Bagging and Boosting with Equivalent Space Consumption

Rishabh Agarwal
Department of I&CT
Manipal Institute of Technology,
Manipal, Karnataka - 576104, India
rishabh.agarwal@hotmail.com

Satwik Mishra
Department of I&CT
Manipal Institute of Technology,
Manipal, Karnataka - 576104, India
sumumishra@gmail.com

Arjun CV
Department of I&CT
Manipal Institute of Technology,
Manipal, Karnataka - 576104, India
arjun.cv@manipal.edu

Abstract*— There are various bagging techniques of which Random Forests and Extra Trees are among the most popular. Similarly, AdaBoost and Gradient Boosting are the most common boosting techniques. There is a lot of discussion as to which machine is better and why and which one should be used with different experts having different opinions and the choice based mostly on the problem and personal choice. But the question is how to effectively compare them since bagging and boosting machines, although being ensemble methods, construct their component estimators differently. This paper aims to compare their respective accuracy and time requirement when all the four considered machines use the same number of trees, with all of these trees having the same size by applying them on a randomly generated uniformly distributed data set to remove any chance of a particular machine providing better results on some standard data set.

Index Terms— Bagging, Boosting, Equivalent Space, Outliers, Trees

I. INTRODUCTION

There is one fundamental difference in performance between Bagging and Boosting that may force one to choose Bagging over Boosting. Bagging can be easily deployed in a distributed fashion due to the fact that they can run in parallel, whereas Boosting Machines only run trial after trial. Furthermore, it provides better generalization when there are a lot of outliers in the data. So, if we are constrained either by the size of the data or the number of trials, we prefer to go with bagging. [1]

On the other hand, we would prefer Boosting over Bagging if not constrained by the size. The idea of reinforced learning is something which appeals to everyone and is closer to the origin and basics of machine learning. It is like when you are preparing for an examination or an assessment, after one prep test, one would certainly spend some time curbing those mistakes, rather than rush to the next prep test.

For our study, we have considered Random Forests and Extra Trees as the Bagging machines and AdaBoost and Gradient Boosting as the Boosting machines.

II. OBJECTIVE

Both Bagging and Boosting have their own advantages and disadvantages over one another. Each of the machines has its appeal and all of them are ensemble machines but their methods of construction of component estimators are so dissimilar that it is difficult to compare them directly. [2] If we consider AdaBoost with Decision Trees as base estimator, then all of the four considered machines are constructed from Decision Trees, making their comparison more plausible. Yet, the manner differs and the trees formed in each of the machines are different in their construction.

Most scholars compare these techniques primarily on their resulting accuracy but that highly depends on the constraining parameters used in the learning and neither does it reveal the resource utilization properly.

The objective of this paper is to analyze and compare these techniques while overcoming such difficulties by ensuring that all the constructed component estimator Decision Trees have the same or almost similar structure, which causes all of the machines to occupy the same amount of space as all of them use the same number of trees with each of the trees requiring approximately the same space as they have the same construction limits. While in most cases, the ultimate choice depends mainly on the data set and the type of data it contains, this study aims to find the best algorithm in a generic case under the above stated limits.

III. ALGORITHMS

A. Decision Trees

A decision tree is a simple classifier that splits the feature space into many regions. Each region can be visualized as rectangles having constant prediction. The visual representation is shown in Fig. 1, Fig. 2 and Fig. 3. [4]

A tree is built greedily from the root and splitting is done such that the mean square error is reduced. The greediness enables us to build trees quickly but this produces sub optimal results. Building an optimal tree is an NP complete problem. [9]

B. Bagging

In bagging the main motivation is to average unbiased models and the noise in data to create a model that has low variance in terms of classification. The combination of learning models increases the classification accuracy.

Extra trees algorithm follows the classical top down approach to build an ensemble of unpruned decision trees. The striking difference from other tree based ensemble methods are that it chooses the splitting point fully at random and it uses the entire learning sample and not only the bootstrap replica to grow the trees. [6]

Random Forest can be defined as a large collection of decorrelated decision trees, hence the term forest. This final prediction can simply be the mean or average of each prediction. It is a form of meta estimator that fits a number of decision trees on various sub-samples of the data set and use averaging to improve the predictive accuracy and control over-fitting. The sub-sample size is always the same as the original input sample size. [7] Decision trees split using greedy algorithm to reduce the error. So, it is believed that Random Forests are better or an improvement over bagged decision trees. Ensemble method works better on uncorrelated or weakly correlated data. In Random Forests, the learning algorithm is restricted to the sample space, it scans through all

the nodes in order to select the optimal split point. Hence the resulting prediction will have less correlation. [8]

C. Boosting

The basic difference between bagging and boosting is that in boosting every time we randomly select the next sample data set it is dependent on the correctness of the previous prediction model, however in bagging there is simply random selection with replacement and the result is an ensemble of classifiers.

