An Empirical Study of Ensemble Techniques (Bagging, Boosting and Stacking)

Conference Paper · March 2019

Rising Odegua
Nossa Data

An Empirical Study of Ensemble Techniques (Bagging, Boosting and Stacking)

Rising O. Odegua *risingodegua@gmail.com*
Department of Computer Science
Ambrose Alli University
Ekpoma, Edo, Nigeria

Abstract

Ensemble methods are popular strategies for improving the predictive ability of a machine learning model. An ensemble consists of a set of individually trained base learners/models whose predictions are combined when classifying new cases. Previous researches have shown that an ensemble is on the average more accurate than a single base model. Bagging, Boosting and Stacking are some popular ensemble techniques which we studied in this paper. We evaluated these ensembles on 9 data sets. From our results, we observed the following. First, an ensemble is always more accurate than a single base model. Secondly, we observed that Boosting ensembles is on the average better than Bagging while Stacking (meta-learning) is on the average more accurate than Boosting and Bagging. Further experiment also shows that the gain in predictive power of any ensembles may sometimes be small or even decrease depending on the data set.

1. Introduction

The technique of combining the predictions of multiple models have long be investigated by researchers (Breiman, 1996c; Clemen, 1989; Wolpert, 1992). This area popularly known as ensemble technique has been demonstrated to outperform a single best model in most tasks. A good ensemble is one where all the individual models are both accurate and diverse in their error.

Consider a supervised learning problem where we are given training example S of the form $\{(x_1, y_1), (x_1, y_1), \dots , (x_n, y_n)\}$ for some function $y = f(x)$. The x values is usually a multidimensional matrix called features while the y values typically drawn from a continuous set of numbers for regression problem and a distinct set of classes for a classification problem is called the target.

Given S of such training example, a learning algorithm h tries to learn the true function f that validates some hypothesis, so that given a new x (test examples), the algorithm can predict the corresponding y values. An ensemble basically looks for the best way to combine individual learning algorithms ($h_1, h_2, \dots , h_L$) to predict new examples typically by weighted or unweighted voting.

The question of whether it is possible in practice to construct good ensembles was addressed by (Dietterich 2000a) who gave three fundamental reasons:

The first being statistical as shown in fig 1(a), where the hypothesis space of a function is too large to explore for limited training data, and that there may exist several hypotheses that gives similar accuracy on the training data.

The second reason is computational as shown in fig 1(b); many algorithm perform some kind of local search that may get stuck in local optima, by running this local search multiple times from different starting points, the combination may provide a better approximation of the unknown true hypothesis.

The third reason is representational as shown in fig 1(c); It is possible to expand the space of representable functions and thus enabling a learning algorithm to form a more accurate approximation of the true unknown hypothesis.

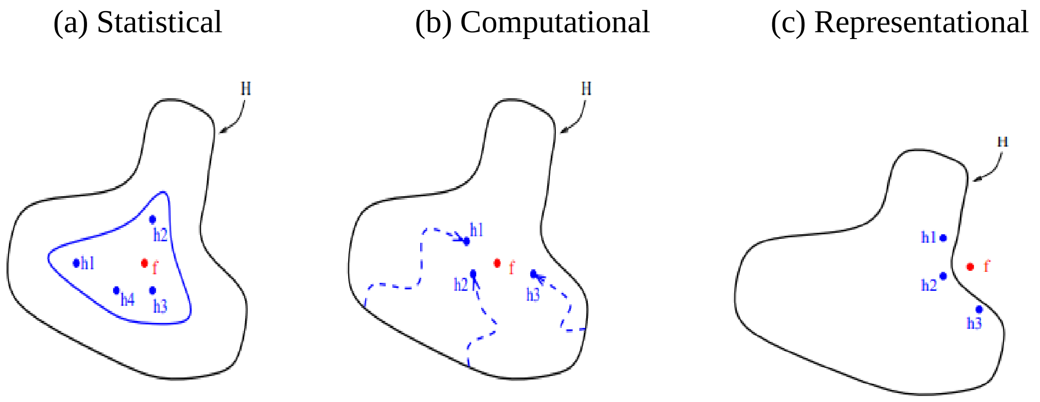


Fig 1.1 Three fundamental reasons while ensembles work: (a) the statistical issue, (b) the computational issue, and (c) he representational issue. The outer curve represents the hypothesis space. (Plot based on figure in [Dietterich, 2000a].)

