attracted a significant attention of the research community. For further diversity and improvement in tree ensembles, Khan et al. [25], [26] proposed selecting the most accurate trees based on their performance on out-of-bag (OOB) observation. These trees were then further assessed for their collective performance using a subset of the training data as internal validation data for final ensemble. They called this method optimal trees ensemble (OTE). OTE not only showed improved predictive accuracy in comparison to several other state-of-the-art methods, but also reduced ensemble size.

However, while selecting the optimal trees, trees in OTE fail to learn from some of the training observations due to the internal validation. This paper suggests modified methods of tree selection to avoid this issue. In the first method, corresponding out-of-bag (OOB) observations are used in both individual and collective performance assessment for each tree. Trees are ranked based on their individual performance on the OOB observations. A certain number of top ranked trees is selected and starting from the most accurate tree, subsequent trees are added one by one and their impact is recorded by using the OOB observations left out from the bootstrap sample taken for the tree being added. A tree is selected if it improves predictive accuracy of the ensemble. In the second approach, trees are grown on random subsets, taken without replacement, of the training data instead of bootstrap samples. The remaining observations from each sample are used in both individual and collective assessments for each corresponding tree similar to the first method. Using 21 benchmark problems, the results from the new approaches are compared with those of kNN, tree classifier, random forest, node harvest, support vector machine, random projection ensemble and OTE. The methods are further assessed by using the simulation models given in [25] by generating datasets of two different sizes. The remainder of the paper is arranged as follows. The proposed modified approaches, their algorithms and some other related methods are given in Section II, experiments and findings based on simulated and benchmark data sets are given in Section III. Conclusion based on the work done in the article is given in Section IV.

II. OPTIMAL TREE SELECTION

Using the notation of [25], let $\mathcal{L} = (\mathbf{X}, Y) = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), ..., (\mathbf{x}_n, y_n)\}$ be the given training data, where $\mathbf{X}$ is an $n \times d$ matrix and Y a vector of length n. . The $\mathbf{x}_i$ are instances on d features and y_i are binary values representing two possible classes. OTE partitions $\mathcal{L} = (\mathbf{X}, Y)$ randomly into two parts, $\mathcal{L}_\mathcal{B} = (\mathbf{X}_B, Y_B)$ and $\mathcal{L}_\mathcal{O} = (\mathbf{X}_O, Y_O)$. The steps of OTE are given as

1) Trees are developed on T bootstrap samples from $\mathcal{L}_\mathcal{B} = (\mathbf{X}_B, Y_B)$, using the random forest approach.
2) The grown trees are ranked in ascending order of their prediction error on out-of-bag data and M top ranked trees are taken.
3) Starting from the highest ranked tree, the M selected trees are added one by one and $\mathcal{L}_\mathcal{O} = (\mathbf{X}_O, Y_O)$ is applied to see whether the added tree improves predictive accuracy. A tree is selected if it improves accuracy and is discarded otherwise.
4) The selected trees are integrated together for the final ensemble that is used for predicting new/test data.

Although OTE has achieved improved performance as compared to the other methods on the given benchmark and simulated datasets as shown in [25], a problem arises when there is a small number of observations in the data. As the trees are grown on a subset of the training data leaving the remaining observations, say V%, for internal validation, this might result in missing out some useful information to learn from during the process of growing the trees and increases the variance of the classifier [27], [28]. It has been investigated that classification tree strongly depends on the amount of information present in the training data [9]. To utilise the whole training data while growing and selecting optimal trees, two approaches are proposed in this paper.

A. OUT-OF-BAG ASSESSMENT

In this method out-of-bag (OOB) observations are used in both individual and ensemble assessment of the trees. In bootstrapping, as the samples are taken with replacement, some observation are repeated and some are left out from the samples. Studies show that while bootstrapping, about 1/3 of the total training data are left out from the samples [29]. These are called out-of-bag (OOB) observations and play no role in growing classification trees. They can rather be used in assessing the predictive ability of the trees and statistic values thus produced are called OOB estimates. Let S_t, $t = 1, \ldots, T$ be the bootstrap sample and $\bar{S}_t$ be the corresponding OOB sample; $H(S_t)$ is the classification tree grown on S_t. Also suppose that $\widehat{err}_t$ is the error of $H(S_t)$ on $\bar{S}_t$ called the OOB error. About 37% of the observations in the training set $\mathcal{L}$ do not appear in a particular bootstrap sample S_t. These observations can thus be used as unseen test examples. The steps of the proposed method under this approach are:

1) Grow T classification trees by the method of random forest on $S_t, t = 1, \ldots, T$. Estimate $\widehat{err}_t$ for each tree as

$$\widehat{err}_t = \frac{1}{|\bar{S}_t|} \sum_{\mathbf{x_i} \in \bar{S}_t} I(y \neq \hat{y}), \tag{1}$$

where y is the true class label in the bootstrap sample $\bar{S}_t$, $\hat{y}$ is the corresponding estimated value by tree $H(S_t)$ and $|\bar{S}_t|$ is the size of the OOB sample. $I(y \neq \hat{y})$ is an indicator function with values 0 or 1 given as

$$I(y \neq \hat{y}) = \begin{cases} 1, & \text{if } y \neq \hat{y}, \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

2) Arrange the trees for ranking in ascending order with respect to $\widehat{err}_t$; select the top ranked M trees. Let $H^{R_1}(.), \ldots H^{R_M}(.)$, be the highest, second highest and so on, ranked trees.
3) Starting from $H^{R_1}(.)$, test consecutive $H^{R_j}(.), j = 2, \ldots, M$ one by one by using the corresponding OOB observations as the test data. Select $H^{R_j}(.)$ if

$$\hat{\mathcal{BS}}^{\langle j+\rangle} < \hat{\mathcal{BS}}^{\langle j-\rangle}, \tag{3}$$

where $\hat{\mathcal{BS}}^{\langle j-\rangle}$ is the Brier score [30] calculated for the ensemble not having the jth tree and $\hat{\mathcal{BS}}^{\langle j+\rangle}$ is the Brier score of the method including the jth tree. An estimator for the Brier score is given as

$$\hat{\mathcal{BS}} = \frac{\sum_{i=1}^{\#\text{ of test observations}} \left(y_i - \hat{P}(y_i|X)\right)^2}{\text{total \# of test observations}}, \tag{4}$$

y_i is the state of the class value for observation i in the $(0, 1)$ form and $\hat{P}(y|X)$ is the response/class probability estimate of the method given the variables.
4) Integrate the trees for predicting new/test data.

B. SUB-SAMPLING/SUB-BAGGING BASED ASSESSMENT

Under this approach, random sub-samples without replacement from the training data $\mathcal{L} = (X, Y)$ are taken for growing the trees. The remaining observations from each sample are used as the test data for assessing the predictive performance of each corresponding tree, in contrary to using the OOB observations. Let $\mathcal{S}_t$, $t = 1, \ldots, T$ be the random sample of size $m < n$, n being the number of instances in the training data, and $\bar{\mathcal{S}}_t$ be the corresponding remaining subset of observations of size $n - m$; $H(\mathcal{S}_t)$ is the classification tree grown on $\mathcal{S}_t$. Also suppose that $\widehat{err\_sub}_t$ is the error of $H(\mathcal{S}_t)$ on $\bar{\mathcal{S}}_t$. Then the steps of the proposed method under this approach are:

1) Grow T classification trees on $\mathcal{S}_t, t = 1, \ldots, T$. Estimate $\widehat{err\_sub}_t$ for each tree using $\bar{\mathcal{S}}_t$.
2) Rank the trees in ascending order with respect to $\widehat{err\_sub}_t$; select the top ranked M trees. Let $H^{R_1}(.), \ldots H^{R_M}(.)$, be the highest, second highest and so on, ranked trees.
3) Starting from $H^{R_1}(.)$, test consecutive $H^{R_j}(.), j = 2, \ldots, M$ one by one by using the corresponding observations in the sample remainder as the test data. Select $H^{R_j}(.)$ based on the criteria used in Step 3 of the method in previous section.
4) Integrate the trees for predicting new/test data.

Both of the above methods are inspired from Breiman's [6] upper bound defined for the overall prediction error PE^* of random forest algorithm given as

$$PE^* \leq \bar{\rho}\, PE_t. \tag{5}$$

In Equation, 5 $t = 1, 2, 3, ..., T$ where T is the total number of trees grown in the forest, $\bar{\rho}$ is the weighted correlation between residuals from two independent classification trees calculated as the mean (expected) value of their correlation over entire random forest and PE_t is the estimated prediction error of some tth tree in the forest.

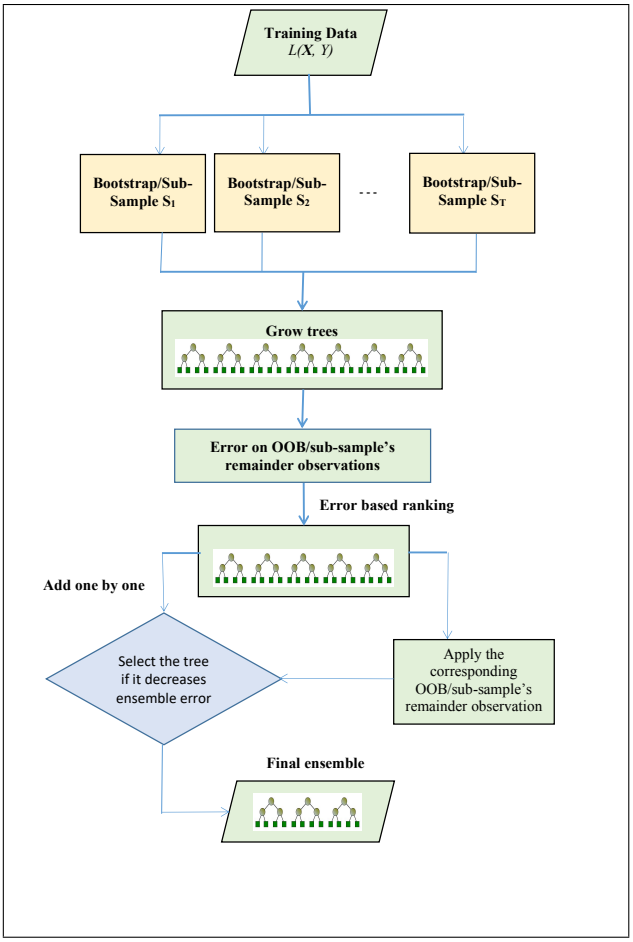

FIGURE 1: Flow chart of the proposed ensembles

A flowchart showing the general work flow of the proposed ensembles in given in Figure 1. Care should be taken for deciding on the size m of sample drawn for growing trees under this approach in relation to the total number of observations in $\mathcal{L}$. This is necessary for avoiding potentially redundant trees in the forest in that there can be only $\binom{n}{m}$ combinations of the training data to grow trees. As the final ensemble selects only a small number of diverse and accurate trees, this approach might be very helpful in small data situations where only a few trees are needed and missing more observations from sample is costly. This approach is expected to work similar to the OOB assessment method when $n-m$ is chosen to be $2/3$ of the training data. Similar study illustrating this has been done in [9].

