Assignment-5: A Mathematical essay on Random Forest.

Ahmed Shmels Muhe
M.Tech In Data Science
Indian Institute of Technology Madras, IIT Madras
Chennai, India
ge22m009@smail.iitm.ac.in

***Abstract*—In this revised (v. 2) mathematical essay on Random Forest, In this work, we demonstrate the application of the Random Forest algorithm and the mathematics behind it. Random Forest classifier illustrated through the Car Evaluation Database Due to the complex nature and the abstractness involved in classifying a car given its features, and it is often difficult to obtain a good solution through manual approaches. Adopting a machine learning technique can help get insights and comprehensively analyse different components in a large amount of data. We try to establish a nexus between the nature of the car and the additional features involving its specifications. In this revised version, I include the mathematics behind Random Forest and the concept of bagging. In Section 3, "Data Analysis, Feature Engineering, and Model Fitting," some of the outputs were modified after some code modifications.**

***Index Terms*—Random Forest, classifier.**

I. INTRODUCTION

In this paper, we will study Random Forests, a technique predominantly based on the collection of tree-like decision models that only contain control statements. It is used to analyze and model either dichotomous or multiple outcomes in a classification setting and could also be extended as a regressor in a regression setting. We will use this technique to classify the nature of a car using factors such as safety, capacity, costs etc. This analysis would aid in obtaining insights among a large dataset and understanding the trend behind the nature of a car and various other factors.

Random Forests can be considered a specialized form of decision tree bagging. Bagging is done so that at each node, just a random subset of all the attributes is considered. We will go over a complete 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 on to base our predictions, but we must evaluate if any of them may significantly influence the model's performance. The dataset for this problem comprises safety, luggage space, capacity, doors, maintenance and buying costs. We use a Random Forest classifier to learn and predict the nature of the car (unaccountable, accountable, good, very good) given these input features.

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 combines many distinct models developed on a subset of data simply by averaging them (or taking a majority vote). High-capacity learners produce unique models when trained on different subsets of the training data. Bagging is an ensemble algorithm that fits multiple models on different subsets of a training data set and then combines the predictions from all models.

Random forest is an extension of bagging that also randomly selects subsets of features used in each data sample. Both bagging and random forests have proven effective on a wide range of different predictive modelling problems.

Although practical, they are unsuitable for classification problems with a skewed class distribution. Nevertheless, many modifications to the algorithms have been proposed that adapt their behaviour and make them better suited to a severe class imbalance.

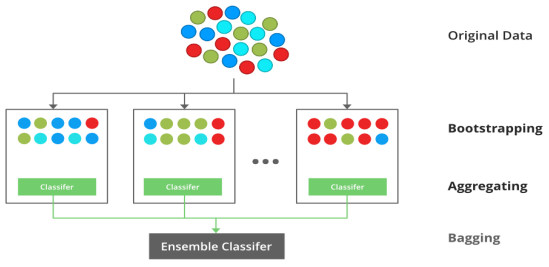

Fig. 1. example for bagging

B. Decision Trees

To explain the Decision trees, consider the tree in the figure below. The first node asks if $x1$ is less than some threshold $t1$. If yes, we then ask if $x2$ is less than some other threshold

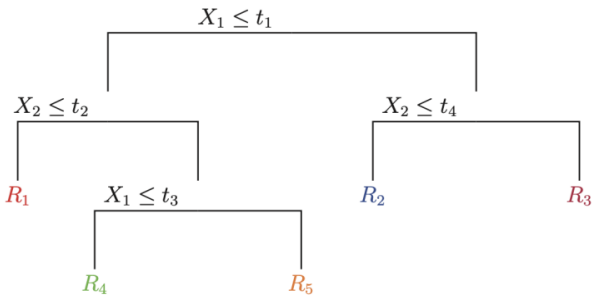

Fig. 2. Decision Trees

$t2$. If yes, we are in the bottom left quadrant of space, $R1$. If no, we ask if $x1$ is less than $t3$. And so on.

We can write the model in the following form:

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

where Rm is the m'th region, Rm is the mean response in this region, and vm 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 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 is 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.