These three issues are among the most important factors for which the traditional learning approaches fail. A learning algorithm that suffers from the statistical issue is generally said to have a high “variance”, a learning algorithm that suffers from the computational issue can be described as having a high “computational variance”, and a learning algorithm that suffers from the representational issue is generally said to have a high “bias”. Therefore, through combination, the variance as well as the bias of learning algorithms may be reduced; this has been confirmed by many empirical studies [Xu et al., 1992, Bauer and Kohavi, 1999].

This paper is organized as follows. In the next section (2) we present an overview of the popular ensemble techniques bagging, boosting and stacking. Next in section 3, we present an extensive empirical analysis of these three methods, and compare their predictive power side by side on 9 data sets. And finally in section 4 we present future research and additional related work before concluding.

2. Background

2.1. Bagging

Ensemble models are basically generated following two paradigms; sequential ensemble methods and parallel ensemble methods. In sequential ensemble method discussed in section 2.2, the base learners are generated sequentially and Ada-Boost as a popular representative.

In parallel ensemble technique, with Bagging [Breiman, 1996d] as a representative, the basic motivation is to exploit the independent predictive ability of the base learners, since the error can be reduced by combining these independent base learners.

In Bagging, multiple learning algorithms/base learners are trained on a set of training samples say *n* drawn randomly with replacement from the original training set of *m* items. Such sampling is called bootstrap aggregation [Efron and Tibshirani, 1993] (from which the term Bagging was derived). The Bootstrap sampling as shown by Breiman [1996d] gives bagging a great advantage, i.e., it enables the creation of diverse models since each base learner is trained on a sample that contains at least 36.8% original training examples.

Bagging adopts the most popular methods for aggregating the outputs of base learners that is, averaging for regression and voting for classification problems. The Bagging algorithm is shown below in Figure 2.1.

```
Input: Data set D = {(x1, y1),(x2, y2),...,($x_m$, $y_m$)};
       Base learning algorithm E;
       Number of base learners T .
Process:
1. for t = 1,...,T :
2.     $D_{bs}$ = (D, $D_{bs}$) % $D_{bs}$ is the bootstrap distribution
3.     $h_t$ = E($D_{bs}$)
4. end

Output: H(x) = $\sum_{t=1}^{T} h_t(x) = y$)
```

Figure 2.1: The Bagging algorithm

2.2 Boosting

Boosting falls under the sequential ensemble method. It refers to a family of algorithms that are able to convert weak learners into strong learners. Boosting originated from the answer to a question posed by Kearns and Valiant [1989]. That is, whether two complexity classes, weakly learnable and strongly learnable problems are equal. The answer to this question was investigated by Schapire [1990] and found to be positive. This means that it is possible by construction to boost a weak learner to a strong one.

The general idea of boosting is quite simple as explained by Zhou [2012]. Using binary classification as an example, suppose we have a weak learner, and sample training instances *X* i.i.d. from a distribution *D*. Suppose *X* is composed of three parts X_1, X_2 and X_3, each takes 1/3 amount of the distribution. We want to get an accurate classifier on the problem, but we are unlucky and only have a weak classifier at hand, which only has correct classifications in spaces X_1 and X_2 and has wrong classifications in X_3, thus has 1/3 classification error. Let's denote this weak classifier as h_1. It is obvious that h_1 is not desired.