These approaches are novel in the following sense:

- The proposed methods investigate optimal tree selection without loosing informative training data.

- The method based on sub-bagging tries to allocate more training data as compared to the out-of-bag assessment. This approach keeps 10% of the given training data for internal validation and the remaining 90% of the data is used for growing the trees.
- The tree selection approaches proposed in the paper are based on individual accuracy of the base tree classifiers as well as their diversity in the ensemble in addition to minimizing the loss of informative information in the learning process.
- Based on the above intuitions, the proposed methods could effectively be used in small data situations for optimal trees selection.

C. OTHER RELATED WORK

Several methods are available in literature that are based on the idea of tree selection from bagged tree forest. These methods are based on bagging or its variant in that they improve on unstable estimators or classifiers. Methods based on bagging are useful especially for high dimensional data set problems. Bühlmann and Bin [31] formalized the idea of instability and derived theoretical results to analyze the variance minimization effect of bagging (or the variants). To do this, Bühlmann and Bin [31] considered hard decision problems including estimation after testing in regression and decision trees for classifiers regression functions. They argued that hard decisions create instability, and bagging is helpful in smoothing such hard decisions which results in smaller variance and mean squared error [32]. Bühlmann and Bin [31] motivated sub-bagging based on sub-sampling as an alternative the conventional aggregation scheme by deriving theoretical explanation. Sub-bagging is shown as computationally cheaper with approximately the same accuracy as bagging. Bagging has led to a large pool of methods including random forest and other ensemble classifiers. Authors have further worked on reducing the size of bagging based ensemble methods. Latinne et al. [33] proposed a method to avoid overproducing trees in the ensemble by determining the least number of classification trees that could give comparable results to a standard size ensemble. McNemar test of significance is used decide between forests with different number of trees based on their prediction error. Bernard et al. [34] proposed the methods of sequential backward elimination and sequential forward selection methods to find sub-optimal forests. Li et al. [35] proposed the idea of weighting the trees for random forest ensemble to learn data with large dimensions. They exploited out-of-bag observation for tree weighting in the forest. Adler et al. [36] have proposed ensemble pruning for solving class imbalanced problem by using Brier score and AUC for Glaucoma detection. Different number of trees for random forest were checked by Oshiro et al. [37] so as to see after what point adding further trees results in no gain. They used 29 benchmark datasets to argue that after a certain number, adding further trees does not contribute to ensemble performance. Zhang and Wang [38] proposed the similarity based approach between the trees of the forest and suggested to remove trees that were similar. Khan et al. [25] proposed the idea of building an ensemble of probability estimation trees that are accurate and diverse and proposed to discard trees that are individually weak and do not contribute to ensemble. Based on a similar idea, ensembles selection for kNN classifiers has been given where in addition to individual strength of classifiers, kNN models are build on different random subsets of the whole features set instead of using the entire features [15], [39].

III. EXPERIMENTS AND RESULTS

A. SIMULATION

This section gives our analysis on simulated datasets using the simulation models porposed in [25]. The main idea of using these simulation models is to present slightly difficult recognition problems for simple classifiers like CART and kNN, and also to give a much challenging task for the most sophisticated classifiers like random forest and SVM. To this end, in all the four models, various complexity levels are taken by varying the weights λ_{ijk} of the tree nodes. This gave four different values of the Bayes error for the models where the smallest error means that the dataset has meaningful structure and the highest Bayes error show that there are less/no meaningful structures. Various values of λ_{ijk} used in Scenarios 1, 2, 3, and 4 are given in Table 1. The corresponding node weights for each of the models to get various complexity levels are given in the columns of the table for $k = 1, 2, 3, 4$. Equation used for generating class membership probabilities of the binary class variable, that is, the conditional probability of $Y = \text{Bernoulli}(p)$ given the $n \times 3T$ dimensional vector X of n iid observations from $\text{Uniform}(0, 1)$ is

$$p(y|X) = \frac{exp\left(\theta_2 \times \left(\frac{\mathcal{P}_m}{T} - \theta_1\right)\right)}{1 + exp\left(\theta_2 \times \left(\frac{\mathcal{P}_m}{T} - \theta_1\right)\right)}, \text{ where } \mathcal{P}_m = \sum_{t=1}^{T} \hat{p}_t. \tag{6}$$

θ_1 and θ_2 are arbitrary values, $m = 1, 2, 3, 4$ represents a scenario and $\mathcal{P}_m$'s are $n \times 1$ probability vectors. T shows total number of trees in a scenario and $\hat{p}_t$'s are class probabilities for a binary response in Y. The probabilities defined in Equation 6 add to 1 for the two class labels of a particular observation. The following structure generate the $\hat{p}_t$'s

$$\begin{aligned}
\hat{p}_1 &= \lambda_{11k} \times 1_{(x_1 \leq 0.5 \& x_3 \leq 0.5)} + \lambda_{12k} \times 1_{(x_1 \leq 0.5 \& x_3 > 0.5)} \\
&\quad + \lambda_{13k} \times \mathbf{I}_{(x_1 > 0.5 \& x_2 \leq 0.5)} + \lambda_{14k} \times \mathbf{I}_{(x_1 > 0.5 \& x_2 > 0.5)}, \\
\hat{p}_2 &= \lambda_{21k} \times \mathbf{I}_{(x_4 \leq 0.5 \& x_6 \leq 0.5)} + \lambda_{22k} \times \mathbf{I}_{(x_4 \leq 0.5 \& x_6 > 0.5)} \\
&\quad + \lambda_{23k} \times \mathbf{I}_{(x_4 > 0.5 \& x_5 \leq 0.5)} + \lambda_{24k} \times \mathbf{I}_{(x_4 > 0.5 \& x_5 > 0.5)}, \\
\hat{p}_3 &= \lambda_{31k} \times \mathbf{I}_{(x_7 \leq 0.5 \& x_8 \leq 0.5)} + \lambda_{32k} \times \mathbf{I}_{(x_7 \leq 0.5 \& x_8 > 0.5)} \\
&\quad + \lambda_{33k} \times \mathbf{I}_{(x_7 > 0.5 \& x_9 \leq 0.5)} + \lambda_{34k} \times \mathbf{I}_{(x_7 > 0.5 \& x_9 > 0.5)}, \\
\hat{p}_4 &= \lambda_{41k} \times \mathbf{I}_{(x_{10} \leq 0.5 \& x_{11} \leq 0.5)} + \lambda_{42k} \times \mathbf{I}_{(x_{10} \leq 0.5 \& x_{11} > 0.5)} \\
&\quad + \lambda_{43k} \times \mathbf{I}_{(x_{10} > 0.5 \& x_{12} \leq 0.5)} + \lambda_{44k} \times \mathbf{I}_{(x_{10} > 0.5 \& x_{12} > 0.5)}, \\
\hat{p}_5 &= \lambda_{51k} \times \mathbf{I}_{(x_{13} \leq 0.5 \& x_{14} \leq 0.5)} + \lambda_{52k} \times \mathbf{I}_{(x_{13} \leq 0.5 \& x_{14} > 0.5)} \\
&\quad + \lambda_{53k} \times \mathbf{I}_{(x_{13} > 0.5 \& x_{15} \leq 0.5)} + \lambda_{54k} \times \mathbf{I}_{(x13 > 0.5 \& x_{15} > 0.5)},
\end{aligned}$$

$$\begin{aligned}\hat{p}_6 &= \lambda_{61k} \times \mathbf{I}_{(x_{16}\leq 0.5 \& x_{17}\leq 0.5)} + \lambda_{62k} \times \mathbf{I}_{(x_{16}\leq 0.5 \& x_{17}>0.5)} \\ &+\lambda_{63k} \times \mathbf{I}_{(x_{16}>0.5 \& x_{18}\leq 0.5)} + \lambda_{64k} \times \mathbf{I}_{(x16>0.5 \& x_{18}>0.5)},\end{aligned}$$

where $0 < \lambda_{ijk} < 1$ are node weights in the trees, $k = 1, 2, 3, 4$ and $\mathbf{I}_{(condition)}$ is an indicator function that yields a 1 if the stated condition is satisfied and 0 if not . Note that the basic principle of random forest is followed while growing the trees by taking $p < d$ variables during nodes splitting. The various simulation scenarios are outlined as given below.

1) Scenario 1

This is a relatively simple scenario consisting of $T = 3$ tree components each with 3 variables, $\mathcal{P}_1 = \sum_{t=1}^{3} \hat{p}_t$ and X is a $n \times 9$ vector.

2) Scenario 2

This scenario has four tree components i.e. $T = 4$ trees where $\mathcal{P}_2 = \sum_{t=1}^{4} \hat{p}_t$ which follows that X becomes a $n \times 12$ vector.

3) Scenario 3

This scenario has $T = 5$ trees such that $\mathcal{P}_3 = \sum_{t=1}^{5} \hat{p}_t$ and X is a $n \times 15$ dimensional vector.

4) Scenario 4

This the most complex scenario having $T = 6$ tree components following that, $\mathcal{P}_4 = \sum_{t=1}^{6} \hat{p}_t$ and X is a $n \times 18$ dimensional vector.

TABLE 1: Node weights, λ_{ijk}, used in simulation scenarios. Tree number is shown by i, node number in each tree by j and k shows a variant of the weights to get the complexity levels in each scenario [25].

