Analysis on the Applicability of the Random Forest

Tongtian Zhu

Department of Mathematics, China University of mining and technology (Beijing), Beijing, 100089, China

Corresponding author’s e-mail: angela@cas-harbour.org

Abstract. Random forest is a flexible algorithm with a wide range of applications and performs well on a large number of data sets. Besides, Random forest is immune to statistical assumptions as well as preprocessing burden and can handle a large data set with high dimensionality and missing values. Nevertheless, random forest struggles with high-cardinality categorical variables, unbalanced data, time series forecasting, variables interpretation, and is sensitive to hyperparameter. Thus, random forest is relatively suitable for processing high-dimensional data and data with missing variables. Besides, random forest works well with a large amount of data, which is previously unprocessed. Moreover, random forest is an appropriate method, when there are prior statistical assumptions. However, random forest is non-ideal, when processing data with endogenous temporal effects or high-cardinality categorical variables, as well as when the interpretation is the primary goal. Despite the shortcomings of the random forest, there are still some improvements that can be made. It will be more convenient for users to screen methods, if there is a rating system to give an overall score towards all alternative algorithms depending on the input data and the users’ goals.

1. Introduction

Random forest was first introduced by Leo Breiman [2] and is a tree-based ensemble learning method with each tree depending upon several randomly selected variables. Besides, random forest is considered to be an extension of bagging [1] and is regarded as a competitor to boosting. Fig. 1 depicts the principle of the random forest:

![Image of the principle of random forest](https://example.com/figure1.png)

Figure 1. The principle of random forest [3].
Nowadays, machine learning methods such as support vector machine, random forest, and neural networks are widely used for their high performance. They do perform well, but users, who regarding them as domain-independent subjects, generally do not understand how they actually work and instead throw data into these algorithms without even consider their applicability.

Hence, this paper will start from the analysis on the advantages and limitations of random forest to discuss its applicability. Through this research, the following questions will come to the conclusion: What kind of data or what kind of predictions is suitable for random forest and what is not. The primary research method of this paper is analyzing and integrating the research results of existing literature. The purpose of this paper is to provide some guidance for the users who have trouble selecting a suitable machine learning algorithm.

2. Advantages of the random forest
With the wisdom of crowds, a large number of relatively unrelated models acting as a committee will outperform any of the single constituent models by buffering and protecting each other from their errors [6].

2.1. Computational advantages
Being surprisingly flexible and having very high accuracy, random forest can deal with both regression and classifications tasks, including multi-class classification, and is relatively fast to train and to make predictions. Additional, random forest is capable of handling the large data set with noise and high dimensionality, and can compute the importance of each variable in the classification process.

2.2. Statistical advantages
Random forest is immune to statistical assumptions and preprocessing burden, and possesses less variance than a single decision tree. Besides, it is more robust compared to boosting. What is more, it overcomes overfitting and can carry out differential class weighting, outlier detection, and missing value imputation.

3. Limitations of the random forest

3.1. Struggle with high-cardinality categorical variables
When users confront a huge number of high cardinality categorical variables, most random forest implementations cannot handle this variable well, because the number of possible splits grows nonlinearly with the cardinality. For example, when people divide n objects into k groups ($g_1, g_2, \ldots, g_k$) by assigning $n_i$ objects to group $g_i (n_1 + n_2 + \ldots + n_k = n)$ [4], the total number of possible partitions is as follows:

$$P_{n_1, n_2, \ldots, n_k} = \frac{n!}{n_1!n_2!\ldots n_k!}$$ (1)

Hence, the computational burden will increase rapidly as n becomes larger. At the same time, there are risks of overfitting. One of the solutions to this problem is to retain the groups that exceed the threshold times and discard the others.

3.2. Uninterpretable
Random forest fails to reflect the intrinsic relationship between features and the target, when the goal is to understand more about the relationship between features and the target, and the degree of influence they have. For instance, in the case to determine whether there is a positive or negative relationship between a feature and a target, and how much it affects the result, then classifiers like LASSO and logistic regression are better choices than random forest. What is more, due to the overfitting, it is difficult for random forest to extrapolate predictions of the data other than the original training data.
3.3. Shortcomings in time series forecasting

Supposing that there is a considerable amount of external (exogenous) variables, random forest can establish an accurate prediction. When people perform series forecasting, the main change is internal effects such as a function of past events and lag time, so most of the variation of targets is from endogenous temporal effects. However, it is hard for random forest to capture these internal effects. For example, random forest is not suitable for increasing or decreasing trends, especially in testing set [3].

A solution is to not utilize features in the data related to the time component to improve generalization. Another idea to further improve predictions is to use time-series forecasting to flatten the time effect, such as seasonality. More directly, the famous ARIMA model can solve the endogenous problems.

3.4. Non-ideal on unbalanced data

Random forest is non-ideal for processing imbalanced data, when there are rare outcomes or rare predictors. A direct solution to this problem is collecting as much balanced down-sampled data as possible. Another two methods, namely, balanced random forest (BRF) and weighted random forest (WRF), were proposed in [3]. To process large amount of imbalanced data, BRF is computationally more efficient. In comparison, WRF is more vulnerable to noise.