The idea of boosting is to correct the mistakes made by h_1. We can try to derive a new distribution D^* from D, which makes the mistakes of h_1 more evident. Then, we can train another classifier h_2 with D^*. Again, suppose we are unlucky and h_2 is also a weak classifier, which has correct classifications in X_1 and X_3 and has wrong classifications in X_2. By combining h_1 and h_2 in an appropriate way, the combined classifier will have correct classifications in X_1, and maybe some errors in X_2 and X_3. Again, we derive a new distribution D^{**} to make the mistakes of the combined classifier more evident, and train a classifier h_3 from the distribution, so that h_3 has correct classifications in X_2 and X_3. Then, by combining h_1, h_2 and h_3, we have a perfect classifier, since in each space of X_1, X_2 and X_3, at least two classifiers make correct classifications.

```
Input: Sample distribution D;
       Base learning algorithm E;
       Number of learning rounds R .
Process:
1.    D1 = D. % Initialize distribution
2.    for r = 1,...,R :
3.        $h_r = E(D_r)$; % Train a weak learner from distribution $D_r$
4.        $e_r = P_x \sim D_r (h_r(x) = f(x))$;  % Evaluate the error of $h_r$
5.        $D_{r+1}$ = Adjust Distribution($D_r$, $e_r$)
6.    end

Output: H(x) = Combined_Outputs({$h_1(x),...,h_r(x)$})
```

Figure 2.1: Boosting algorithm

In summary, boosting works by training a set of learners sequentially and combining them for prediction, where the latter learners focuses more on the mistakes of the earlier learners. Figure 2.1 shows the boosting algorithm.

2.3 Stacking

Stacking [Wolpert, 1992, Breiman, 1996b, Smyth and Wolpert, 1998] is another important ensemble technique that has been widely used since its inception. It is a technique where a learner usually called the meta-learner/second level learner is trained to efficiently combine the individual learners called the first-level learners.

The basic process of stacking is to train and make predictions on the original training data set using the first-level learners. These predictions are then combined and make up the training data for the meta-learner. I.e. The output of the first level learners serves as input for the meta-learner. The first level learners are often made up of different and diverse learning algorithms although it is possible to create stacked ensembles from the same learning algorithms. The general algorithm for the stacked ensemble is shown in Figure 2.2

Stacking is mostly classified under combination method in the literature. Since the training data is used to generate new data for the mea-learner, if the exact data is used, there will be a high risk of overfitting. Hence, it is recommended that new instances created from the training data are excluded from the original data set. It is also advisable to use a good cross-validation strategy or leave-one-out procedure.

Experiments have shown that stacked ensemble works quite well and on the average better than a single best classifier. Breiman [1996b] demonstrated this using linear regression models with different variables as first-level learners, and least-square linear regression models as the meta-learner. For classification problem, Wolpert [1992] showed that it is important to consider the types of features for the new training data, and the types of learning algorithms for the meta-learner. Ting and Witten [1999] recommended to use class probabilities instead of class labels as

```
Input: Data set $D = \{(x_1, y_1),(x_2, y_2),...,(x_m, y_m)\}$;
       First-level learning algorithms $L_1,..., L_T$ ;
       Second-level learning algorithm L.
Process:
1. for t = 1,...,T :    % Train a first-level learner by applying the
2.     $h_t = L_t(D)$;      % first-level learning algorithm Lt
3. end
4. $D^* = \emptyset$;              % Generate a new data set
5. for i = 1,...,m:
6.     for t = 1,...,T :
7.         $z_{it} = h_t(x_i)$;
8.     end
9.     $D^* = D^* \cup ((z_{i1},...,z_{iT}), y_i)$;
10. end
11. $h^* = L(D^*)$;      % Train the second-level learner $h^*$ by applying
                      % the second-level learning algorithm L to the
                      % new data set $D^*$.

Output: $H(x) = h^*(h_1(x),...,h_T(x))$
```

Figure 2.2: The general stacking ensemble algorithm

features for the new data, since it makes it possible to take into account not only the predictions but also the confidence of the first level learners.

3. Experiment

This section describes our empirical study of Bagging, Boosting, and stacking ensemble techniques. Each of these techniques is trained on 9 different datasets using various learning algorithms. The experiment was carried out using the popular Sci-Kit Learn package in Python. We used Random Forest, Extra Trees, Linear Regression and K-Nearest Neighbors as single base models. Ada-Boost, GradientBoosting, LigthGBM for boosting. And finally for stacking, we combined multiple algorithms such as Linear regression, KNNs, Random Forests, Extra trees.