Scenario 1						Scenario 2						Scenario 3						Scenario 4					
		k						k						k						k			
i	j	1	2	3	4	i	j	1	2	3	4	i	j	1	2	3	4	i	j	1	2	3	4
1	1	0.9	0.8	0.7	0.6	1	1	0.9	0.8	0.7	0.6	1	1	0.9	0.9	0.9	0.8	1	1	0.9	0.9	0.9	0.8
	2	0.1	0.2	0.3	0.4		2	0.1	0.2	0.3	0.4		2	0.1	0.1	0.1	0.2		2	0.1	0.1	0.1	0.2
	3	0.1	0.2	0.3	0.4		3	0.1	0.2	0.3	0.4		3	0.1	0.1	0.1	0.2		3	0.1	0.1	0.1	0.2
	4	0.9	0.8	0.7	0.6		4	0.9	0.8	0.7	0.6		4	0.9	0.9	0.9	0.8		4	0.9	0.9	0.9	0.8
2	1	0.9	0.8	0.7	0.6	2	1	0.9	0.8	0.7	0.6	2	1	0.9	0.9	0.9	0.8	2	1	0.9	0.9	0.9	0.8
	2	0.1	0.2	0.3	0.4		2	0.1	0.2	0.3	0.4		2	0.1	0.1	0.1	0.2		2	0.1	0.1	0.1	0.2
	3	0.1	0.2	0.3	0.4		3	0.1	0.2	0.3	0.4		3	0.1	0.1	0.1	0.2		3	0.1	0.1	0.1	0.2
	4	0.9	0.8	0.7	0.6		4	0.9	0.8	0.7	0.6		4	0.9	0.9	0.9	0.8		4	0.9	0.9	0.9	0.8
3	1	0.9	0.8	0.7	0.6	3	1	0.9	0.8	0.7	0.6	3	1	0.9	0.8	0.7	0.7	3	1	0.9	0.9	0.9	0.8
	2	0.1	0.2	0.3	0.4		2	0.1	0.2	0.3	0.4		2	0.1	0.2	0.3	0.3		2	0.1	0.1	0.1	0.2
	3	0.1	0.2	0.3	0.4		3	0.1	0.2	0.3	0.4		3	0.1	0.2	0.3	0.3		3	0.1	0.1	0.1	0.2
	4	0.9	0.8	0.7	0.6		4	0.9	0.8	0.7	0.6		4	0.9	0.8	0.7	0.7		4	0.9	0.9	0.9	0.8
						4	1	0.9	0.8	0.7	0.6	4	1	0.9	0.8	0.7	0.7	4	1	0.9	0.8	0.7	0.7
							2	0.1	0.2	0.3	0.4		2	0.1	0.2	0.3	0.3		2	0.1	0.2	0.3	0.3
							3	0.1	0.2	0.3	0.4		3	0.1	0.2	0.3	0.3		3	0.1	0.2	0.3	0.3
							4	0.9	0.8	0.7	0.6		4	0.9	0.8	0.7	0.7		4	0.9	0.8	0.7	0.7
												5	1	0.9	0.8	0.7	0.7	5	1	0.9	0.8	0.7	0.6
													2	0.1	0.2	0.3	0.3		2	0.1	0.2	0.3	0.4
													3	0.1	0.2	0.3	0.3		3	0.1	0.2	0.3	0.4
													4	0.9	0.8	0.7	0.7		4	0.9	0.8	0.7	0.6
																		6	1	0.9	0.8	0.7	0.6
																			2	0.1	0.2	0.3	0.4
																			3	0.1	0.2	0.3	0.4
																			4	0.9	0.8	0.7	0.6

TABLE 2: Misclassification rate of kNN, tree, random forest, node harvest, SVM, OTE, OTE_{oob} and OTE_{sub}. Number of observations is 1000 for each model. Bayes error is given in the fourth column against each model.

Scenario	Number of Variables	Number of Observations	Bayes Error	kNN	Tree	RF	NH	SVM (Radial)	SVM (Linear)	SVM (Bessel)	SVM (Laplacian)	OTE 10% V	OTE 20% V	OTE 30% V	OTE_{oob}	OTE_{sub}
			9.2%	23%	11%	10%	11%	20%	20%	20%	20%	10%	11%	12%	10%	11%
			14%	27%	16%	15%	16%	23%	23%	24%	22%	15%	16%	17%	15%	17%
Scenario 1	9	1000	18%	33%	19%	23%	26%	29%	29%	29%	29%	24%	25%	27%	24%	26%
			33%	43%	37%	36%	38%	39%	39%	39%	38%	37%	39%	40%	37%	39%
			21%	31%	23%	21%	23%	25%	24%	30%	25%	21%	23%	24%	21%	21%
			24%	31%	26%	24%	25%	27%	27%	33%	27%	24%	25%	26%	24%	24%
Scenario 2	12	1000	28%	37%	31%	29%	30%	32%	31%	37%	32%	29%	30%	31%	29%	30%
			30%	40%	33%	32%	33%	34%	34%	38%	33%	32%	32%	33%	32%	32%
			16%	32%	23%	19%	23%	25%	25%	56%	25%	20%	21%	22%	19%	19%
			18%	34%	25%	23%	25%	27%	26%	56%	27%	23%	24%	25%	23%	23%
Scenario 3	15	1000	21%	34%	26%	25%	28%	28%	28%	56%	28%	25%	27%	28%	25%	25%
			24%	37%	31%	28%	30%	30%	30%	57%	31%	28%	30%	32%	28%	28%
			22%	35%	29%	23%	26%	26%	26%	72%	28%	24%	25%	26%	23%	24%
			22%	36%	29%	24%	27%	29%	28%	72%	28%	25%	27%	28%	24%	25%
Scenario 4	18	1000	25%	40%	32%	26%	30%	32%	33%	68%	36%	30%	31%	32%	30%	30%
			27%	41%	32%	29%	32%	33%	34%	70%	37%	31%	33%	34%	31%	32%

TABLE 3: Misclassification rate of kNN, tree, random forest, node harvest, SVM, OTE, OTE_{oob} and OTE_{sub}. Number of observations is 100 for each model. Bayes error is given in the fourth column against each model.

Scenario	Number of Variables	Number of Observations	kNN	Tree	RF	NH	SVM (Radial)	SVM (Linear)	SVM (Bessel)	SVM (Laplacian)	OTE	OTE_{oob}	OTE_{sub}
			29%	25%	23%	24%	25%	23%	28%	25%	27%	21%	20%
			32%	28%	27%	27%	30%	28%	37%	30%	30%	26%	25%
Scenario 1	9	100	36%	33%	31%	34%	34%	33%	37%	34%	34%	33%	32%
			39%	46%	43%	45%	43%	42%	44%	44%	42%	42%	43%
			37%	29%	28%	29%	30%	29%	38%	30%	31%	29%	27%
			39%	32%	30%	32%	32%	31%	40%	33%	34%	32%	30%
Scenario 2	12	100	40%	38%	36%	38%	37%	36%	42%	39%	38%	37%	37%
			41%	41%	38%	41%	40%	38%	45%	42%	39%	40%	40%
			39%	35%	31%	32%	31%	30%	53%	34%	33%	32%	30%
			40%	36%	32%	34%	32%	32%	53%	36%	34%	33%	32%
Scenario 3	15	100	42%	37%	31%	36%	34%	33%	53%	37%	34%	33%	33%
			46%	40%	35%	39%	36%	36%	52%	40%	37%	36%	37%
			40%	36%	32%	34%	33%	33%	63%	40%	36%	34%	32%
			42%	37%	33%	37%	34%	34%	62%	40%	36%	35%	34%
Scenario 4	18	100	63%	38%	36%	39%	35%	35%	63%	42%	39%	38%	37%
			46%	39%	38%	40%	38%	37%	63%	44%	40%	41%	40%

Arbitrary constants θ_1 and θ_2 are taken as 0.5 and 15, respectively, for all cases (models and scenarios). To see how the methods perform in small and relatively large sample situations, first we consider generating a total of $n = 1000$ observation using the above setup. All the methods; kNN, CART, node harvest, random forest, , SVM (with four different kernels), OTE, OTE$_{oob}$ and OTE$_{sub}$ are trained by using 70% of the available data as training data and the remaining 30% of the data is used as test data. For OTE$_{sub}$, random sample without replacement for growing the trees are taken from 90% of the training data and the remaining 10% of the training data are used for trees assessment based on individual and ensemble performance. A total of $T = 1000$ trees are grown for OTE as the initial ensemble. For all the methods considered, the same training and test parts are used. Under each scenario, experiments are iterated 1000 times thus getting 1000 realizations of the data for all the methods. Averaging results under the 1000 realizations gives the final results in all the cases. The results are given in Tables 2 and 3. Node weights λ_{ijk} are altered in a way that lead to patterns in the data less or more meaningful and thus getting a high or low Bayes error as shown in column 4 of Table 2. For each of the scenario, four different values of the Bayes error are obtained. The simulation also show that Bayes error of a simulation scenario can be changed by altering the number of trees and/or the node weights. For instance, weights of 0.1 and 0.9 given to extreme nodes (left most and right most) and internal nodes, respectively, will lead to a tree that is less complex compared to a tree with 0.2 and 0.8 such weights. For further explanation, see [25].

Unsurprisingly, tree and kNN have the maximum errors in all the cases of the four scenarios. OTE and Random forest performed comparable with little variations in some cases. For OTE, three values of validation set size $V = 10\%, 20\%, 30\%$ are considered. As can be seen in Table 2, that increasing number of observations V in the validation set the performance of OTE decrease in all the cases of each scenario. In cases where the models generate data with meaningful patterns indicated by low Bayes errors, the results of OTE$_{oob}$ and random forest are better or comparable. OTE$_{sub}$ did not perform well compared to random forest, OTE and OTE$_{oob}$. The reason for this might be that as OTE$_{oob}$ selects only a few trees for the final ensemble, enough randomness in trees could not be guaranteed by growing them on samples of size 90% of training data size drawn without replacement. SVM show similar results to kNN and tree in almost all the cases.

To see how the methods perform in relatively small sample situations, the same simulation scenarios are used to generate datasets consisting of $n = 100$ observations. The results are given in Table 3. This time OTE$_{sub}$ outperforms the rest of the methods in datasets with meaningful structures, i.e. with low Bayes error (not shown). A few accurate and diverse trees can better capture the patterns in the relatively small sized data compared to other methods. In datasets with less meaningful structures, the method still performs similar to SVM which is considered as a promissing classifier in small sample situations. The overall performance of SVM relative to other methods has also improved as compared to the previous situation with $n = 1000$.