3.5. Sensitive to hyperparameters

Previous research conducted by Barbara F.F., Huang and Paul C. Boutros [5] suggest that when users perform the random forest, parameterization is very important, for it is highly correlated with prediction accuracy both on high data and low data. Thus, there is a significant benefit to tuning RFs from their default parameter settings.

![Figure 2. Hyperparameter and model parameter](image)

Hyperparameter tuning

Model training

Figuratively speaking, hyperparameters is like the settings of a machine that can be adjusted to optimize its performance. In the random forest, hyperparameters control tradeoffs between variance, bias, and compute time. For example, stricter splitting rules can reduce the correlation between individual trees, but make each tree more biased. There are three hyperparameters [7] that may be tuned to improve accuracy: m, the number of random variables chosen at each node, j, the number of trees in the forest, and tree size n, as measured by the maximum number of terminal nodes or the smallest node size for splitting. By increasing m, performance improves, while the USP of the random forest, namely, the diversity of individual tree, as well as the speed decreases. In addition, higher number of trees returns better performance, but makes the model more time-consuming. Therefore, to make the prediction more
stable and stronger, it is necessary to choose as high value as the processor can handle. At last, a small tree size can make the forests tend to capturing noise in training data.

In classification, the default is \( m = \sqrt{M} \), where \( M \) stands for the total number of predictors. And in regression, the standard default is \( m = N / 3 \), where \( N \) represents the sample size. According to the former research [7], when tuning is necessary, \( M \) can be chosen in terms of the out-of-bag error rate. That is, it is recommended for users to capture hyperparameters through specific case studies. And as \( J \) increases Breiman showed in [3] that the generalization error of random forest almost certainly converges to a limit. And for \( J \), it can be chosen as large as desired in practice.

3.6. Fast diminishing marginal returns (DMR)
In contrast to another widely used machine learning algorithms called neural network, the training performance of random forest will not continue to improve with the continuous massive data input.

3.7. Strict requirements for the data type
Random forest can only work on tabular data and is not suitable for processing images, audio data, and text data. When it comes to imagine possessing and natural language processing, neural networks will be a better choice.

4. Conclusion
Random forest is a flexible and robust algorithm that unexcelled in accuracy, immune to statistical assumptions and preprocessing burden. Besides, random forest can handle a large data set with high dimensionality and missing values.

However, random forest is not omnipotent and has its limitations. The followings are the solutions in different situations. Firstly, to conduct random forest on high-cardinality categorical variables, diminishing the number of groups (cardinalities) in advance can reduce the computational burden drastically. Secondly, to interpret the internal relationship between features and target, LASSO and logistic regression are better choices. Besides, to process time-series data using the random forest, it is recommended to flatten the time effect. Meanwhile, the ARIMA model is more useful to conduct time series forecasting compared to the random forest.

Additionally, parameter tuning can significantly improve the prediction accuracy of the random forest. And the tuning parameter can be chosen by out-of-bag error rates. What is more, to continually change the model based on input data, neural networks enjoy less diminishing marginal returns (DMR) than the random forest. Last but not least, when users conduct imagine processing and natural language processing, neural networks outperform the random forest typically.

Each method has its area for application. According to George Box, a renowned British statistician, “All models are wrong, but some are useful.”[8]. There are various methods to combine several models into a better one, including boosting, stacking, and random forest. Whereas, it is hard to select a perfect one that works well in all datasets. How to choose models depends on the research field and the goal. Consequently, random forest should not be treated as an application-independent mathematical method. What is more, research should always be conducted in the context of its end-use.

Acknowledgments
First of all, I would like to express my deepest gratitude to the teachers and professors in my university, who provided me with irreplaceable guidance for each stage of my writing. I would also like to thank all my friends and classmates for their encouragement and support. Without all their enlightening guidance and impressive friendliness, I cannot complete my thesis.

References
[1] Breiman, L. (1996). Bagging Predictors. Machine learning, 24(2), 123-140.
[2] Breiman, L. (2001). The Random Forest. Machine learning, 45(1), 5-32.
[3] Chen, L. (2019). Basic Ensemble Learning (the Random forest, AdaBoost, Gradient Boosting)- Step by Step Explained. Accessed on April 23, 2020. Retrieved from: https://towardsdatascience.com/basic-ensemble-learning-random-forest-adaboost-gradient-boosting-step-by-step-explained-95d49d1e2725

[4] Chen, C., Liaw, A., & Breiman, L. (2004). Using Random forest to learn imbalanced data. University of California, Berkeley, 110(1-12), 24.

[5] Ghosh, P. (2017). Combating High Cardinality Features in Supervised Machine Learning. Accessed on April 18, 2020. Retrieved from: https://pkghosh.wordpress.com/2017/10/09/combating-high-cardinality-features-in-supervised-machine-learning/ 2020/4/18

[6] Huang, B. F., & Boutros, P. C. (2016). The parameter sensitivity of the Random forest . BMC bioinformatics, 17(1), 331.

[7] Yiu, T. (2020). Understanding the Random forest. Accessed on April 18, 2020. Retrieved from: https://towardsdatascience.com/understanding-random-forest-58381e0602d2

[8] Zhang, C., & Ma, Y. (2012). Ensemble machine learning: methods and applications: Salmon Tower Building. New York City: Springer.

[9] Box, G. E. (1976). Science and statistics. Journal of the American Statistical Association, 71(356), 791-799.