3.1 Data Sets

To evaluate the performance of the ensembles, we obtained 9 data sets from UCL data set repository, Kaggle and Zindi. Table 1 outlines the characteristics of our data sets. The data sets chosen vary in sizes, dimensions and features.

S/N	Data sets	Instances	Total Features	Features Numerical	Categorical	Size of features after encoding
1	German Bank Credit	1000	21	12	9	63
2	Automobile Pricing	195	24	14	10	67
3	Avocado	18249	11	9	2	66
4	Real Estate	414	5	5	-	-
5	Loan default DSN	4368	30	12	18	130
6	House Pricing	1458	80	26	54	221
7	Retweets pred data	89309	19	8	11	95
8	Traffic prediction kenya	51645	10	2	8	69
9	Chukwudi Supermarkets	4990	12	4	8	33

Table 1: Summary of the datasets used in this paper

3.2 Methodology

First, we performed basic data preparation and cleaning. These included taking care of missing values, and encoding categorical features. We avoided extensive feature engineering on the data sets.

Secondly, we trained 4 single base models (Linear Regression, Random Forest, Extra Trees, KNN) on our data sets and measured their mean absolute error. We used a 10- fold cross validation strategy.

Thirdly, we performed bagging using each of the single base models listed above and also measured their errors.

Fourthly, we used 3 boosting implementations; the vanilla Ada-Boost and modern techniques like GradientBoosting and LigthGBM. And finally, we performed stacking using Random

Forest, Extra Trees, KNN, Extra Trees, GradientBoosting and LightGBM as first level learners and LightGBM as the meta-learner.

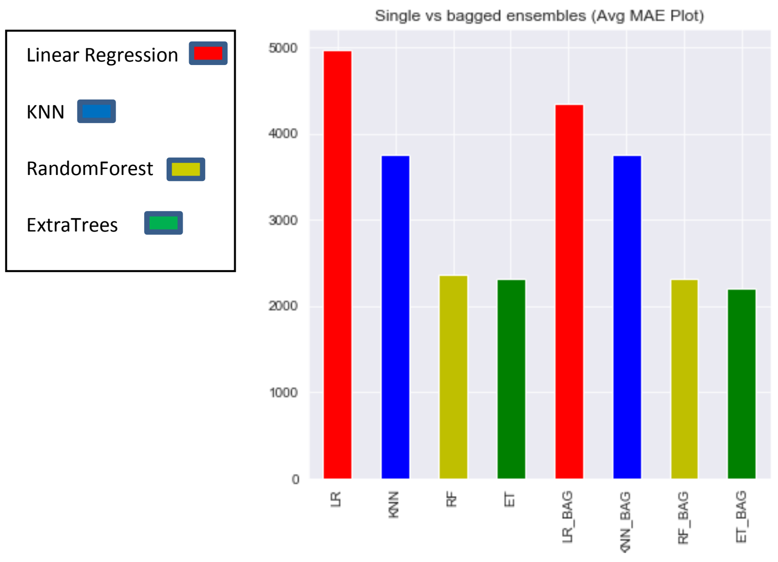
To avoid a bias result, we used the same hyperparameters for all learners.

3.3 Result

The result of our experiments is summarized in Table II and Fig 3,4,5,6. For all the datasets, the mean absolute error was lower for Bagging, Boosting and Stacking than for a single base model. It can also be noticed that Boosting on the average does better than Bagging and Stacking better than both of these. It is worth mentioning that these gain are mostly little depending on the data set.