In Adaptive boosting the aim is to fit a sequence of weak learners on different weighted training data. First it gives equal weightage to each predicted observation, if the prediction is not correct using the first estimator then a higher weight is given to that particular observation. This is an iterative process and it continues to add estimators till a limit is reached say in terms of accuracy. We can use any machine learning algorithm as a base learner if it accepts weight on training data set. [5] In Gradient boosting the prediction is done by an ensemble of simple estimators. It is possible to create an ensemble of various estimators theoretically, but in practice gradient boosting over decision trees is used. Given that we know how to train a single decision tree, an ensemble of trees can be created one by one and then the prediction of the individual trees is aggregated. The next decision tree's job is to cover the discrepancy between the target or desired function and the current ensemble prediction by the reconstruction of the residual. Residual means the difference between the target and the current prediction of the ensemble. The next tree in the ensemble should minimize the training error. [4]

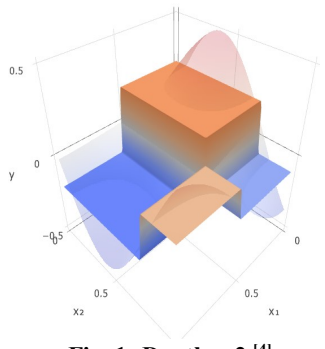

Fig. 1: Depth = 2 [4]

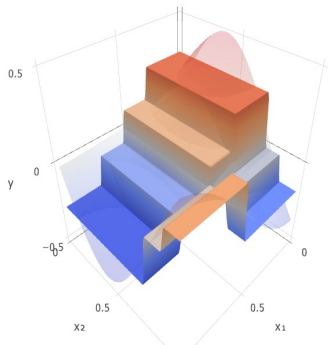

Fig. 2: Depth = 3 [4]

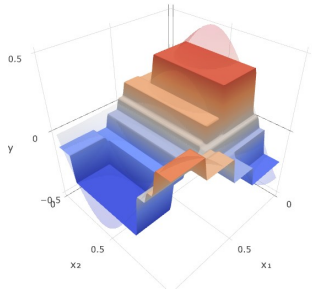

Fig. 3: Depth = 4 [4]

IV. METHODOLOGY

A. Data Sets

In our classification problem, three data sets were used with all of them being entirely randomly generated so as to avoid any existing bias of the performance of one particular machine on a standard data set. Each data set contains more than 2500 samples.

For each iteration n, n ϵ [2,10] on the data set, the data set is divided into 'n' classes, with each class having an equal number of clusters within it. Each cluster is gaussian and located around the vertices of a hypercube in a 2-dimensional subspace, thus establishing some interdependence [3].

Each sample has 20 features, 10 of which are true features holding actual information, 5 are repeated i.e. they are duplicate or highly positively correlated to some other feature and as a result don't provide much extra useful information and the other 5 features are redundant (also known as uninformative features) i.e. their values have no impact on the assignment of the class to which the sample belongs.

The individual values of the informative features of each cluster are drawn independently from a normally distributed environment and linearly combined randomly within the cluster for covariance. For some of the samples, the class labels are interchanged to create outliers. Finally, the samples are shuffled.

All the three data sets are generated the same way, with the only difference being the number of outliers. The first data set has only 1% outliers, while the second one has 25% outliers and the third one with half of them.

B. Ensuring equivalent space consumption

Random Forests, Extra Trees and Gradient Boosting use Decision Trees as base estimators by default. We consider the AdaBoost machine to be constructed from Decision Trees as well. For each individual component Decision Tree estimator, we limit its depth to 5 and the number of features to be considered for splitting to 5. A minimum of 2 samples are to be required to split a node in the tree. Each of the 4 machines shall use the same number of base estimators, assuring that all of them acquire an equivalent amount of space.

