A comparison of Random Forest and Bagging for different numbers of trees

Study carried out by the Data Science Practice

Special thanks to Elijo BYLYKBASHI

canopee

empowering *ecosystem*

www.canopee-group.com

Summary

Introduction

Expecting combined classifiers to perform better than a single classifier is an important guideline of Machine Learning. Therefore, we can expect a better classification performance when we have numerous decision trees from the same data set, as opposed to having only one decision tree. For instance, the authors in [1] show empirically that a random forest composed of 100 trees yields a signicative (although small) improvement in Accuracy. In fact, techniques such as Random Forests and Bagging, base their predictions on the combination of the outcomes of all the decision trees that compose them.

These two algorithms rely heavily on the decision tree technique. For more information on the decision tree algorithm, you can look at [2]. These algorithms (Random Forest and Bagging) draw with replacement n elements from the original data set. They repeat the process k times to create k different (yet very similar) samples. Afterwards, for every sample a decision tree is created. However, there are some differences (which we will explain in this note) in the way these trees get generated by each algorithm.

The Random Forest algorithm was initially designed as an alternative to Bagging in order to achieve better results. Nonetheless, there's no theory which guarantees such result. While the Random Forest might be expected to perform better than Bagging, there might be data sets whose structure is better fitted for the bagging algorithm. The expected superiority of the Random Forest (and Bagging) on decision trees is also an empirical result. For instance, in [3] a modified version of the decision trees outperforms the Random Forest.

The objective of this note is to benchmark the performance (in terms of accuracy) of these two algorithms, for different numbers of trees and for a specific type of datasets. A corrected version of the student test (see [2] p. 159) is then used to test whether the mean of the performance gaps is significant or not.

More specifically, the 8 data sets we consider in this study are of a small size (inferior to 1000 observations). Secondly, they have less than 10 explanatory variables. Finally, the explained variable in all these data sets takes only two values (ex. 0 and 1).

We conclude that despite the number of trees used to develop our algorithms, there are no significant differences in terms of Accuracy between the two algorithms. In the following section we present and explain these two algorithms. In section 3 we talk about the train/test split and cross-validation. In section 4 we display our methodology and the related research. In section 5 we present our results. In the last section we state the conclusions of our research.

1 Explanation of the algorithms

1.1 The Bagging method

Suppose you have a sample of n elements $(E_1, ..., E_n)$, where each element E_i for $i \in 1, \ldots, n$ has the following structure. Furthermore, $X_{i,j} \in R$ for $j \in 1, \ldots, k$ and $i \in 1, \ldots, n$. Whereas, $Y_i \in 0, 1, \ldots l$ for $i \in 1, \ldots, n$, such variable represents the number of our classes/labels.

When we perform the bagging algorithm, we need to determine the number of decision trees. Such number is a hyper parameter of the Bagging algorithm and can only be determined through the process of parameter tuning. That is, you need to try different values until you find the most performing one. This process is called hyper parameter search and there are techniques advising how to perform it efficiently. These techniques aren't the goal of this note so we won't talk about them. For a general overview of these techniques, their advantages and downfalls,see [4].

Now let's explain the algorithm. Suppose that we decided to have T trees. The bagging algorithm consists in repeating the following process T times:

1. We extract with replacement n elements from our original sample of size n.
2. We build a decision tree algorithm given the sample at step 1.

Obviously, at the end of such a process we have T decision trees. Now, suppose you have an element. Suppose furthermore that we know the value of its explanatory variables but we don't know the value of its explained variable and we aim at predicting it using the Bagging algorithm. To do so, the bagging algorithm proceeds as follows:

1. For each of the T trees predict a class/label for the element.
2. Look at all the classes predicted by the T trees and choose the modal class.

1.2 The Random Forest method

The samples of the bagging method tend to be very similar, for obvious reasons. Furthermore the variables on which every tree does the cuts tend to be very similar. Therefore the trees composing the Bagging algorithm are highly correlated. Such correlation affects our predictions negatively. The random forest algorithm comes as a potential solution to avoid such problem. More specifically, at each cut of each tree, the random forest algorithm limits the number of features (variables) where the cut can happen. Let's look at the above example, where each E element has k features among which we can do the cut. Every time the algorithm has to perform a cut in a tree, it randomly determines p features (p<k, usually p is the square root of k) among which the cut should happen. The trees obtained with such a method don't suffer from correlation. Moreover Breiman proves theoretically in his paper

that even when we increase the number of trees to infinity, the random forest algorithm doesn't suffer from over-fitting. For more information see [5].