	Single Base Models				Bagging				Boosting			Stacking
DATASETS	**LR**	**KNN**	**RF**	**ET**	**LR_BAG**	**KNN_BAG**	**RF_BAG**	**ET_BAG**	**GB**	**ADAB**	**LGB**	**STACKED MODEL**
Loan_Data	0.321	0.324	0.328	0.328	0.319	0.314	0.311	0.32	0.303	0.355	0.293	0.3
House_Pricing	2.457	0.164	0.097	0.089	0.082	0.161	0.094	0.084	0.08	0.102	0.079	0.079
Tweets_Data	6.503	5.706	6.133	6.124	6.4	5.787	6.118	6.102	5.822	20.937	5.768	5.611
Traffic_Data	15.5	4.833	4.367	4.393	15.13	4.35	4.32	4.251	4.104	4.147	4.059	4.027
Avocado_Data	0.145	0.227	0.128	0.137	0.145	0.225	0.122	0.13	0.123	0.21	0.115	0.132
Real_Estate	5.654	5.31	4.768	4.571	5.611	5.258	4.315	4.314	4.558	5.429	4.62	4.277
AutoMobile	4848.5	3649.4	2253.3	2199.3	4234.3	3641.1	2217.8	2101.1	2184.8	2209.5	2199.9	2101.0
German_Cred	7.55	10.544	7.36	8.004	7.617	10.256	7.181	7.225	7.725	8.531	7.854	7.276
Supermarket_Sale	75.732	81.908	80.974	86.29	75.627	80.93	79.418	81.093	74.802	75.16	74.47	71.458

Table II shows the mean absolute errors of all single and ensemble models on our data sets



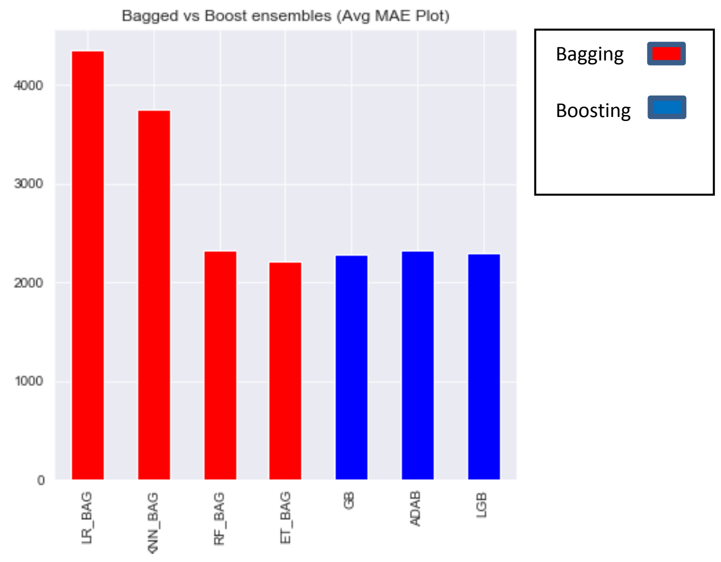
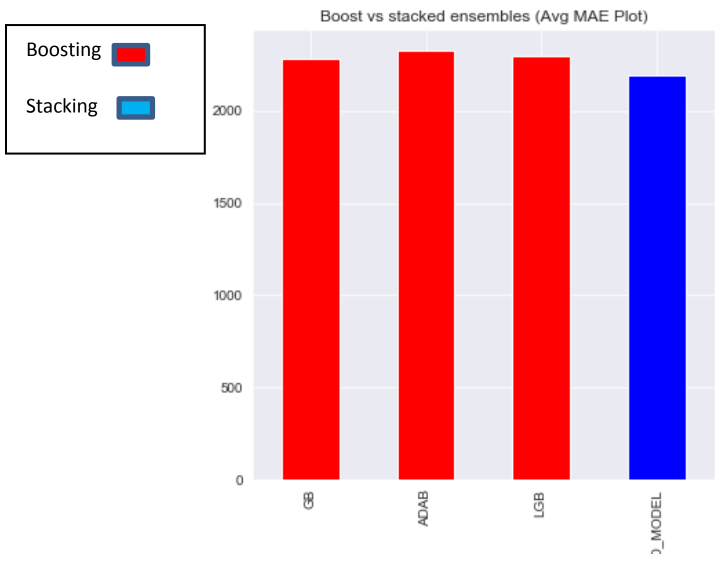


Fig.3 Comparison of Single vs Bagging Ensemble Fig.4 Comparison of Bagging vs Boosting Ensembles