C. Procedure

Once a data set has been generated, each of the 4 techniques is applied to it with each of them having 50 base estimators and their respective accuracy and time consumption measured after a 10-fold cross validation. The folds are stratified to ensure proportionate representation of each class in both the training and the test sets. [10] The same thing is repeated for the same data set but with 100 estimators each in all the machines. The whole of the above specified procedure is repeated for all data sets.

D. Model Evaluation

The primary metric to evaluate is the accuracy which is the percentage of predicted class labels matching the true class labels in the data set.

$$\text{accuracy}(y, \hat{y}) = \frac{1}{n_{\text{samples}}} \sum_{i=0}^{n_{\text{samples}}-1} 1(\hat{y}_i = y_i)$$

The other important metric to consider is the total time taken to train the model and then predict and compare with the true class labels.

V. RESULTS AND DISCUSSION

A. Data set with 1% outliers

The figures 4-9 show the results of the operations on the data set containing 1% outliers.

As we can see in Fig. 4 and Fig. 6, as the no of classes increases the time taken by Gradient Boosting increases almost linearly whereas for the other three machines, the time required varies by a very small amount. Random Forest is slightly faster than AdaBoost while Extra Trees is the fastest due to its lack of search of the optimal splitting criterion. Fig. 5 and Fig. 7, which depict accuracy show that the accuracy generally decreases with the increase in the number of classes. However, the accuracy of Gradient Boosting is much higher throughout, followed by Random Forests, then AdaBoost and finally Extra Trees.

In Fig. 8, when comparing the machines with 50 estimators to that of 100 estimators with respect to time, we see that the time consumed in each machine of 100 estimators is approximately double than that of its corresponding machine of 50 estimators, as expected, and which also shows that there are not many instances of early stopping occurring which might produce different results. Fig. 9 compares the accuracy of the machines of 50 estimators and 100 estimators and once again, as expected, the accuracy increases with the number of estimators but only slightly. Yet again, Gradient Boosting significantly outperforms the other machines. But most importantly, if we consider both Fig. 8 and Fig. 9 together, then we notice that although a Gradient Boosting model with 50 estimators takes as much time as other models with 100 estimators when the number of classes are high, it still provides better accuracy than these machines with 100 estimators.

B. Data set with 25% outliers

The figures 10-15 show the results of the operations on the data set containing 1% outliers.

As we can see in figures 10-15, the results are very similar to that of the data set with 1% outliers. Obviously, the accuracy decreases with the increase in the number of outliers. However, Gradient Boosting still is the clear winner. But, the accuracy of AdaBoost which was earlier closer to Random Forests now drops closer to Extra Trees.

