Assignment-5: A mathematical essay on Random Forest

Vasudev Gupta
Interdisciplinary Dual Degree - Data Science
Indian Institute of Technology Madras
Chennai, India
me18b182@smail.iitm.ac.in

***Abstract*—In this work, we will present a mathematical overview of the Random Forest. We will use the Random Forest to classify a car based on its safety. In order to provide the correct data to the model, we will also try exploratory data analysis and feature selection. Finally, we will examine if our model can properly generalise to previously unknown data.**

I. Introduction

Machine learning techniques are gaining popularity, helping many organisations solve challenging issues using data-driven approaches. Machine learning approaches try to estimate the function based on supplied training data by capturing the underlying distribution of the training data and then predicting unseen data using the learnt distribution. When creating the model, one wants the training data to be similar to previously unseen data and generalise well to future data.

Random Forests can be considered a specialised form of decision tree bagging. Bagging is done in such a way that at each node, just a random subset of all the attributes is considered. We will go over a full mathematical analysis in the next part.

The Car Evaluation Database was derived from a simple hierarchical decision model. The prediction task is to classify a car based on its safety. We are given many qualities to base our predictions on, but we must evaluate if any of them may significantly influence the model's performance.

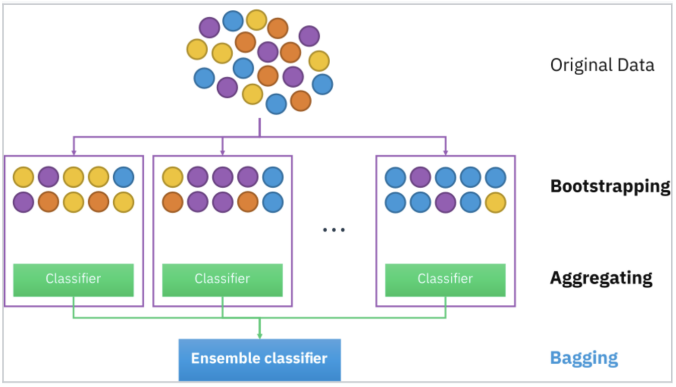
This article will infer a mathematical overview of the Random Forest and use existing Python packages to fit the Random Forest Classifier to the given data. We will also discuss the data insights we discovered and why we chose to alter or delete particular features.

II. Random Forest

Before we get into random forests, we'll go over Bagging and Decision Trees briefly.

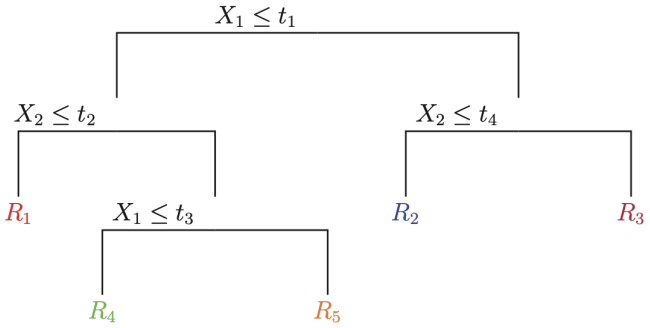
A. Bagging

Bagging is a technique for combining many distinct models developed on a subset of data simply by averaging them (or taking a majority vote). When trained on different subsets of the training data, high capacity learners produce extremely unique models.



B. Decision Trees

To explain the Decision trees, consider the tree in the figure below. The first node asks if x_1 is less than some threshold t_1. If yes, we then ask if x_2 is less than some other threshold t_2. If yes, we are in the bottom left quadrant of space, R_1. If no, we ask if x_1 is less than t_3. And so on.



We can write the model in the following form:

$$f(x) = E[y|x] = \sum_{m=1}^{M} w_m I(x \in R_m) = \sum_{m=1}^{M} w_m \phi(x; v_m)$$

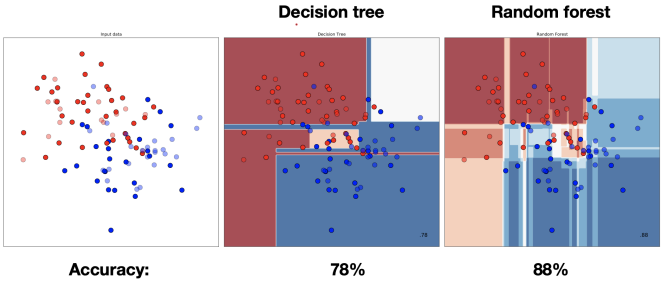
where R_m is the m'th region, w_m is the mean response in this region, and v_m encodes the choice of variable to split on, and the threshold value, on the path from the root to the m'th leaf. This makes it clear that a decision tree is just a an adaptive basis-function model, where the basis functions

define the regions, and the weights specify the response value in each region. We discuss how to find these basis functions below.

We can generalize this to the classification setting by storing the distribution over class labels in each leaf, instead of the mean response.