2 The Train-Test Split/Cross-Validation

When required to build an algorithm on a given data set, the first thing we do is the random division of the data set elements into the training part and the testing part. More precisely, the training data set is used to build the algorithm whereas the testing set is used to analyze how well our algorithm performs with data it has never seen before. Both of these sets are fundamental when assessing the quality of an algorithm for a specific task. There is no specific ratio for the train/test division. Nonetheless an advisable one would be 70% for the training data set and 30% for the testing one. 60%-40% is another plausible combination. The Train-Test Split technique is recommended when the size of your data-set is relatively big (ex. superior to 1000) is advised to use when the size of your data-set is relatively big (ex. Superior to 1000).

For smaller sample sizes another technique called cross-validation can be used instead of the Train-Test Split. The most used cross-validation forms are the 5-fold and 10-fold cross-validations. Let's explain the 5-fold cross-validation. Such technique consists in randomly dividing your data into 5 groups. Group 1 Data is to be used as a testing set. Groups 2-5 are used as the training set. The same process is repeated 5 times with every time using a different (and never used before) group as a testing set and the rest of the groups as training sets. In the end you average the performance of the 5 testing sets in order to evaluate the performance of the algorithm with your data.

3 Methodology and Related Research

We took 8 data sets from the sites [6] and [7]. All these data sets had a small sample (less than 1000 observations), less than 10 explanatory variables and an explained variable taking only two possible values (ex. 0 and 1). More specifically, the datasets were: Blood Transfusion ([6]), Diabetes ([6]), Immunotherapy ([7], [8], [9]), Cryotherapy ([7], [8], [9]), Habberman ([6]), visualizing_environmental ([6]), fri_c3_100_5 ([6]) and hayes-roth([6]). For each of these datasets, we looked at the performance of the Bagging algorithm and the Random Forest algorithm for different numbers of trees. More precisely the numbers of trees that we analyzed were 2, 5, 10, 20, 100, 200, 500, 1000, 4000.

Since the size of the data sets was small, we used the cross-validation to measure the performance of our algorithm in terms of accuracy. Moreover, instead of using a simple cross-validation we used the repeated cross-validation. More specifically we used the 10-fold cross validation repeated 5 times. For every iteration, we randomly split the data into 10 groups. Then we implemented the 10-fold cross validation technique. After 5 iterations, this technique (i.e. repeating the 10-fold cross-validation 5 times) gave us 50 different results in total. In the end we averaged these 50 results to evaluate the performance of each algorithm. Many sources advise the use of repeated cross validation instead of the simple cross validation (see [10] and [11]). Furthermore the authors in [10] assert that the evaluation and selection of models requires this technique. Their study focuses on a specific type of data set (QSAR datasets). On the other hand authors in [12] state that while the number of repeated cross validations grows, the confidence intervals of accuracy tend to be narrower. Nonetheless, the true value (true Accuracy) is outside of such interval. They use different repetition numbers like 10, 30 etc… with 10 being the smallest. We chose 5 as a number which would give us a sufficiently large sample of data for the hypothesis tests and more robust results (than the 10 times repeated cross-validation).

In order to test whether the mean difference in terms of accuracy is significant we use a corrected version of the student test. For more information you can see ([2] p.159) at the 95% level of confidence. More specifically for a given tree number and a dataset we'll obtain 50 Accuracy results for the Random Forest algorithm and 50 Accuracy results for the Bagging algorithm. Afterwards we do the difference between the respective accuracies and we obtain a sample of 50 Accuracy differences. We calculate the mean of these differences. The Corrected Student Test consists in analyzing whether the mean of these differences is significantly different from zero or not.

Such comparison between the two algorithms is novel to the best of our knowledge. Research papers such as [13], [14], [3], [15], [16], [17], [18] and [19] treat problems which are connected to our research. For instance, authors in [19] concluded that the efficient number of trees for random forests in terms of the Area Under Curve measure (a performance measure different from Accuracy) resides between 64 and 128 trees. Furthermore, [16] is considered as one of the most classical papers of algorithm comparison. Nevertheless, the paper being relatively old it doesn't analyze algorithms such as Bagging and Random Forests. Authors in [14] and [3] do a similar comparison of the algorithms as the authors in [16]. Paper [15] can be seen as one of the most complete papers of algorithm comparison. In fact it compares 179 classifiers from 121 datasets. Authors in [18] follow a similar methodology compared to the above papers. The novelty of this paper resides in the comparison of algorithms once noise has been injected in the data.