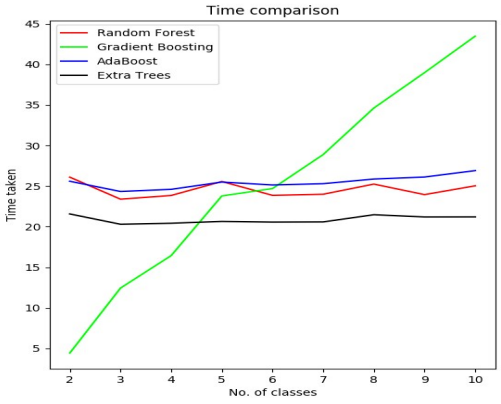

Fig. 4: Comparison with respect to time with 50 estimators with 1% outliers

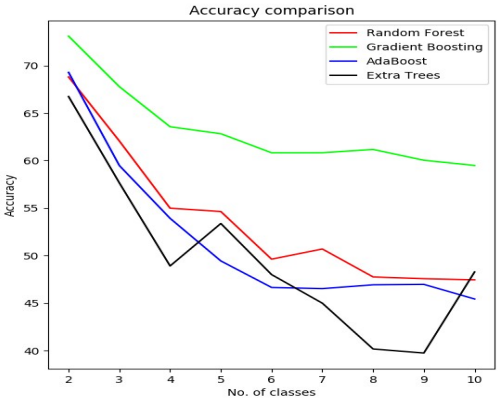

Fig. 5: Comparison with respect to accuracy with 50 estimators with 1% outliers

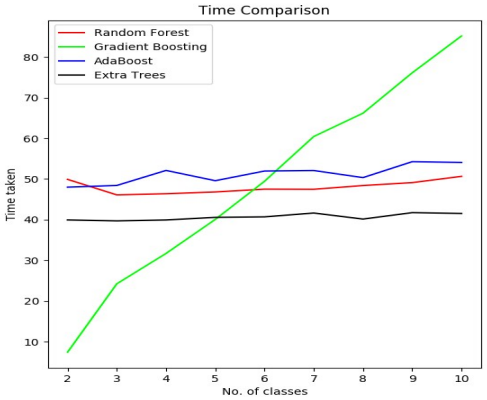

Fig. 6: Comparison with respect to time with 100 estimators with 1% outliers

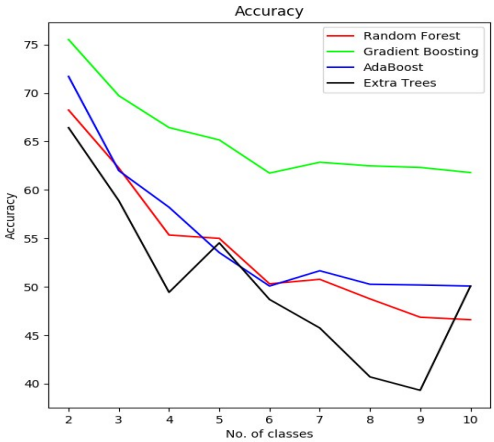

Fig. 7: Comparison with respect to accuracy with 100 estimators with 1% outliers

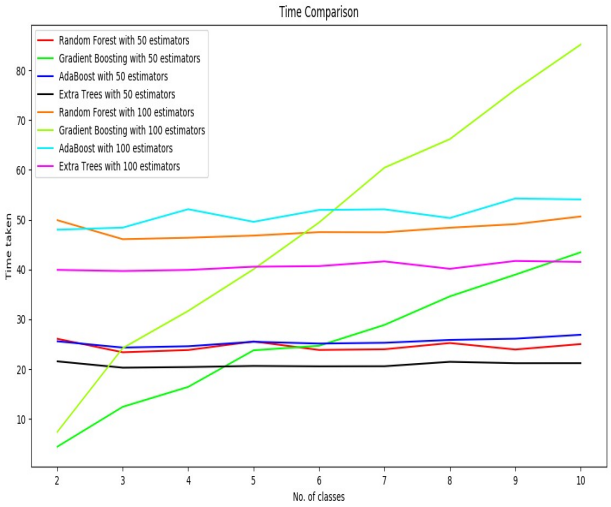

Fig. 8: Comparison with respect to time with 1% outliers

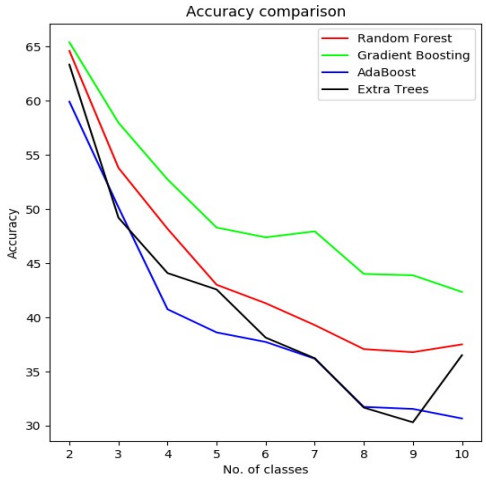

Fig. 11: Comparison with respect to accuracy with 50 estimators with 25% outliers