B. ANALYSIS OF BENCHMARK DATA SETS

This section presents our analysis on benchmark data sets for OTE$_{oob}$, OTE$_{sub}$ and the other methods considered. A total of 21 data sets are used for comparison purposes. These data sets are described briefly in Table 4. Against each dataset, the number n of observations, number d of features and the corresponding sources from where the data can be taken, are given. Number of features by feature type are also given against each data set. The domain of each dataset is also given in the table.

TABLE 4: Datasets for regression and classification with the corresponding number of instances n, number of variable d and feature/variable type; F: real, I: integer and N: nominal variable in a data set. Sources of the data and their domain are also given.

Data Set	n	d	**Feature type (R/I/N)**	**Source**	Domain
Mammographic	830	5	(0/5/0)	http://sci2s.ugr.es/keel/category.php?cat=clas	Medical Science
Dystrophy	209	5	(2/3/0)	[40]	Medical Science
Monk3	122	6	(0/6/0)	[41]	Machine Learning Benchmark
Appendicitis	106	7	(7/0/0)	http://sci2s.ugr.es/keel/dataset.php?cod=183	Medical Science
SAHeart	462	9	(5/3/1)	http://sci2s.ugr.es/keel/dataset.php?cod=184#sub1	Medical Science
Tic-Tac-Toe	958	9	(0/0/9)	[41]	Gambling
Heart	303	13	(1/12/0)	[41]	Medical Science
House vote	232	16	(0/0/16)	[41]	Medical Science
Bands	365	19	(13/6/0)	http://sci2s.ugr.es/keel/dataset.php?cod=184#sub1	Physical Science
Hepatitis	80	20	(2/18/0)	[41]	Medical Science
Parkinson	195	22	(22/0/0)	[41]	Medical Science
Body	507	23	(22/1/0)	[42]	Medical Science
Thyroid	9172	27	(3/2/22)	[41]	Medical Science
WDBC	569	29	(29/0/0)	[41]	Medical Science
WPBC	198	32	(30/2/0)	[41]	Medical Science
Oil-Spill	937	49	(40/9/0)	http://openml.org/	Environmental Science
Spam base	4601	57	(55/2/0)	[41]	Fraud Detection
Glaucoma	196	62	(62/0/0)	[40]	Medical Science
Nki 70	144	76	(71/5/0)	[43]	Medical Science
Musk	476	166	(0/166/0)	[44]	Chemical Science

C. EXPERIMENTAL SETUP

Experimental setup for applying the methods on the 21 datasets is as follows. Given datassets are divided into training and testing parts consisting of 70% and 30%, respectively, of the total data. Splitting into $50\% - 50\%$ and $30\% - 70\%$ parts for training-testing are also considered. A total of 1000 random splittings of the given data are done into training and testing parts with methods trained on the training parts and tested by testing parts. Final result is obtained by averaging the results of all these 1000 splittings.

For the original OTE, various values of validation set sizes, i.e. $|V| = 10, 15, 20$, are used to see its effect on the predictive performance of the method. A total of $T = 1500$ trees are grown on independent bootstrap samples from the respective 90% , 85% and 80% of training data by the method of random forest. The remaining 10%, 15% and 20% data, respectively, are used for internal validation as mentioned above.

For OTE$_{oob}$, $T = 1500$ trees are grown on bootstrap samples from the whole of the available training data. OOB observations are stored for individual and ensemble tree assessment. For OTE$_{sub}$ $T = 1500$ trees are grown on random samples without replacement of size 90% of training data size. The remaining 10% are used for individual and ensemble performance assessment of each corresponding tree. The number p of features is fixed at $p = \sqrt{d}$ for all data sets. M is fixed at 20% of T.

Various hyper-parameters of CART are tuned by using the `tune.rpart` R-function available within the R-Package `e1071` [45]. Various values, (5,10,15,20,25) are tried to find the minimal optimal depth and optimal number of splits for the trees.

In the case of random forest, node size (`nodesize`), number of trees (`ntree`) and subset size (p) of features (`mtry`) are tuned by using `tune.randomForest` function available with in the R-Package `e1071` as used by [25], [46]. Searches for the best node size (`nodesize`) are mede among values (1,5,10,15,20,25,30), for `ntree` amongst values (500,1000,1500,2000) and for `mtry` (sqrt(d), d/5, d/4, d/3, d/2) are checked. All the possible values of `mtry` are checked where $d < 12$.

For node harvest estimator, the only heper-parameter is the number of nodes in the initial ensemble. Meinshausen [47] has shown that for its large values the changes in the results are negligable and stated that initial number of nodes greater than 1000 gives almost the same results. In this paper, this value is fixed at 1500. R implementation as given in the package `nodeHarvest` [48] is used. For support vector machine, automatic estimation of sigma is utalised from the R package `kernlab` [44]. For the remining parameters, their default values are used with four kernels, Radial, Linear, Bessel and Laplacian. k-nearest neighbours classifier, kNN, is tuned for the optimal value of its hyper-parameter k, the number of nearest neighbouts, by using `tune.knn` R function within the R library `e1071`. Values of $k = 1, \ldots, 10$ are tried.

For random projection (RP) ensemble method [49], the R package `RPEnsemble` [50] is used. Due to computational constraint B_1 and B_2 are fixed at 30 and 5 respectively. Quadratic discriminant analysis `base = "QDA"` and linear discriminant analysis `base = "LDA"` procedures are used as the base learner along with `d=5, projmethod = "Haar"`. The remaining parameters are kept at their default values.

For a fair comparison, training and test data are taken the same for tree, node harvest, random forest, SVM, RP, OTE, OTE$_{oob}$ and OTE$_{oob}$. Average classification errors are recorded for all the methods on all the data sets. R version 4.0.1 [51], on a 3 GHz Intel Core i7 computer with 8 GB memory running under mac OS X operating system, is used for the experiments. The results for various training and testing parts are given in Tables 5, 6 and 7. For further assessment of the proposed methods in comparison with the rest, Brier score, sensitivity and Kappa statistics values are also estimated. These statistics are estimated based on 30% training and 70% testing partitions of the given datasets for checking the behaviour of the ensembles in small sample training data. The results in terms of Brier score, sensitivity and Kappa are given in Tables **??**, 3 and 4, respectively.

TABLE 5: Misclassification rates of kNN, tree, random forest, node harvest, SVM (with four kernels), random projection with linear and quadratic discriminant analyses, OTE, OTE_{oob} and OTE_{sub}. Results are based on 70% training and 30% testing parts of the data. Overall best performing method result is shown in bold. The results are italicised when OTE_{oob} and/or OTE_{sub} are/is better than *OTE*.

Dataset	n	p	kNN	Tree	NH	SVM (Radial)	SVM (Linear)	SVM (Bessel)	SVM (Laplacian)	RP (LDA)	RP (QDA)	RF	OTE 10% V	OTE 15% V	OTE 20% V	OTE_{oob}	OTE_{sub}
Mammographic	830	5	0.2015	0.1648	**0.1632**	0.1882	0.1763	0.1862	0.1860	0.1928	0.1966	0.1643	0.1808	0.1804	0.1801	0.1839	0.1912
Dystrophy	209	5	0.1294	0.1495	0.1527	0.1039	0.1137	0.1085	0.1038	0.1221	**0.0946**	0.1256	0.1226	0.1247	0.1263	*0.1218*	0.1243
Monk3	122	6	0.1388	0.0946	0.2740	0.1048	0.2325	0.1007	0.1380	0.2177	0.1150	0.0708	0.0802	0.0761	0.0755	***0.0701***	0.0780
Appendicitis	106	7	0.1511	0.1569	0.1458	0.2086	0.1771	0.1883	0.1599	**0.1280**	0.1559	0.1336	0.1519	0.1534	0.1547	*0.1451*	0.1470
SAHeart	462	9	0.3471	0.3241	**0.2826**	0.3132	0.3134	0.3403	0.3191	0.3036	0.3057	0.2995	0.3184	0.3192	0.3190	*0.3166*	0.3286
Tic-Tac-Toe	958	9	0.3798	0.1955	0.2916	0.2392	0.4013	0.1943	0.3800	0.3151	0.2424	0.0511	0.0557	0.0649	0.0693	0.0579	**0.0421**
Heart	303	13	0.3608	0.2290	0.1999	0.1734	0.1779	0.3865	**0.1672**	0.2033	0.2159	0.1738	0.1995	0.2023	0.2031	*0.1845*	0.1875
House Vote	232	16	0.0911	0.0443	0.1112	0.0486	0.0456	0.0477	0.0418	0.0686	0.0687	**0.0401**	0.0423	0.0429	0.0435	*0.0420*	0.0416
Bands	365	19	0.3255	0.3056	0.3775	0.2988	0.2798	0.3622	0.5279	0.3379	0.3126	0.2333	0.2379	0.2445	0.2482	0.2380	**0.2301**
Hepatitis	80	20	0.3966	0.1846	0.1330	0.1368	0.1650	0.4884	0.1576	0.1883	0.1512	0.1530	0.1320	0.1353	0.1370	*0.1305*	**0.0419**
Parkinson	195	22	0.1752	0.1414	0.1334	0.1622	0.2012	0.2653	0.2089	0.1812	0.1568	0.1026	0.0976	0.0994	0.1011	*0.0966*	**0.0951**
Body	507	23	0.0315	0.0828	0.0854	0.0186	0.0170	0.5419	**0.0352**	0.0208	0.0242	0.0435	0.0409	0.0429	0.0438	0.0410	0.0382
Thyroid	9172	27	0.0391	0.0128	0.0290	0.1118	0.0328	0.3084	0.0766	0.0497	0.0440	**0.0102**	0.0105	0.0107	0.0108	*0.0103*	**0.0102**
WDBC	569	29	0.0771	0.0683	0.0613	0.0432	0.0272	0.6175	0.0444	0.0575	0.0564	0.0419	0.0416	0.0426	0.0438	*0.0404*	**0.0389**
WPBC	198	32	0.2691	0.2953	0.2328	0.2960	0.2862	0.5521	0.3546	0.2198	0.2320	0.2088	**0.2077**	0.2139	0.2181	0.2078	0.2127
Oil-Spill	937	49	0.0562	0.0394	0.0410	0.0774	0.0956	0.3641	0.1189	0.0440	0.0434	0.0371	0.0348	0.0354	0.0357	*0.0347*	**0.0344**
Spam base	4601	58	0.1788	0.1064	0.1004	0.0917	0.0743	0.4919	0.1058	0.2157	0.2258	0.0493	0.0493	0.0503	0.0486	*0.0482*	**0.0468**
Sonar	208	60	0.1819	0.2901	0.2429	0.1832	0.2562	0.5389	0.3011	0.2624	0.2138	0.1916	0.1823	0.1887	0.1925	*0.1746*	**0.1642**
Glaucoma	196	62	0.2010	0.1339	0.1254	0.1200	0.1573	0.6468	0.1233	**0.1076**	0.1432	0.1125	0.1125	0.1146	0.1166	0.1163	0.1190
Nki 70	144	76	0.1878	0.1662	0.1565	0.2278	0.3321	0.4031	0.4958	0.1813	0.1900	0.1460	**0.1456**	0.1478	0.1491	0.1494	0.1665
Musk	476	166	0.1438	0.2221	0.2547	0.1490	0.1623	0.4894	0.5187	0.1010	**0.0844**	0.1200	0.1134	0.1188	0.1248	*0.1079*	0.1068