Finally, paper [13] can be seen as an improvement of paper [1]. As a matter of fact, the authors compare the algorithms of Bagging and Random Forest for a fixed number of trees (1000 trees). And they use the Student test to check whether the differences (of the Accuracy scores obtained after cross-validation) are significant. While this aspect of their research is similar to ours there are some fundamental differences between our methodologies. First, we use different datasets which have a very specific structure (size inferior to 1000, no more than 10 explanatory variables and explained variable takes only two values). Secondly, instead of using simple

cross-validation we use repeated cross-validation which gives us a larger sample of data (50 instead of 10) which is more convenient to perform Student-like tests. Thirdly, we use a corrected version of the Student test which avoids the "artificial increase" of the test statistic as a result of the large sample. By doing so, we avoid falsly rejecting H0 (mean difference of Accuracies between the two algorithms = 0) as a result of the increase of the sample size. Futhermore, the confidence level for this test is 95% instead of 99%, which proves to be a stronger proof given the results we get later (If we don't reject H0 at the 95% confidence level we're not going to reject it at the 99% confidence level but not vice-versa). Forthly, and most importantly, we compare the results (the average Accuracy) between the two algorithms for different numbers of trees (and not for a single number of trees).

4 Results

In this section we show and compare the performances of the random forest algorithm and the bagging algorithm throughout all the datasets and for different numbers of trees, in terms of Accuracy. The last row of each table shows whether the differences in terms of performance (mean Accuracy) between the two algorithms are significant or not after applying the corrected version of the Student Test (for more information on this test see [2] p.159).

Note: The Accuracy values are expressed in %

Table 1: Habberman ([6]) dataset —> 3 explanatory variables and 306 instances

Algorithm/Trees	2	5	10	20	100	200	500	1000	4000
Random Forest	72.7	73.33	74.2	73.6	74.05	73.85	73.86	74.22	73.36
Bagging	73.52	72.43	73.4	73.76	74.18	73.19	73.4	73.9	73.93
Significant	No	No	No	No	No	No	No	No	No

Table 2: visualizing_environmental ([6]) dataset —> 3 explanatory variables and 111 instances

Algorithm/Trees	2	5	10	20	100	200	500	1000	4000
Random Forest	64.89	66.83	64.52	65.92	66.42	67.12	67.32	65.94	67.94
Bagging	64.58	64.82	67.48	66.68	66.89	66.91	66.7	67.98	68.97
Significant	No	No	No	No	No	No	No	No	No

Table 3: fri_c3_100_5 ([6]) dataset —> 5 explanatory variables and 100 instances

Algorithm/Trees	2	5	10	20	100	200	500	1000	4000
Random Forest	71.2	72.6	76	76.2	77.2	75.8	78.4	78.2	77.2
Bagging	70.2	74.4	75.2	76.8	77.6	74.2	77.4	77.8	75.8
Significant	No	No	No	No	No	No	No	No	No

Table 4: hayes-roth ([6]) dataset —> 4 explanatory variables and 132 instances

Algorithm/Trees	2	5	10	20	100	200	500	1000	4000
Random Forest	75.19	75.46	80.78	82.21	83.6	84.22	82.74	83.33	83.79
Bagging	76.92	79.27	81.79	83.52	84.05	84.19	84.14	84.12	83.48
Significant	No	No	No	No	No	No	No	No	No

Table 5: Blood Transfusion([6]) dataset —> 4 explanatory variables and 748 instances

Algorithm/Trees	2	5	10	20	100	200	500	1000	4000
Random Forest	76.85	77.89	77.51	77.32	77.38	77.94	77.06	77.75	77.91
Bagging	78.77	78.4	78.29	78.32	78.69	78.66	78.46	78.55	78.63
Significant	No	No	No	No	No	No	No	No	No

Table 6: Diabetes ([6]) dataset —> 8 explanatory variables and 768 instances

Algorithm/Trees	2	5	10	20	100	200	500	1000	4000
Random Forest	72.89	74.86	75.61	75.59	75.8	75.62	75.67	75.75	75.3
Bagging	75.2	75.9	75.72	75.82	75.64	76.2	75.62	75.8	75.75
Significant	No	No	No	No	No	No	No	No	No