C. Random Forest Classifier

1) Introduction: Random Forests may be created by doing specialised bagging on decision trees. The decision trees are trained in such a way that only a random subset of all the characteristics are considered at each node. As a result, the variance of each decision tree increases during bagging. This is useful when there is a highly predictive feature, because all learners will use it and so be correlated.



On the same data, the figure above contrasts decision trees vs random forest. We can easily observe that random forest outperformed decision trees in terms of generalisation.

2) Algorithm:

- Repeat $n_{estimators}$ times:
  - Select `a*m` samples at random (with replacement preferably)
  - Learn a decision tree with the above sample, with a small variant (While constructing each node randomly select `b*d` attributes and choose the best split only among these).
- Make prediction by averaging the $n_{estimators}$ learned models.

III. The Problem

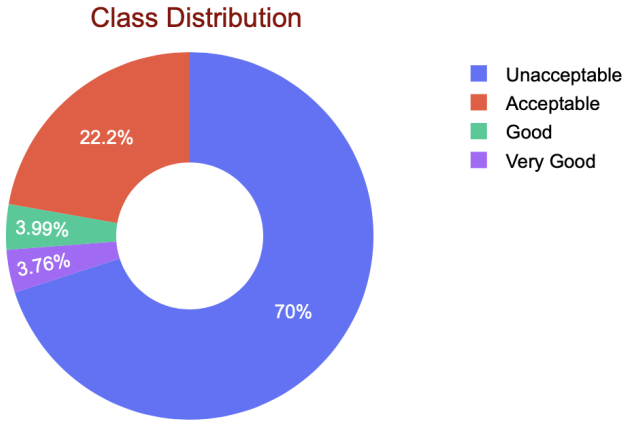
The prediction task is to classify a car based on its safety. We will first analyse the data before attempting to fit the Random Forest to it. The trained model will next be tested on an unknown dataset.

A. Data Analysis and Feature Engineering

Data was clean and did not have any `NaN` values, so we did not worry about them.

```
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   Price        1728 non-null   object
 1   Maintenance  1728 non-null   object
 2   NumDoors     1728 non-null   object
 3   NumPersons   1728 non-null   object
 4   LugBoot      1728 non-null   object
 5   Safety       1728 non-null   object
 6   Class        1728 non-null   object
```

We tried to visualize the class distribution in the data and see if there was any imbalance in the target variable.

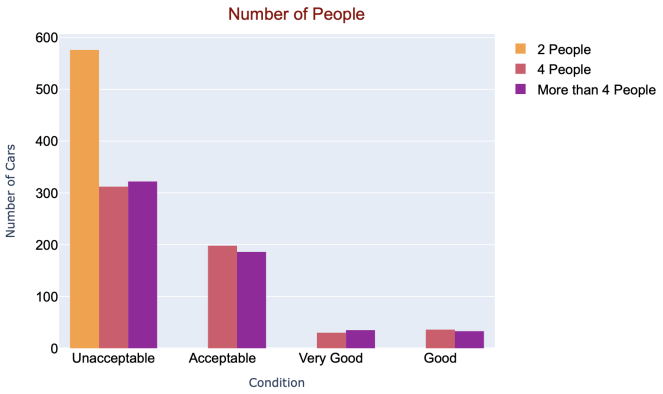


The above pie chart displays that data is quite imbalanced and most of the samples belongs to `Unacceptable` and `Acceptable` categories.

Further, before fitting the model to the data, we tried to understand the categorical features in the data using the bar plots.

The above table depicts that safety in cars is directly proportional to the number of acceptable cars. Hence, the more unsafe the car is, the more likely it will not be acceptable.

The above figure depicts that more people prefer a car with either medium/high luggage capacity.



The above figure clearly says that more people tend to buy cars which can accommodate at least four people.

From the above plot, we can conclude that highly-priced cars are unacceptable and people like to get a car for less money.

B. Model Fitting

We fitted the Random Forest using the `sklearn` library. Since tree-based models are pretty prone to overfitting, we first tuned the hyper-parameters using `RandomizedSearchCV`. We also used stratified K-fold to find the best hyper-parameters of this dataset. Accuracy score and Mathews Correlation Coefficient are presented in the following table:

Mathews Correlation Coeff	94.5%
Accuracy	97.9%

In following table, we report the best hyper-parameters for this model and data.

min samples split	2
min samples leaf	1
max features	”log2”
criterion	”gini”

IV. Conclusions

In this essay, we presented the mathematical intuition behind Random Forest. We also applied Random Forest to real-world data and inspected whether the fitted model can predict well. We conclude that Random Forest can be used on complicated real-world datasets after proper tuning.

References

[1] Christopher M. Bishop, ”Pattern Recognition and Machine Learning”
[2] Kevin P. Murphy, ”Machine Learning, A Probabilistic Perspective”