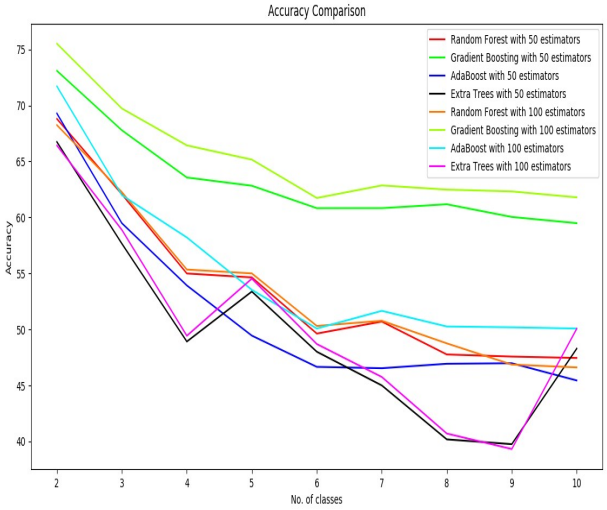

Fig. 9: Comparison with respect to accuracy with 1% outliers

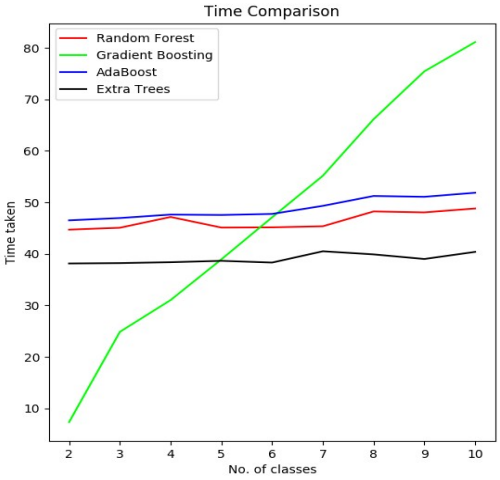

Fig. 12: Comparison with respect to time with 100 estimators with 25% outliers

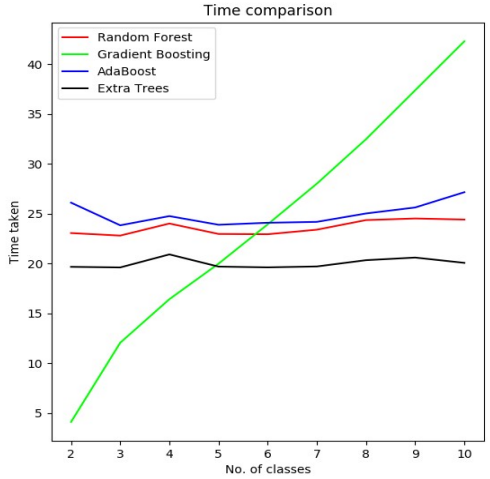

Fig. 10: Comparison with respect to time with 50 estimators with 25% outliers

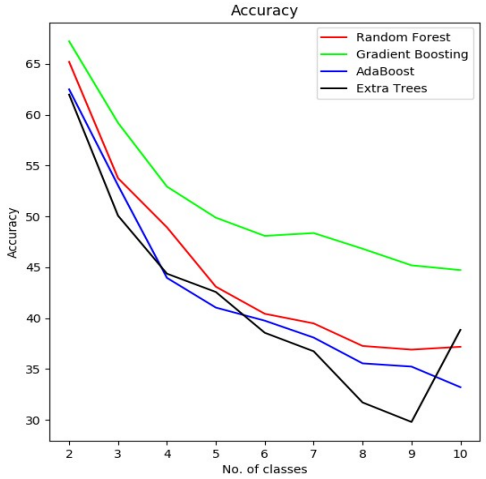

Fig. 13: Comparison with respect to accuracy with 100 estimators with 25% outliers

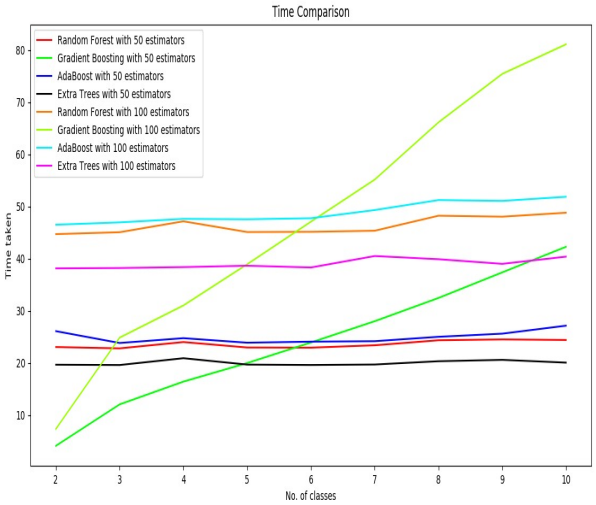

Fig. 14: Comparison with respect to time with 25% outliers

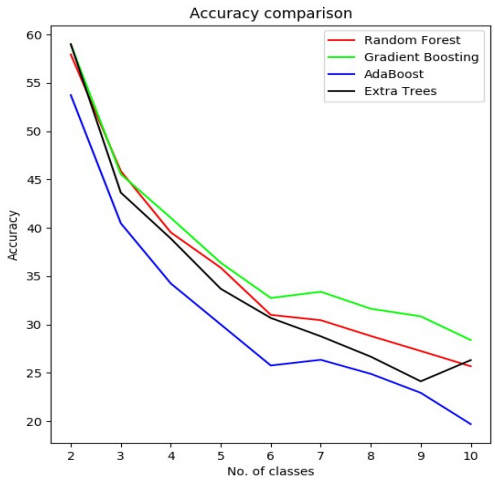

Fig. 17: Comparison with respect to accuracy with 50 estimators with 50% outliers