TABLE 6: Misclassification rates of kNN, tree, random forest, node harvest, SVM (with four different kernels), random projection with quadratic and linear discriminant analyses, OTE, OTE_{oob} and OTE_{sub}. Results are based on 50% training and 50% testing parts of the data. Overall best performing method result is shown in bold. The results are italicised when OTE_{oob} and/or OTE_{sub} are/is better than *OTE*.

Dataset	n	p	kNN	Tree	NH	SVM (Radial)	SVM (Linear)	SVM (Bessel)	SVM (Laplacian)	RP (LDA)	RP (QDA)	RF	OTE	OTE_{oob}	OTE_{sub}
Mammographic	830	5	0.2100	0.1661	0.1760	0.1882	0.1772	0.1836	0.1845	0.1890	0.1993	**0.1614**	0.1793	0.1824	0.1916
Dystrophy	209	5	0.1354	0.1831	0.1632	0.1101	0.1191	0.1112	0.1087	0.1233	**0.0959**	0.1316	0.1311	*0.1285*	0.1308
Monk3	122	6	0.1421	0.0773	0.2855	0.1228	0.2367	0.1137	0.1731	0.2308	0.1352	0.0790	0.0887	***0.0735***	0.0833
Appendicitis	106	7	0.1624	0.1777	0.1569	0.1937	0.2026	0.1885	0.1631	**0.1358**	0.1692	0.1418	0.1681	*0.1493*	0.1526
SAHeart	462	9	0.3499	0.3409	**0.2876**	0.3179	0.3153	0.3430	0.3249	0.3057	0.3140	0.2979	0.3184	0.3202	0.3282
Tic-Tac-Toe	958	9	0.3790	0.1884	0.3109	0.2771	0.4073	0.2233	0.4462	0.3266	0.2536	0.0751	0.0791	0.0826	***0.0642***
Heart	303	13	0.3701	0.2370	0.2140	0.1761	0.1830	0.3807	**0.1703**	0.2120	0.2288	0.1790	0.2037	*0.1899*	0.1929
House Vote	232	16	0.0987	0.0472	0.1211	0.0514	0.0467	0.0497	0.0655	0.0699	0.0717	**0.0418**	0.0441	*0.0434*	0.0438
Bands	365	19	0.3299	0.3289	0.3827	0.3432	0.2969	0.4397	0.5185	0.3413	0.3184	**0.2522**	0.2580	*0.2557*	*0.2525*
Parkinson	195	22	0.1822	0.1720	0.1430	0.1831	0.2071	0.2564	0.2234	0.1842	0.1671	0.1200	0.1143	*0.1135*	***0.1093***
Body	507	23	0.0410	0.1006	0.0887	0.0229	0.0187	0.5273	0.0420	**0.0214**	0.0240	0.0496	0.0473	*0.0472*	*0.0454*
Thyroid	9172	27	0.0401	0.0130	0.0302	0.1153	0.0370	0.3546	0.0718	0.0500	0.0454	0.0110	0.0111	*0.0110*	***0.0108***
WDBC	569	29	0.0790	0.0749	0.0754	0.0503	0.3020	0.5979	0.0524	0.0584	0.0567	0.0461	0.0468	*0.0445*	***0.0443***
WPBC	198	32	0.2712	0.2836	0.2483	0.3117	0.2970	0.5208	0.4181	**0.2242**	0.2463	0.2246	0.2276	*0.2260*	0.2324
Oil-Spill	937	49	0.0654	0.0465	0.0521	0.0799	0.0933	0.3087	0.1273	0.0446	0.0435	0.0490	0.0380	***0.0379***	***0.0379***
Spam base	4601	58	0.1912	0.1059	0.1124	0.0941	0.0758	0.4895	0.1190	0.2180	0.3149	0.0527	0.0522	*0.0520*	***0.0506***
Sonar	208	60	0.1900	0.2984	0.2568	0.2045	0.2596	0.5393	0.4000	0.2689	0.2249	0.2075	0.2046	*0.2035*	***0.1920***
Glaucoma	196	62	0.2102	0.1526	0.1375	0.1374	0.1663	0.6430	0.2024	**0.1182**	0.1439	0.1200	0.1279	*0.1255*	0.1279
Nki 70	144	76	0.1912	0.1508	0.1648	0.2000	0.3281	0.3780	0.5097	0.1922	0.1984	0.1561	**0.1545**	0.1635	0.1865
Musk	476	166	0.1599	0.2583	0.2654	0.1700	0.1896	0.4946	0.5152	0.1128	**0.0964**	0.1415	0.1412	*0.1373*	*0.1341*

TABLE 7: Misclassification rates of kNN, tree, random forest, node harvest, SVM (with four different kernels), random projection with quadratic and linear discriminant analyses, OTE, OTE_{oob} and OTE_{sub}. Results are based on 30% training and 70% testing parts of the data. Overall best performing method result is shown in bold. The results are italicised when OTE_{oob} and/or OTE_{sub} are/is better than *OTE*.

Dataset	n	p	kNN	Tree	NH	SVM (Radial)	SVM (Linear)	SVM (Bessel)	SVM (Laplacian)	RP (LDA)	RP (QDA)	RF	OTE	OTE_{oob}	OTE_{sub}
Mammographic	830	5	0.2190	0.1665	0.1792	0.1911	0.1806	0.1836	0.1860	0.1857	0.1980	**0.1640**	0.1787	0.1827	0.1947
Dystrophy	209	5	0.1399	0.2098	0.1725	0.1212	0.1272	0.1175	0.1176	0.1276	**0.1041**	0.1379	0.1458	*0.1392*	0.1479
Monk3	122	6	0.1499	0.0883	0.2988	0.1732	0.2542	0.1610	0.2357	0.2466	0.1938	0.0973	0.1230	***0.0874***	0.0962
Appendicitis	106	7	0.1711	0.1998	0.1659	0.1954	0.2285	0.1889	0.1664	0.1506	0.1751	**0.1496**	0.1934	*0.1744*	*0.1730*
SAHeart	462	9	0.3929	0.3549	**0.2987**	0.3299	0.3210	0.3500	0.3422	0.3108	0.3225	0.3044	0.3232	0.3258	0.3328
Tic-Tac-Toe	958	9	0.3817	0.1988	0.3337	0.3453	0.4172	0.2783	0.5037	0.3381	0.2633	0.1343	0.1203	***0.0828***	0.1067
Heart	303	13	0.3798	0.2749	0.2259	0.1835	0.1999	0.3677	**0.1838**	0.2225	0.2456	0.1939	0.2155	*0.2029*	0.2034
House Vote	232	16	0.0995	0.0520	0.1357	0.0596	0.0483	0.0567	0.0810	0.0745	0.0783	**0.0465**	0.0509	*0.0471*	0.0472
Bands	365	19	0.3332	0.3433	0.3876	0.4465	0.3210	0.4970	0.5070	0.3462	0.3318	0.2780	0.2834	***0.2771***	0.2815
Parkinson	195	22	0.1899	0.1883	0.1687	0.2046	0.2150	0.2530	0.2370	0.1919	0.1891	0.1466	0.1433	*0.1401*	***0.1381***
Body	507	23	0.0419	0.0994	0.0901	0.0290	0.0216	0.5019	0.0555	**0.0236**	0.0259	0.0589	0.0579	*0.0563*	*0.0540*
Thyroid	9172	27	0.0411	0.0137	0.0402	0.1289	0.0456	0.3479	0.0712	0.0501	0.0479	0.0131	0.0124	*0.0122*	***0.0120***
WDBC	569	29	0.0793	0.0802	0.0812	0.0604	0.0350	0.5698	0.0600	0.0602	0.0575	0.0537	0.0544	*0.0525*	***0.0513***
WPBC	198	32	0.2914	0.2985	0.2549	0.3273	0.3104	0.4366	0.4534	0.2348	0.2530	**0.2446**	0.2580	*0.2570*	0.2624
Oil-Spill	937	49	0.0697	0.0483	0.0621	0.0845	0.0824	0.1789	0.1754	0.0463	0.0429	0.0431	0.0411	0.0412	***0.0409***
Spam base	4601	58	0.1999	0.1078	0.1215	0.1176	0.0798	0.4829	0.3014	0.2368	0.3255	0.0577	0.0581	*0.0572*	***0.0564***
Sonar	208	60	0.2562	0.3318	0.2659	0.2484	0.2714	0.5335	0.5051	0.2870	0.2605	0.2494	0.2453	*0.2378*	***0.2327***
Glaucoma	196	62	0.2270	0.1953	0.1463	0.1681	0.1861	0.6269	0.2516	0.1365	0.1565	0.1589	0.1585	***0.1519***	*0.1541*
Nki 70	144	76	0.1979	0.1822	0.1758	0.2797	0.3304	0.3589	0.5099	**0.1928**	0.2103	0.1943	0.2218	0.2166	0.2310
Musk	476	166	0.1614	0.3058	0.2846	0.2147	0.2327	0.5015	0.5043	0.1280	**0.1089**	0.1849	0.1906	*0.1863*	*0.1838*