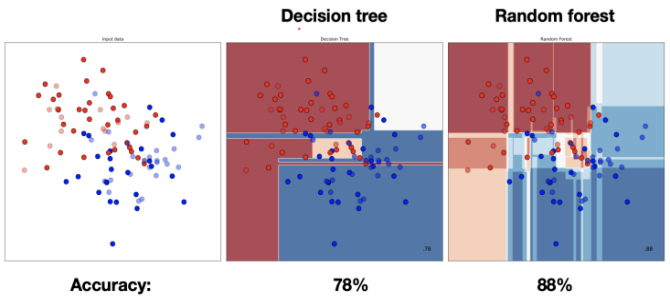

Fig. 3. On the same data, the figure above contrasts decision trees vs random forests. We can easily observe that random forests outperformed decision trees in terms of generalisation.

D. Algorithm for Random Forest Classifier

- 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 data set.

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

Fig. 4. We tried to visualize the class distribution in the data and see if there was any imbalance in the target variable.

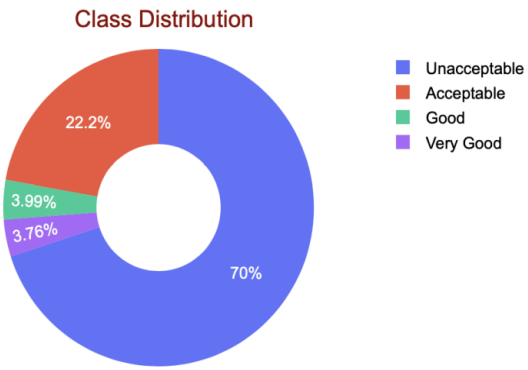

Fig. 5. The above pie chart displays that data is quite imbalanced and most of the samples belong to Unacceptable and $Acceptablecategories$.

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

Fig. 6. 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

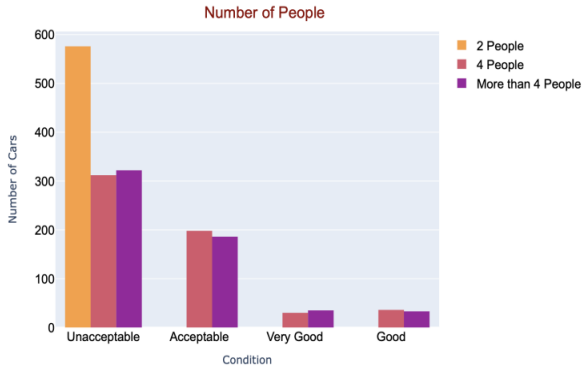

Fig. 7. The above figure clearly says that more people tend to buy cars which can accommodate at least four people.

Fig. 8. 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 RandomizedSearch CV. We also used stratified K-fold to find the best hyper-parameters of this dataset. Accuracy score and Mathews Correlation Coefficient is presented in the following table:

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

Mathews Correlation Coeff	94.5%
Accuracy	97.9%

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

IV. IMPROVEMENTS

We selected a subset of features using the Recursive feature elimination technique. Recursive feature elimination, in short RFE, is a greedy optimization algorithm that aims to find the best-performing feature subset. Each iteration trains a classification model (in this case, Random Forest Classifier).

It eliminates the worst-performing feature until all the features are exhausted or satisfy the stopping criteria.

We observed that RFE helped in increasing the model's performance by a significant amount. With RFE, we were able to reduce the number of features by 16%. Our new results are summarized in the following table:

Mathews Correlation Coeff	95.7%
Accuracy	98.2%

We obtained the above results using the Random Forest as a base estimator in RFE. In the base estimator, the following hyperparameters are used:

min samples split	2
max depth	12
max features	”log2”
criterion	”gini”

V. 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] Belgiu, M., Drăguţ, L. (2016). Random forest in remote sensing: A review of applications and future directions. ISPRS journal of photogrammetry and remote sensing, 114, 24-31.
[2] Rigatti, S. J. (2017). Random forest. Journal of Insurance Medicine, 47(1), 31-39.
[3] Breiman, L. (2001). Random forests. Machine learning, 45(1), 5-32.