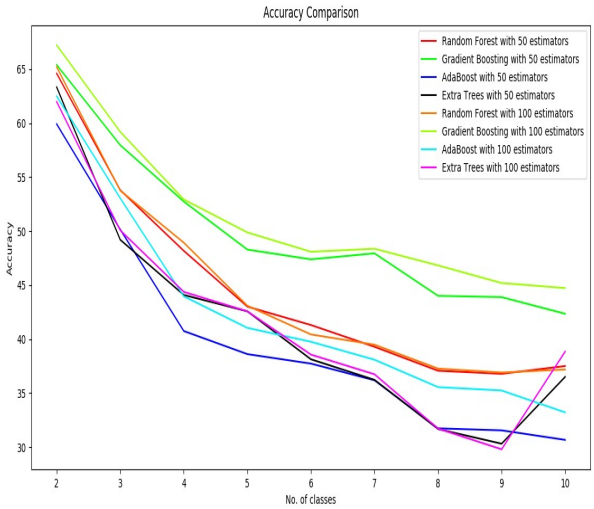

Fig. 15: Comparison with respect to accuracy with 25% outliers

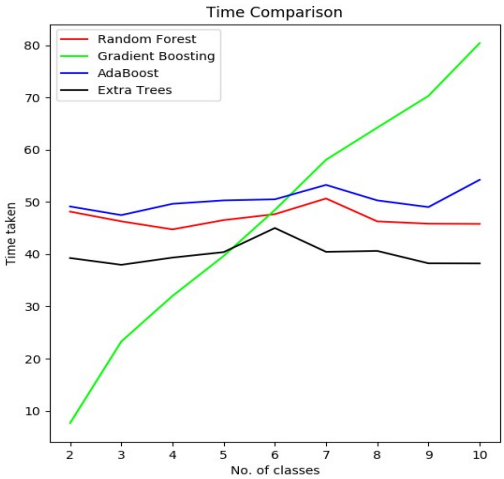

Fig. 18: Comparison with respect to time with 100 estimators with 50% outliers

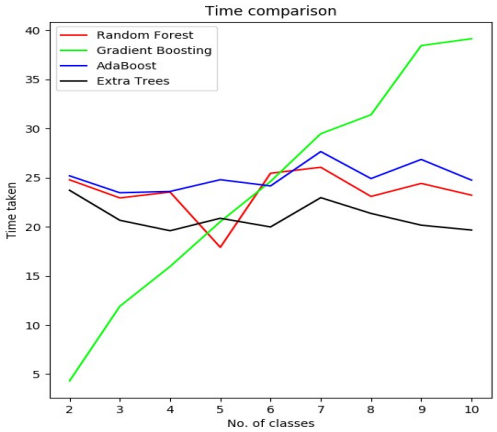

Fig. 16: Comparison with respect to time with 50 estimators with 50% outliers

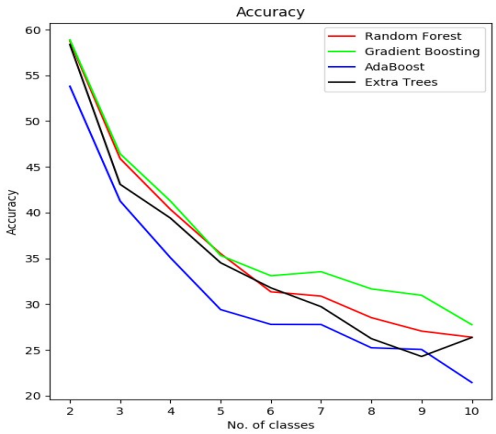

Fig. 19: Comparison with respect to accuracy with 100 estimators with 50% outliers