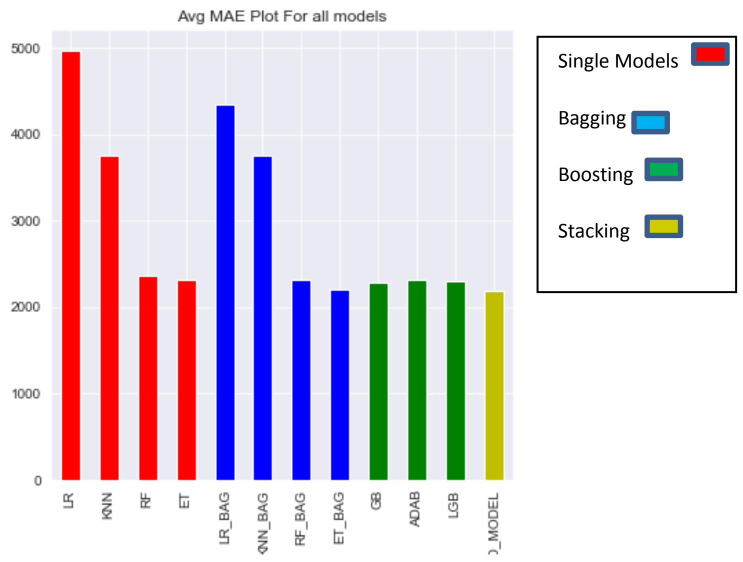


Fig.5 Comparison of Boosting vs Stacking Ensemble Fig.6 Comparison of all models

4. Future Work

One interesting area we plan to investigate is the effect of the meta-learner on stacked ensembles. The important question we ask here is whether a stacked ensemble will perform better if we use a neural network, complex boosting algorithms or a modern implementation of bagging as the meta-learner.

5. Conclusion

This paper presented a comprehensive empirical evaluation of Bagging, Boosting and Stacking ensembles. Our results demonstrated that a Bagging ensemble nearly always outperforms a single classifier. Our results also showed that Boosting ensemble on the average will outperform both Bagging and a single classifier. And finally we showed that Stacking ensemble nearly always outperform Boosting, Bagging and a single base classifier. However, for some data sets Boosting and Stacking may show zero gain or even a decrease in performance from a single classifier. Further research also indicated that Boosting may suffer from overfitting in the presence of noise which may explain some of the decrease in performance for Boosting.

References

Breiman, L. (1996a). Bagging predictors. Machine Learning, 24 (2), 123–140.

Breiman, L. (1996c). Stacked regressions. Machine Learning, 24 (1), 49–64.

Dietterich, T. G. Approximate statistical tests for comparing supervised classification learning algorithms. Neural Computation, 10(7):1895– 1923, 1998.

Dietterich, T. G. Ensemble methods in machine learning. In Proceedings of the 1st Workshop on Multiple Classifier Systems, pages 1–15, Sardinia, Italy, 2000a.

Dietterich, T. G. An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization. Machine Learning, 40(2):139–157, 2000b.

Efron, B., & Tibshirani, R. (1993). An Introduction to the Bootstrap. Chapman and Hall, NY

Freund, Y., & Schapire, R. (1996). Experiments with a new boosting algorithm. In Proceedings of the Thirteenth International Conference on Machine Learning, pp. 148–156 Bari, Italy

J. Nikita and S.Shweta, “Improving Classification Accuracy Using Ensemble Learning Technique (Using Different Decision Trees)”, International Journal of Computer Science,Vol.3, Issue 5, pp. 105 – 727, 732

Wolpert, D. (1992). Stacked generalization. Neural Networks, 5, 241–259.

Y. Freund. Boosting a weak learning algorithm by majority. Information and Computation, 121(2):256–285, 1995.

Y. Freund. An adaptive version of the boost by majority algorithm. Machine Learning, 43(3):293–318, 2001.

Z.-H. Zhou and M. Li. Ensemble Methods. Foundations and Algorithms, -13: 978-1-4398-3005 -5, 2012.