Random Forest Algorithm Report

Donal Loitam

July 20, 2022

Contents

1 Introduction

- Random forest is a Supervised Machine Learning Algorithm that is used widely in Classification and Regression problems.
- It builds decision trees on different samples and takes their majority vote for classification and average in case of regression.
- **Has higher accuracy than Decision trees though built out of Decision Trees :** as it is based on the concept of **ensemble learning**, which is a process of combining multiple classifiers to solve a complex problem and to improve the performance of the model.
- As it relies not only on one Decision Tree but greater number of trees in the forest it yields higher accuracy and solves the problem of overfitting.

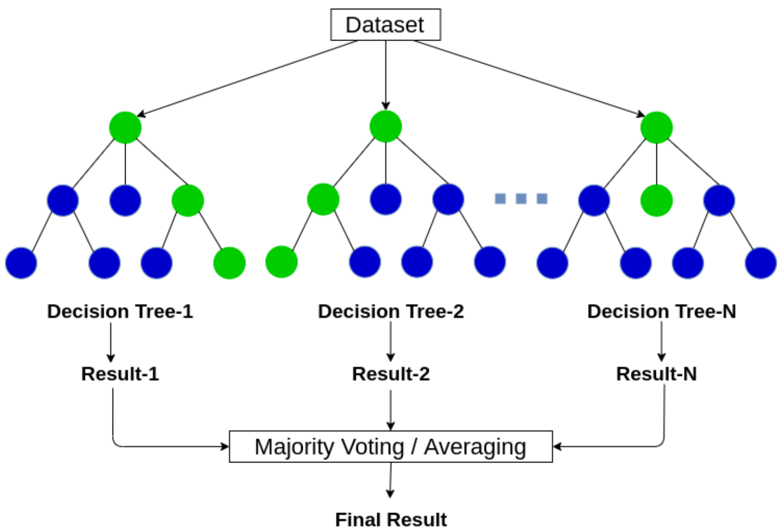

Figure 1: Ensemble learning : Working of a Random Forest

2 Key Points of the Algorithm

The fundamental concept behind random forest is a simple but powerful one — the wisdom of crowds.
"A large number of relatively uncorrelated models(trees) operating as a committee will outperform any of the individual constituent models."
The low correlation between models is the key.
The reason why Random forest produces exceptional results is that the trees protect each other from their individual errors.
Steps of the algorithm
Step 1: Bagging or Bootstrap Aggregation
The random forest allows each individual tree to randomly sample from the dataset with **replacement** (allowing repitition), resulting in different trees. This process is called Bagging.
NOTE : Here, we are not subsetting the training data into smaller chunks and training each tree on a different chunk. Rather, if we have a sample of size N, we are still feeding each tree a training set of size N. But instead of the original training data, we take a random sample of size N with replacement.

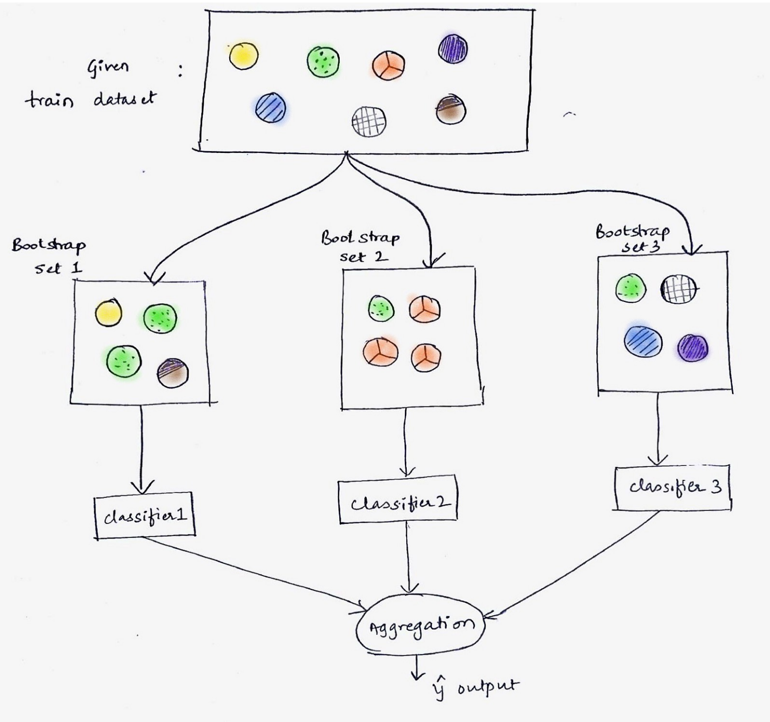

Figure 2: Bagging(with replacement)