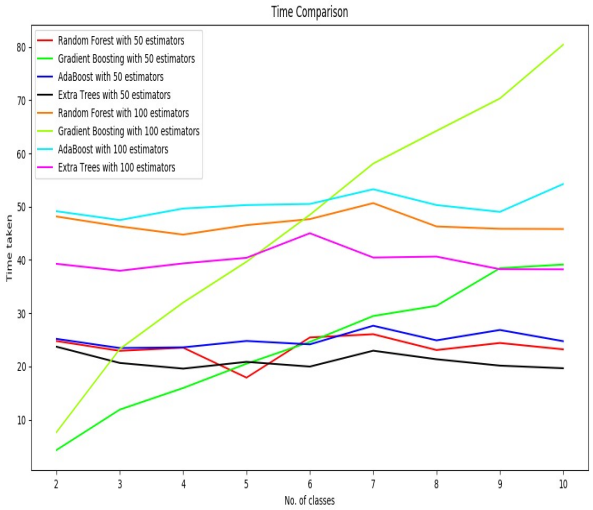

Fig. 20: Comparison with respect to time with 50% outliers

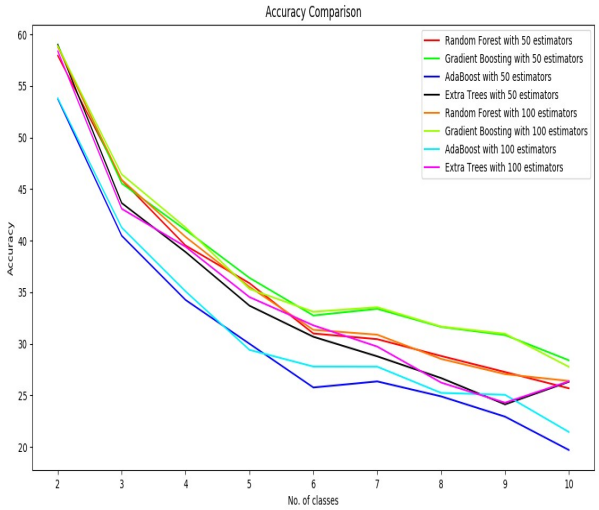

Fig. 21: Comparison with respect to accuracy with 50% outliers

C. Data set with 50% outliers

The figures 16-21 show the results of the operations on the data set containing 1% outliers.

Once again, the results of this scenario are similar to that of the previous two with a high drop in accuracy due to the high presence of outliers. However, AdaBoost's accuracy is now the weakest while Gradient Boosting's accuracy is now much closer to the other machines.

VI. CONCLUSION

This paper presents the results of 4 elite machine learning ensemble methods with all of them occupying the same number and construction of the Decision Trees which are the component base estimators. We restricted the depth and the number of features considered in each tree and applied the algorithms to three different randomly generated data sets of 20 features with more than 2500 samples consisting of different percentage of outliers. The study demonstrates the following trends:

- In a generic case, Gradient Boosting provides better accuracy with less estimators in as much time as other considered models do with twice as many estimators.
- The presence of outliers affects boosting techniques more than bagging techniques.
- As the number of classes increase, the time required by Gradient Boosting increases linearly while that of the other machines vary marginally.
- As the number of classes increase, the accuracy decreases.
- As the number of estimators increase, the time required increases and the accuracy increases.

We conclude that although neither of Boosting and Bagging techniques is clearly better than the other, Gradient Boosting appears to be a preferable choice for implementation in a generic scenario. Although time and accuracy are of paramount importance, we cannot ignore the space requirements either as it indirectly affects the required time.

REFERENCES

[1] Simone Borra, Agostino Di Ciaccio, "Performance evaluation of Bagging and Boosting in nonparametric regression"

[2] David Opitz, Richard Maclin, "Popular Ensemble Methods: An Empirical Study"

[3] http://scikit-learn.org/stable/modules/generated/sklearn.datasets.make_classification.html

[4] Alex Rogozhnikov, http://arogozhnikov.github.io/2016/06/24/gradient_boosting_explained.html

[5] https://www.analyticsvidhya.com/blog/2015/11/quick-introduction-boosting-algorithms-machine-learning/

[6] Pierre Geurts, Damien Ernst, Louis Wehenkel, Extremely randomized trees http://www.montefiore.ulg.ac.be/\~ernst/uploads/news/id63/extremely-randomized-trees.pdf

[7] http://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

[8] Machine Learning, http://machinelearningmastery.com/bagging-and-random-forest-ensemble-algorithms-for-machine-learning/

[9] Ensemble Methods: Foundations and Algorithms, By Zhi-Hua Zhou: Section 1.2.2 Decision Trees

[10] Ron Kohavi, "A Study of Cross Validation and Bootstrap for Accuracy Estimation and Model Selection"