Classification and Regression with Breiman Random Forests

(Paper: Random Forests - Breiman 2001)

Meghana Killi and Nam Tran
(Dated: Mar 6, 2019)

I. INTRODUCTION

Classification and regression are common problems in machine learning. The former requires prediction of a discrete label for a new observation, based on training data that has already been grouped into categories. In contrast, the latter requires prediction of a continuous numerical quantity for new observations, based on a function obtained by mapping the input training data to required training output.

Currently there exist several methods to solve these problems, the most common being some application of decision trees (described in II below). *Random Forests* (detailed in IV) is one such method that can solve both classification and regression problems.

The current paper, (Breiman 2001), discusses an improvement on regular random forests through the injection of random feature selection (random subspace method). Thus, in Breiman Random Forests (BRF), boostrap aggregating (bagging) is applied to decision trees in conjunction with the random subspace method. The key idea behind this method is that randomness can reproduce, or even exceed, deterministic approaches owing to the Strong Law of Large Numbers, which states that the sample average converges almost surely to the expected value (Loeve 1977). Breiman conjectures that the deterministic method known as *AdaBoost* emulates a random forest at some stage.

In this write-up we will briefly describe the idea of decision tree learning used to define the random forest procedure, explain bagging and random subspace methods, and finally summarize some of the results in (Breiman 2001).

II. DECISION TREES

A decision tree can be considered as linear combination of indicator functions and is usually viewed as a tree-like model as illustrated in fig. 2. To be more specific suppose we have made some observations $(\mathbf{x_1}, y_1), \ldots, (\mathbf{x_n}, y_n) \in \mathcal{X} \times \mathbb{R}$. The idea of a *decision tree* is to partition the set $\mathcal{X}$ in smaller disjoint subsets $\mathcal{R}_1, \ldots, \mathcal{R}_N$ and determine a *classifier*

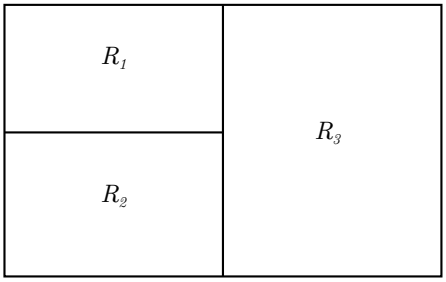

FIG. 1. An example on how the parameter space can be partitioned. The partition of the parameter will depend on the algorithm used to generate a decision tree.

$f : \mathcal{X} \to \mathbb{R}$ of the form,

$$f(\mathbf{x}) = \sum_{i=1}^{N} c_i \mathbb{1}_{\mathcal{R}_i}(\mathbf{x}) \tag{1}$$

that can predict the values $y_1, \ldots, y_n$. Here $\mathbb{1}_{\mathcal{R}_i} : \mathcal{X} \mapsto \{0, 1\}$ is the indicator function defined as

$$\mathbb{1}(\mathbf{x}) = \begin{cases} 1, & \text{if } \mathbf{x} \in \mathcal{R}_i \\ 0, & \text{if } \mathbf{x} \notin \mathcal{R}_i \end{cases} \tag{2}$$

Similar to to the least square method the goal is to determine appropriate constants $c_1, \ldots, c_N$ and in this case also proper regions $\mathcal{R}_1, \ldots, \mathcal{R}_N$ such that the *loss function* for the problem is minimized. It should be noted that although (Breiman 2001) uses a simple sum of squares loss function for regression with random forests, the loss function is not unique and its form should depend on the specific problem. A proper loss function is important since the prediction of the classifier will highly depend on the chosen function (Hastie et al. 2009).

III. BOOTSTRAPPING, BAGGING, AND RANDOM SUBSPACE METHODS

Taking the notation as in section II, the number of observations in the training data is n, and each data point has N features if the elements in the vector $\mathbf{x}$ has N elements; e.g $\mathbf{x}_i = (x_1, \ldots, x_N)$.