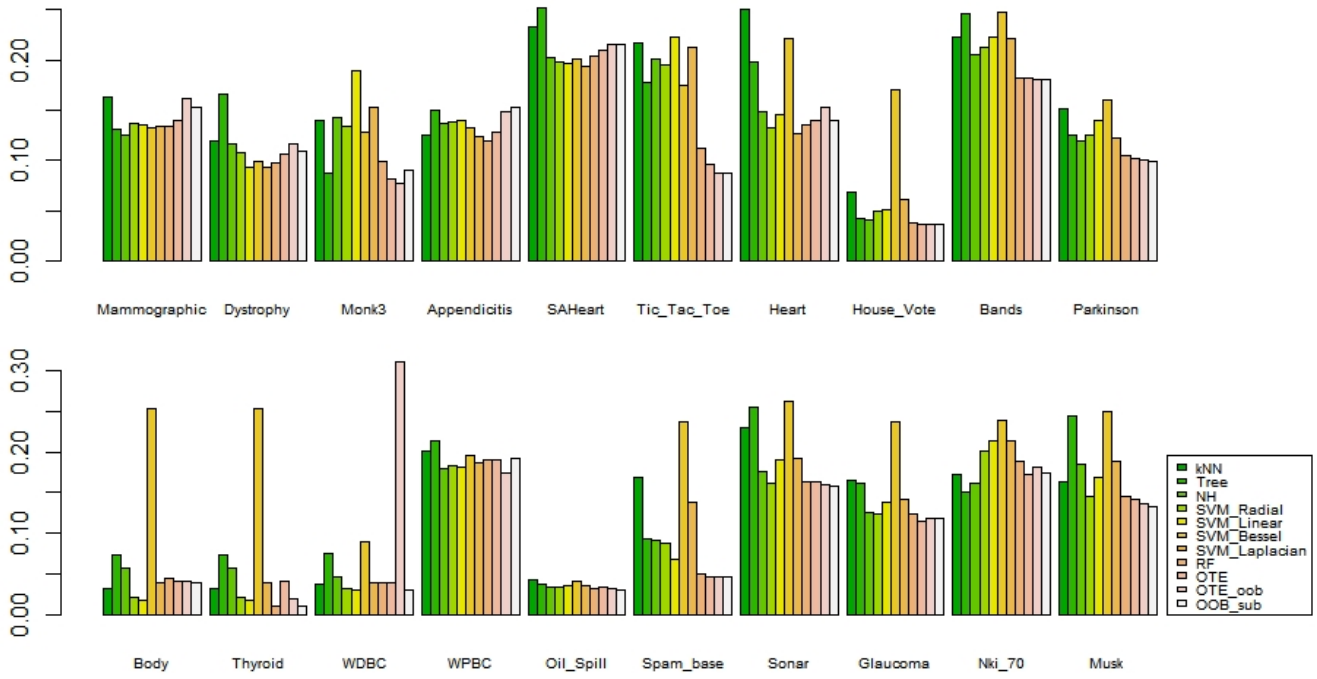

FIGURE 2: Barplots for Brier score.

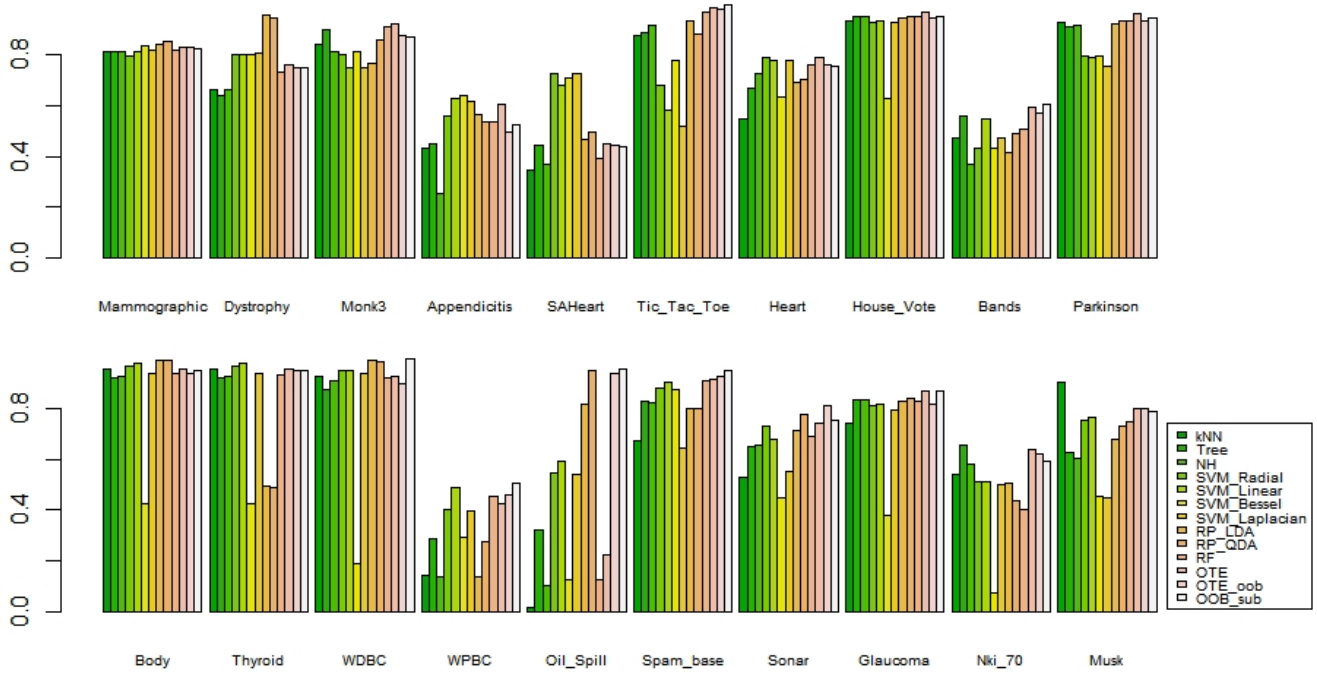

FIGURE 3: Barplots for Sensitivity.

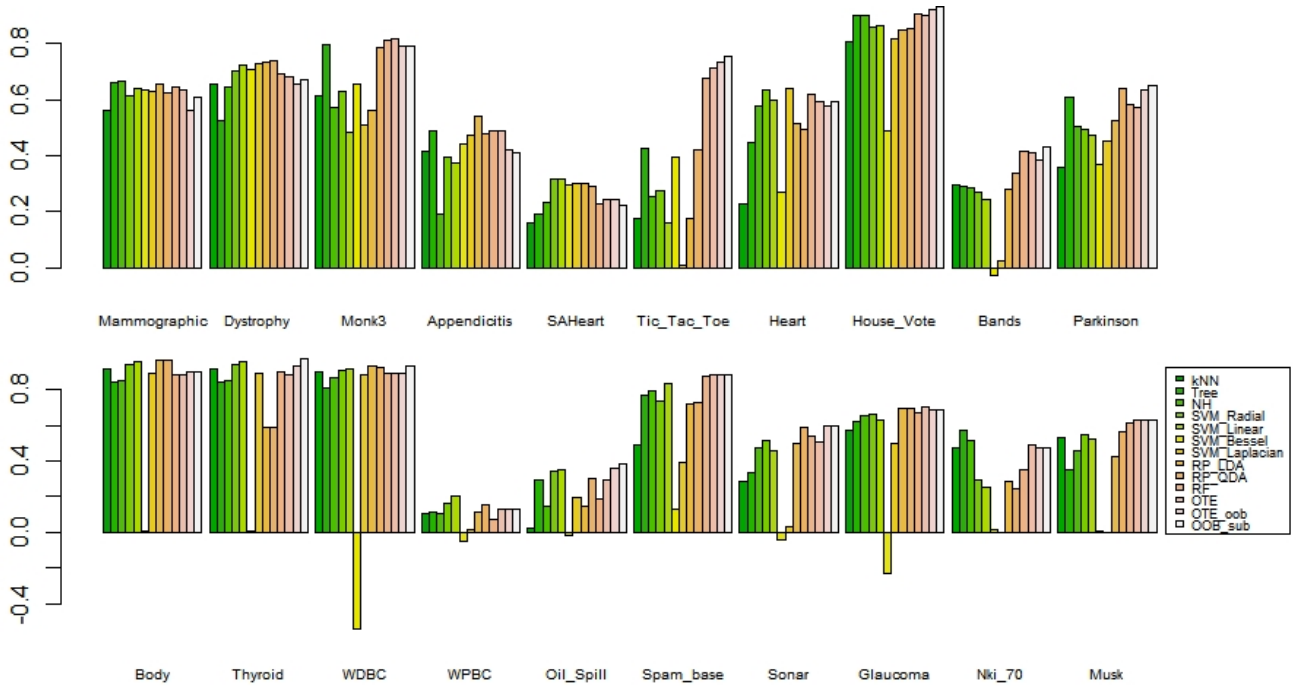

FIGURE 4: Barplots for Kappa.

D. DISCUSSION

Results given in Tables 5, 6 and 7, and barplots given in Figures 2,3 and 4 reveal that the OTE_{oob} and OTE_{sub} are almost always better than OTE. The results in Table 5 also show that OTE with $V = 10\%$ is always giving better results than OTE with $V = 15\%$ and so on, with the exception of Mammographic dataset only. From Table 5, that shows results based on 70% and 30% splitting of the data, it can be seen that node harvest and SVM gave better results than the others on 2 data sets each. RP ensemble gave better results than the rest on 4 datasets 2 each for LDA and QDA base learner. OTE is better than the others on 2 datasets with $V = 10\%$. OTE_{oob} although better than OTE in most of the cases, outperformed the rest of the methods on 1 dataset. OTE_{sub} gave better results than the others on 9 of the datasets. Tree and kNN methods could not outperformed the rest of the methods on any dataset.

From Table 6, that shows results based on 50% and 50% splitting of the data, it can be seen that node harvest and SVM gave better results than the others on 1 data sets each. RP ensemble gave better results than the rest on 6 datasets. Random forest is better than the others on 3 data sets. OTE is better than the others on 1 dataset. OTE_{oob} is better than OTE in most of the cases, and outperformed the rest of the methods on 2 dataset. OTE_{sub} gave better results than the others on 7 of the datasets. Tree and kNN methods could not outperformed the rest of the methods on any dataset.