Step 2: Random feature selection
In a normal decision tree, when it is time to split a node, we consider every possible feature and pick the one that seperates the best. While in contrast, each tree in a random forest can pick only from a random subset of features. This ultimately results in low correlation across trees. And we just build the tree as usual, but only considering a random subset of variables at each step
We built a tree using .....

1. Using a bootstrapped datset (N)
2. Only consid a random subset of variables/features ('k') at each step.

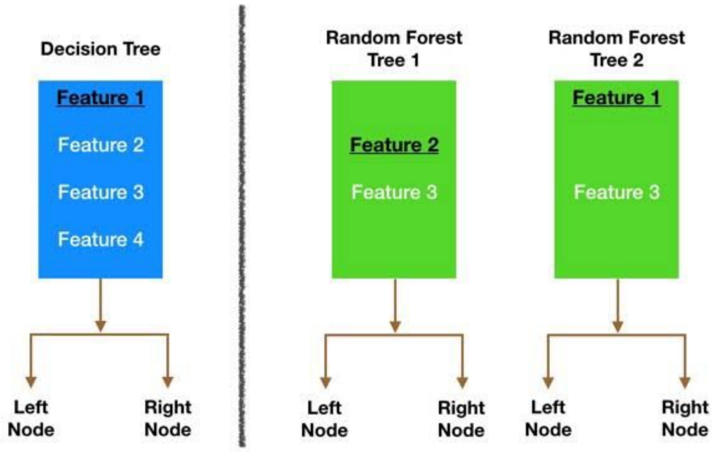

Figure 3: Bagging(with replacement)

3 Some Questions

1. Write a Pseudo Code for the Random Forest algorithm
Ans: Random Forest creation pseudocode:

- Randomly select "k" features from total "m" features where k ¡¡ m
- Among the "k" features, calculate the node "d" using the best split point
- Split the node into daughter nodes using the best split
- Repeat the 1 to 3 steps until some "max_num" number of nodes has been reached
- Build forest by repeating steps 1 to 4 for "n" number times to create "n" number of trees.

2. What is Out-of-Bag Error?
Ans. Out-of-Bag(OOB) is equivalent to validation or test data. Remember when we built the bootstrap dataset, we allowed duplicates and hence some of the datas were left out. Typically about 1/3rd of the samples does not end up in the bootstrapped dataset.
Each tree is tested on 1/3rd of the samples (36.8%) that are not used in building that tree (similar to the validation data set).This is known as the out-of-bag error estimate
3. Why does the Random Forest algorithm not require split sampling methods?
Ans This is because it performs training on 2/3rd of the available training data that is used to grow each tree and the remaining one-third portion of training data is always used to calculate out-of-bag error to compute the model performance.
4. Prove that in the Bagging method only about 63% of the total original examples (total training set) appear in any of sampled bootstrap datasets. Provide proper justification.
Ans. Input : n labelled training examples $S = (xi, yi), i = 1, .., n$
Suppose we select n samples out of n with replacement to get a training set S_i still different from working with the entire training set.

$$Pr(S_i = S) = n!/n^n \quad \text{(very small number, exponentially small in n)} \tag{1}$$

$$Pr((x_i, y_i) \notin S_i) = (1 - 1/n)^n = e^{-1} = 0.37 \tag{2}$$

Hence for large data sets, about 37% of the data set is left out!
5. How does random forest define the Proximity (Similarity) between observations?
Ans Random Forest defines proximity between two data points in the following way:

- Initialize proximities to zeroes.
- For any given tree, apply all the cases to the tree.
- If case i and case j both end up in the same node, then proximity $prox(i, j) += 1$
- Accumulate over all trees in Random Forest and normalize by twice the number of trees in Random forest.
- Finally, it creates a proximity matrix i.e, a square matrix with entry as 1 on the diagonal and values between 0 and 1 in the off-diagonal positions.
- Proximities are close to 1 when the observations are "alike" and conversely the closer proximity to 0, implies the more dissimilar cases are.