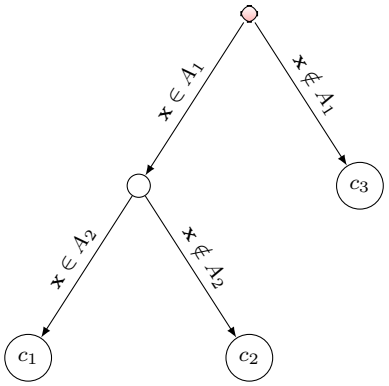

FIG. 2. A decision tree that corresponds to the function $\mathbf{x} \mapsto c_1 \mathbb{1}_{\mathcal{R}_1}(\mathbf{x}) + c_2 \mathbb{1}_{\mathcal{R}_2}(\mathbf{x}) + c_3 \mathbb{1}_{\mathcal{R}_3}(\mathbf{x})$ with $\mathcal{R}_1 = A_1 \cap A_2$, $\mathcal{R}_2 = A_1 \cap A_2^{\complement}$ and $\mathcal{R}_3 = A_1^{\complement}$

A. Bootstrapping

Bootstrapping is a sampling technique in which we randomly sample, with replacement, from the n known observations, to create multiple bootstrap samples of n elements each. Since we allow for replacement, this bootstrap sample is most likely not identical to our initial sample. Some data points may be duplicated, and others may be omitted.

B. Bagging

Bootstrap aggregating (or bagging) is a machine learning ensemble method that uses bootstrap sampling to independently train classifiers, and then aggregates the predictions. When applied to decision trees, bagging provides bootstrap samples to several decision trees and then combines their outputs. Here, the number of trees is a free parameter, and at each node, the algorithm searches over all the N features to find the one that best splits the data.

C. Random Subspace Method

In the Random Subspace Method, instead of selecting from the total N number of features, we instead randomly generate a subspace with $m < N$ features at each node. Then, we either directly use a few features randomly selected from the subspace (Forest-RI), or use random linear combinations of the features from the subspace (Forest-RC), such that the final selected features best split the data.

IV. BREIMAN RANDOM FORESTS

A random forest is a collection of decision trees working in tandem to create a single (aggregate) tree, where each decision tree has some contribution to the final output. For classification problems, each tree *votes* on the discrete label output, and the decision with the most votes will be the decision that the single tree will take. In case of regression problems, each tree will output some numerical value and the single tree will combine them all through a mean or a weighted mean.

Whereas a normal random forest only utilizes bagging, BRF further extends this technique through the addition of the random subspace method. Therefore, in a BRF, after each decision tree receives a bootstrapped sample as input, and performs random subspace sampling of features at each node, the single (aggregate) tree combines their outputs into a final decision.

V. SELECTED RESULTS FROM BREIMAN

BRF, unlike single decision trees and regular bagging methods, avoids overfitting because for a sufficiently large number of iterations, the generalized error converges to some asymptotic value. It is also mostly insensitive to the final number of features selected to split the tree at each node (usually, 1 or 2 features is enough). Moreover, BRF, especially Forest-RC is robust to outliers and noise - more so than Adaboost because Adaboost gives more weight to misclassified data, thus giving more weight to noise, whereas random forests give no weight to any one result.

VI. SUMMARY

Breiman Random Forests utilize a *forest* of decision trees that each makes decisions by randomly sampling from the input data, and from the features at each node. For classification, the algorithm outputs the result that the majority of the trees agree with (i.e., the mode of the results of individual trees), and for regression, it outputs the average value (i.e., the mean (or weighted mean) of the results of individual trees). It makes use of the Strong Law of Large Numbers to avoid overfitting, which is a common problem with other machine learning algorithms such as boosting and bagging.

L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, Oct 2001. ISSN 1573-0565. doi: 10.1023/A:1010933404324. URL https://doi.org/10.

1023/A:1010933404324.

T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer series in statistics. Springer, 2009. ISBN 9780387848846. URL https://books.google.dk/books?id=eBSgoAEACAAJ.

M. Loeve. *Probability Theory*. Springer Science and Business Media. Springer, 1977. ISBN 9781468494648.