The results in Table 7, based on 70% and 30% splitting of the data, show that node harvest and SVM gave better results than the others on 1 data sets each. RP ensemble gave better results than the rest on 4 datasets 2 each for LDA and QDA base learner. Random forest is better than the others on 4 data sets. OTE could not outperformed the others on any of the data set. OTE_{oob} is better than OTE in most of the cases, and outperformed the rest of the methods on 4 dataset. OTE_{sub} gave better results than the others on 6 of the datasets. Tree and kNN methods could not outperformed the rest of the methods on any of the datasets.

Furthermore, the results of the methods in terms of Brier score, sensitivity and Kappa, given in Figures 2, 3 and 4, respectively, indicate that the proposed ensembles outperformed the rest of the methods on majority of the datasets. Brier score values are not estimated for random projection ensemble (Table **??**) as the current implementation of the algorithm given in the R package [50] does not support probability estimation.

Moreover, the effect of choosing various number of trees on the three methods, i.e. OTE, OTE_{oob} and OTE_{sub} in terms of classification error rates are shown in Figure 5, 6 and 7, respectively. In the given figures, the value of M in percentage is shown on the x-axis and error rate on the y-axis. Number of trees selected are also shown in brackets on the x-axis, e.g. $10(< 40)$ means that the method selected less than 40 trees for the datasets at $M = 10\%$.

Getting better/comparable results by using a forest of few accurate and diverse trees to those based on thousands of weak trees is encouraging in that this might reduce computational costs in terms of storage resources. From size assessment of the proposed ensemble methods, it is evident that they provide the best result with the number of trees less than 50. This is a clear reduction in the ensembles size and could have significant practical implications.

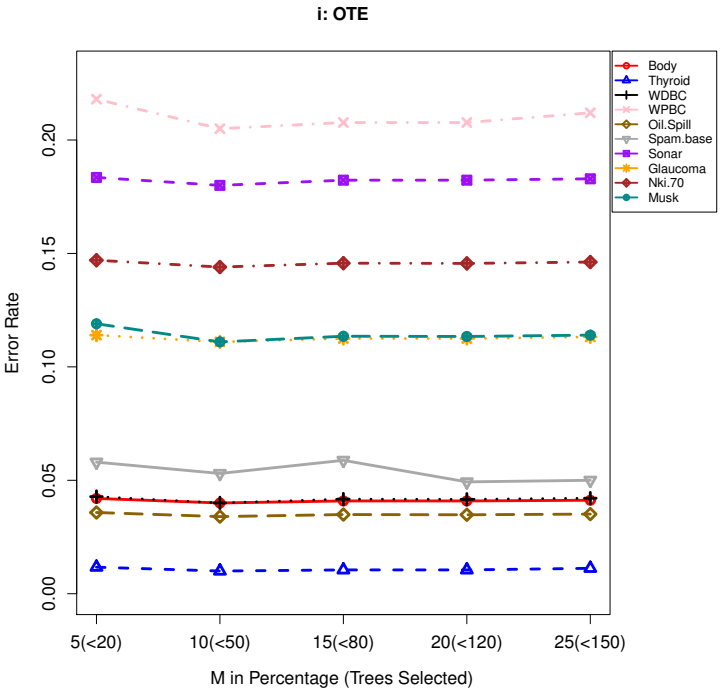

FIGURE 5: Effect of M on the error rate of the data sets shown using *OTE*. The value of M in percentage is on the x-axis and error rate is on the y-axis. Number of trees selected are also shown in brackets on the x-axis, e.g. $10(< 50)$ means that the method selected less than 50 trees for the datasets at $M = 10\%$

IV. CONCLUSION

Two methods of selecting optimal trees, based on the individual strenght of a tree and trees collective performance, from an original ensemble of a large number of trees are proposed as an improvement to OTE. The selected trees are then combined together to vote for the class labels of the unseen data. Using as much as possible of the training data for growing trees in the two proposed method guarantees better results. This makes the trees individually strong and as the methods implement a diversity check on the trees while selecting them for the final ensemble, enough randomness is maintained in base learners meeting the basic principles of ensemble learning. The analyses given in the paper, both on simulated and benchmark datasets, revealed that the proposed methods outperform the other state-of-the-art methods.

R implementation of the proposed ensembles is given in Package "OTE" [52].

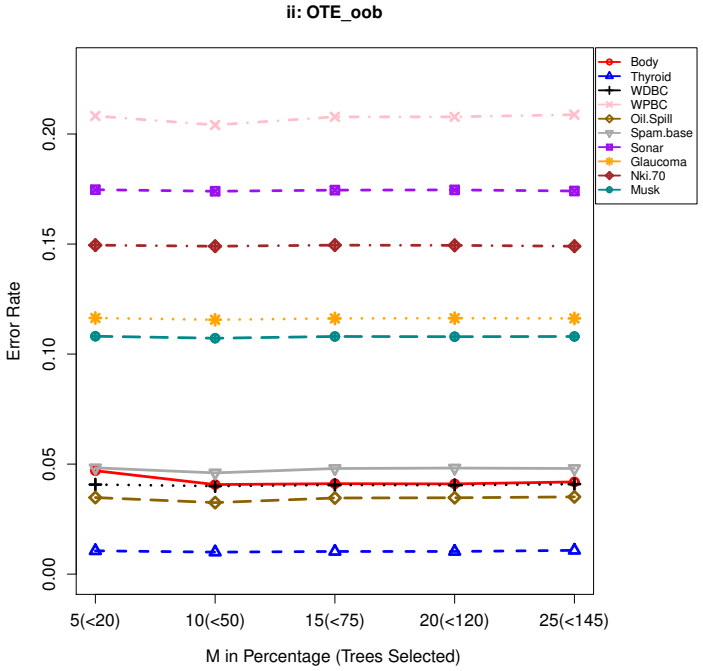

FIGURE 6: Effect of M on the error rate of the data sets shown using OTE_{oob}. The value of M in percentage is on the x-axis and error rate is on the y-axis. Number of trees selected are also shown in brackets on the x-axis, e.g. $10(< 50)$ means that the method selected less than 50 trees for the datasets at $M = 10\%$

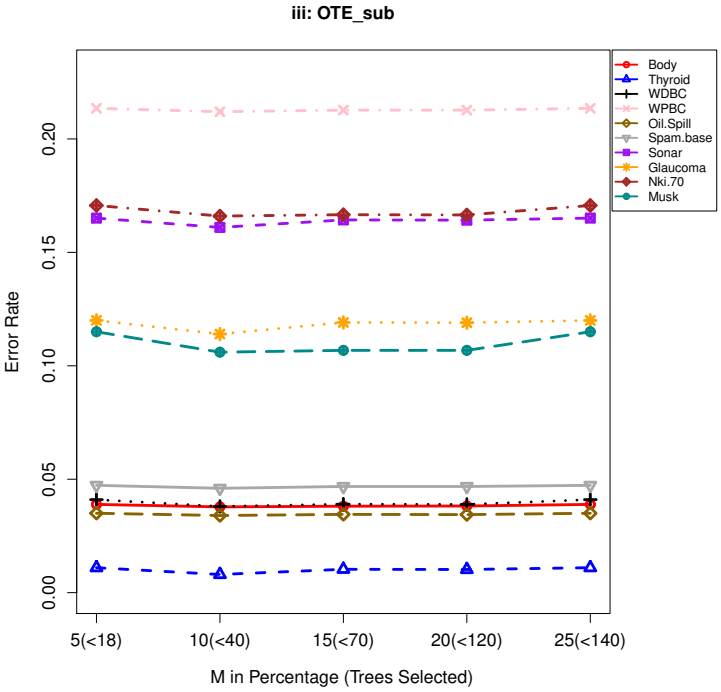

FIGURE 7: Effect of M on the error rate of the data sets shown using OTE_{oob}. The value of M in percentage is on the x-axis and error rate is on the y-axis. Number of trees selected are also shown in brackets on the x-axis, e.g. $10(< 40)$ means that the method selected less than 40 trees for the datasets at $M = 10\%$

The proposed ensemble in its current version takes more training time than the random forest algorithm. For example, with Thyriod data ($n = 9172, d = 27$), the training times for random forest and the proposed methods were 4.56 and 6.41 seconds, respectively, on a 3 GHz Intel Core i7 computer with 8 GB memory running under mac OS X operating system. The methods proposed in the paper can model massive data with ultra high dimension using parallel computing as implemented in the R package [53], for example. Using feature selection methods, [54]–[61], might, in conjunction with the proposed ensembles, result in further improvements [62]. Using random projection approach as given in [49], [50] with the tree selection methods proposed in this paper, may also give further improvements. The idea of classifier selection based on clustering (CSBS) [63], for ensemble creation could also be used with the proposed ensembles for efficient results. For data sets with features measured on different scales, random forest with P-value adjusted split criteria can avoid biased feature selection within the tree algorithm [64]–[66]

REFERENCES

[1] H. Quintián and E. Corchado. A novel ensemble beta-scale invariant map algorithm. IEEE Access, 8:108857–108884, 2020.

[2] H. Yang, H. Peng, J. Zhu, and F. Nie. Co-clustering ensemble based on bilateral k-means algorithm. IEEE Access, 8:51285–51294, 2020.

[3] P. Wang and X. Chen. Three-way ensemble clustering for incomplete data. IEEE Access, 8:91855–91864, 2020.

[4] A. Ali, M. Hamraz, P. Kumam, D. M. Khan, U. Khalil, M. Sulaiman, and Z. Khan. A k-nearest neighbours based ensemble via optimal model selection for regression. IEEE Access, 8:132095–132105, 2020.

[5] J. Jia and W. Qiu. Research on an ensemble classification algorithm based on differential privacy. IEEE Access, 8:93499–93513, 2020.

[6] L. Breiman. Random forests. Machine learning, 45(1):5–32, 2001.

[7] L. Breiman, J. Friedman, C. J. Stone, and R. A. Olshen. Classification and regression trees. Chapman & Hall/CRC, New York, 1984.

[8] L. Breiman. Bagging predictors. Machine learning, 24(2):123–140, 1996.

[9] M. Sebban, R. Nock, J. Chauchat, and R. Rakotomalala. Impact of learning set quality and size on decision tree performances. IJCSS, 1(1):85, 2000.

[10] R. E. Schapire. The strength of weak learnability. Machine learning, 5(2):197–227, 1990.