Table 7: Immunotherapy ([15,18,19]) dataset —> 7 explanatory variables and 90 instances

Algorithm/Trees	2	5	10	20	100	200	500	1000	4000
Random Forest	80.89	81.56	83.33	84	83.56	84	83.78	84	83.56
Bagging	82.89	82.67	84.67	84.89	84.22	83.78	83.33	82.89	82.89
Significant	No	No	No	No	No	No	No	No	No

Table 8: Cryotherapy ([15,18,19]) dataset: —> 6 explanatory variables and 90 instances

Algorithm/Trees	2	5	10	20	100	200	500	1000	4000
Random Forest	82.88	85.78	86.67	86	89.56	88.22	90	90.89	91.33
Bagging	83.33	84.44	84.89	85.11	85.11	85.11	85.33	84.22	83.78
Significant	No	No	No	No	No	No	No	No	No

Conclusion

The superiority of an algorithm [13] compared to another depends on numerous factors such as the structure of the algorithm and also the dataset at hand. In this paper we conclude that there aren't significant differences between the algorithms of Bagging and Random Forest in terms of Accuracy, for every number of trees and data set that we use (as long as the data sets fulfill the conditions specified at the beginning of our paper). Therefore, it could be more efficient to use one of these two algorithms. Like many other conclusions in machine learning, our results remain empirical and they can be further consolidated by doing a similar study with other data sets fulfilling the same conditions as the data sets in this paper.

References

[1] O.H. et al. Lawrence. Comparing pure parallel ensemble creation techniques against bagging., 2003.

[2] I.H. et al. Witten. Data mining: Practical machine learning tools and techniques, third edition., 2011.

[3] Warda M. El-Talbany, M.E. An empirical comparison of tree-based learning algorithms: An egyptian rice diseases classification case study., 2016.

[4] De Moor B. Claesen M. Hyperparameter search in machine learning, 2015.

[5] L. Breiman. Random forests, 2001.

[6] Open ml.

[7] R.E. et al Banfield. A comparison of ensemble creation techniques, 2004.

[8] F. et al Khozeimeh. An expert system for selecting wart treatment method, 2017.

[9] F. et al Khozeimeh. Intralesional immunotherapy compared to cryotherapy in the treatment of warts., 2017.

[10] D. et al. Krstajic. Cross-validation pitfalls when selecting and assessing regression and classification models., 2014.

[11] Uci machine learning repository.

[12] Blockeel H. Vanwinckelen G. On estimating model accuracy with repeated cross-validation., 2011.

[13] R.E. et al Banfield. A comparison of ensemble creation techniques, 2004.

[14] Niculescu-Mizil A Caruana R. An empirical comparison of supervised learning algorithms using different performance metrics., 2006.

[15] M. et al Fernández-Delgado. Do we need hundreds of classifiers to solve real world problems?, 2014.

[16] R.D et al. King. Statlog: Comparison of classification algorithms on large real-world problems., 1995.

[17] P. et al. Latinne. Limiting the number of trees in random forests., 2001.

[18] R.E. et al Banfield. An empirical comparison of decision trees and other classification methods, 1998.

[19] T.M. et al. Oshiro. How many trees in a random forest, 2012.

Nous sommes Canopee, un cabinet de conseil indépendant, spécialiste en Finance, DATA et transformation digitale.

Nous sommes l'*empowering ecosystem* ! Un écosystème en perpétuel mouvement.
Un écosystème qui donne le pouvoir à nos collaborateurs, nos projets et nos clients de se nourrir de l'émulation collective pour lui donner de la force.

Depuis 2009, nous intervenons dans les secteurs de la BFI de l'Asset Management, la banque privée ou de détail, les services financiers et l'assurance et cela auprès de clients tels que SG, HSBC,BNP Paribas ou encore ALLIANZ GI et AXA IM. Des écosystèmes réglementaires et spécifiques qui ont été nos premiers terrains de jeu.
C'est ici que nous avons su développer notre agilité, nos compétences, notre sens de l'engagement. Aujourd'hui, nous grandissons et continuons à nous investir et nous engager sur des écosystèmes spécifiques, techniques, mais diversifiés tels que l'industrie, le retail ou encore la pharma.
Toujours autour de nos 3 expertises que sont la finance, la data et la transformation digitale.

canopee
empowering *ecosystem*

www.canopee-group.com