[11] P. Domingos. Using partitioning to speed up specific-to-general rule induction. In Proceedings of the AAAI-96 Workshop on Integrating Multiple Learned Models, pages 29–34. Citeseer, 1996.

[12] J. R. Quinlan. Bagging, boosting, and c4. 5. In Proceedings of the National Conference on Artificial Intelligence, pages 725–730, 1996.

[13] R. Maclin and D. Opitz. Popular ensemble methods: An empirical study. Journal of Artificial Research, 11:169–189, 2011.

[14] T. Hothorn and B. Lausen. Double-bagging: Combining classifiers by bootstrap aggregation. Pattern Recognition, 36(6):1303–1309, 2003.

[15] A. Gul, A. Perperoglou, Z. Khan, O. Mahmoud, M. Miftahuddin, W. Adler, and B. Lausen. Ensemble of a subset of knn classifiers. Advances in Data Analysis and Classification, pages 1–14, 2016.

[16] L. Lausser, F. Schmid, L. R. Schirra, A. F. Wilhelm, and H. A. Kestler. Rank-based classifiers for extremely high-dimensional gene expression data. Advances in Data Analysis and Classification, pages 1–20, 2016.

[17] E. Bauer and R. Kohavi. An empirical comparison of voting classification algorithms: Bagging, boosting, and variants. Machine learning, 36(1):105–139, 1999.

[18] P. Tzirakis and C. Tjortjis. T3c: improving a decision tree classification algorithmâĂŹs interval splits on continuous attributes. Advances in Data Analysis and Classification, 11(2):353–370, 2017.

[19] T. M. Mitchell. Machine learning. Burr Ridge, IL: McGraw Hill, 1997.

[20] K. Tumer and J. Ghosh. Error correlation and error reduction in ensemble classifiers. Connection science, 8(3-4):385–404, 1996.

[21] K. M. Ali and M. J. Pazzani. Error reduction through learning multiple descriptions. Machine Learning, 24(3):173–202, 1996.

[22] M. Zhu, L. Xia, X. Jin, M. Yan, G. Cai, L. Yan, and G. Ning. Class weights random forest algorithm for processing class imbalanced medical data. IEEE Access, 6:4641–4652, 2018.

[23] W. Lin, Z. Wu, L. Lin, A. Wen, and L. Li. An ensemble random forest algorithm for insurance big data analysis. Ieee access, 5:16568–16575, 2017.

[24] S. Kim, S. Kwak, and B. C. Ko. Fast pedestrian detection in surveillance video based on soft target training of shallow random forest. IEEE Access, 7:12415–12426, 2019.

[25] Z. Khan, A. Gul, O. Mahmoud, M. Miftahuddin, A. Perperoglou, W. Adler, and B. Lausen. An ensemble of optimal trees for class membership probability estimation. In Analysis of Large and Complex Data, pages 395–409. Springer, 2016.

[26] Z. Khan, A. Gul, A. Perperoglou, M. Miftahuddin, O. Mahmoud, W. Adler, and B. Lausen. Ensemble of optimal trees, random forest and random projection ensemble classification. Advances in Data Analysis and Classification, 14(1):97–116, 2020.

[27] B. Efron and R. Tibshirani. Improvements on cross-validation: the 632+ bootstrap method. Journal of the American Statistical Association, 92(438):548–560, 1997.

[28] W. Adler and B. Lausen. Bootstrap estimated true and false positive rates and roc curve. Computational Statistics & Data Analysis, 53(3):718–729, 2009.

[29] B. Efron and R. Tibshirani. An introduction to the bootstrap, volume 57. Chapman & Hall/CRC, 1993.

[30] G. W. Brier. Verification of forecasts expressed in terms of probability. Monthly weather review, 78(1):1–3, 1950.

[31] P. Bühlmann and B. Yu. Analyzing bagging. The Annals of Statistics, 30(4):927–961, 2002.

[32] A. Andreas and W. Stuetzle. The effect of bagging on variance, bias, and mean squared error. Preprint. AT&T Labs-Research, 2000.

[33] P. Latinne, O. Debeir, and C. Decaestecker. Limiting the number of trees in random forests. Multiple Classifier Systems, pages 178–187, 2001.

[34] S. Bernard, L. Heutte, and S. Adam. On the selection of decision trees in random forests. In International Joint Conference on Neural Networks, pages 302–307. IEEE, 2009.

[35] H. B. Li, W. Wang, H. W. Ding, and J. Dong. Trees weighting random forest method for classifying high-dimensional noisy data. In IEEE 7th International Conference on e-Business Engineering (ICEBE), 2010, pages 160–163. IEEE, 2010.

[36] W. Adler, O. Gefeller, A. Gul, F. K. Horn, Z. Khan, and B. Lausen. Ensemble pruning for glaucoma detection in an unbalanced data set. Methods of Information in Medicine, 55(6):557–563, 2016.

[37] T. Oshiro, P. Perez, and J. Baranauskas. How many trees in a random forest? Machine Learning and Data Mining in Pattern Recognition, pages 154–168, 2012.

[38] H. Zhang and M. Wang. Search for the smallest random forest. Statistics and its interface, 2(3):381–388, 2009.

[39] A. Gul, Z. Khan, A. Perperoglou, O. Mahmoud, M. Miftahuddin, W. Adler, and B. Lausen. Ensemble of subset of k-nearest neighbours models for class membership probability estimation. In Analysis of Large and Complex Data, pages 411–421. Springer, 2016.

[40] A. Peters and T. Hothorn. ipred: Improved predictors, 2012. R package version 0.9-1.

[41] K. Bache and M. Lichman. UCI machine learning repository, 2013.

[42] C. Hurley. gclus: Clustering Graphics, 2012. R package version 1.3.1.

[43] N. Chaturvedi M. Lueder J. Goeman, R. Meijer. penalized: Penalized generalized linear models., 2012. Penalized R package, version 0.9-42.

[44] A. Karatzoglou, A. Smola, K. Hornik, and A. Zeileis. kernlab – an S4 package for kernel methods in R. Journal of Statistical Software, 11(9):1–20, 2004.

[45] D. Meyer, E. Dimitriadou, K. Hornik, A. Weingessel, and F. Leisch. e1071: Misc Functions of the Department of Statistics (e1071), TU Wien, 2014. R package version 1.6-4.

[46] W. Adler, A. Peters, and B. Lausen. Comparison of classifiers applied to confocal scanning laser ophthalmoscopy data. Methods of information in medicine, 47(1):38–46, 2008.

[47] N. Meinshausen. Node harvest. The Annals of Applied Statistics, 4(4):2049–2072, 2010.

[48] N. Meinshausen. nodeHarvest: Node Harvest for regression and classification, 2013. R package version 0.6.

[49] T. I. Cannings and R. J. Samworth. Random projection ensemble classification. Journal of Royal Statistical Society, Series B (with discussion) (2017), to appear.

[50] T. I. Cannings and R. J. Samworth. RPEnsemble: Random Projection Ensemble Classification, 2016. R package version 0.3.

[51] R Core Team. R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing, Vienna, Austria, 2020.

[52] Z. Khan, A. Gul, A. Perperoglou, O. Mahmoud, W. Adler, M. Miftahuddin, and B. Lausen. OTE: Optimal Trees Ensembles for Regression, Classification and Class Membership Probability Estimation, 2020. R package version 1.0.1.

[53] R Core Team. R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing, Vienna, Austria, 2020.

[54] O. Mahmoud, A. Harrison, A. Perperoglou, A. Gul, Z. Khan, M. V. Metodiev, and B. Lausen. A feature selection method for classification within functional genomics experiments based on the proportional overlapping score. BMC Bioinformatics, 15(1):274, 2014.

[55] O. Mahmoud, A. Harrison, A. Perperoglou, A. Gul, Z. Khan, and B. Lausen. propOverlap: Feature (gene) selection based on the Proportional Overlapping Scores, 2014. R package version 1.0.

[56] S. Ahmed, K. K. Ghosh, P. K. Singh, Z. W. Geem, and R. Sarkar. Hybrid of harmony search algorithm and ring theory-based evolutionary algorithm for feature selection. IEEE Access, 8:102629–102645, 2020.

[57] D. Rahman Wijaya and F. Afianti. Stability assessment of feature selection algorithms on homogeneous datasets: A study for sensor array optimization problem. IEEE Access, 8:33944–33953, 2020.

[58] Y. Gao, Y. Zhou, and Q. Luo. An efficient binary equilibrium optimizer algorithm for feature selection. IEEE Access, 8:140936–140963, 2020.

[59] A. B. Brahim and M. Limam. Ensemble feature selection for high dimensional data: a new method and a comparative study. Advances in Data Analysis and Classification, pages 1–16, 2017.

[60] J. Liu, S. Tang, G. Xu, C. Ma, and M. Lin. A novel configuration tuning method based on feature selection for hadoop mapreduce. IEEE Access, 8:63862–63871, 2020.

[61] Y. Li and Z. Yang. Application of eos-elm with binary jaya-based feature selection to real-time transient stability assessment using pmu data. IEEE Access, 5:23092–23101, 2017.

[62] G. T. Reddy, M. P. K. Reddy, K. Lakshmanna, R. Kaluri, D. S. Rajput, G. Srivastava, and T. Baker. Analysis of dimensionality reduction techniques on big data. IEEE Access, 8:54776–54788, 2020.

[63] H. Parvin, M. MirnabiBaboli, and H. Alinejad-Rokny. Proposing a classifier ensemble framework based on classifier selection and decision tree. Engineering Applications of Artificial Intelligence, 37:34–42, 2015.

[64] B. Lausen, W. Sauerbrei, and M. Schumacher. Classification and regression trees (CART) used for the exploration of prognostic factors measured on different scales. In Dirschedl, P., & Ostermann, R. (eds.), Computational Statistics, Physica-Verlag, Heidelberg, Germany, pages 483–496. Springer, 1994.

[65] B. Lausen, T. Hothorn, F. Bretz, and M. Schumacher. Assessment of optimal selected prognostic factors. Biometrical Journal: Journal of Mathematical Methods in Biosciences, 46(3):364–374, 2004.

[66] S. Potapov. Improving the split criteria for classification trees and ensemble methods. Phd dissertation, University of Erlangen-Nuremberg, Germany, 2012, urn:nbn:de:bvb:29-opus-36307.