Assignment 5: Random Forest Classifier

Prajwal Dinesh Sahu
ME18B114
Dept. of Mechanical Engineering
Indian Institute of Technology, Madras
me18b114@smail.iitm.ac.in

***Abstract*—This paper gives a brief introduction of Random Forest Classifier and the basic mathematical theory behind it. We also look at some metrics to evaluate the performance of our model. Further, we apply the classifier on the Car Evaluation Database, derived from a simple hierarchical decision model. The prediction task is to classify based on its safety. Through exploratory data analysis and decision trees classifier, we investigate trends in data to predict which cars are safer. Then we evaluate our model using confusion matrix and F1 score. Our findings reveal some interesting trends.**

I. Introduction

There are multiple types of algorithm methods used in machine learning. One such popular and commonly used machine learning method is Random Forest Classifier.

Random forests are an ensemble learning method for classification, regression and other tasks that operates by constructing a multitude of decision trees at training time. For classification tasks, the output of the random forest is the class selected by most trees. For regression tasks, the mean or average prediction of the individual trees is returned. Random decision forests correct for decision trees' habit of over fitting to their training set. Random forests generally outperform decision trees.

In this paper, we apply Random Forest Classifier to the Car Evaluation Database, which was derived from a simple hierarchical decision model, to classify a car based on its safety.

The paper will first give a brief explanation of decision trees classifier. Then, we will try to solve the Income prediction and model our data-set by
1. Cleaning the data.
2. Exploratory and visual analysis.
3. Feature Engineering for Categorical Variables.
4. Checking for Correlation.
5. Training the model.
6. Evaluating the model.
7. Interpreting the results.

II. Random Forest Classifier Theory

A. Classification

Classification is a process of categorizing a given set of data into classes. Predicting a qualitative response for an observation can be referred to as classifying that observation, since it involves assigning the observation to a category, label or class.

Many real world problems require classification rather than regression. In the classification setting we have a set of training observations (x1, y1), . . . , (xn, yn) that we can use to build a classifier.

Regression is not useful in problems like if we are trying to predict the medical condition of a patient in the hospital and we have 3 possible diagnoses- stroke, drug overdose, and epileptic seizure based on the symptoms of the patient. We can label these diseases at 1,2 and 3. In this case, even if we create a metric to accurately convert the symptoms to numeric data, regression won't be able to classify the data as it cannot give the output as 1,2 or 3. Alternatively, we can create a threshold that if predicted value is between 0.5 to 1.5, we will assign it to label 1. This brings us to the basic idea behind classification algorithms which can give qualitative response.

B. Decision Trees

A tree based classifier involves stratifying or segmenting the predictor space into a number of simple regions. In order to make a prediction for a given observation, we typically use the mean or the mode of the training observations in the region to which it belongs. Since the set of splitting rules used to segment the predictor space can be summarized in a tree, these types of approaches are known as decision tree methods. Tree-based methods are simple and useful for interpretation.

It is a tree-structured classifier, where internal nodes represent the features of a data set, branches represent the decision rules and each leaf node represents the outcome.

Terminology

- **Root Node** -Root node is from where the decision tree starts. It represents the entire data set, which further gets divided into two or more homogeneous sets.
- **Leaf Node** - Leaf nodes are the final output node, and the tree cannot be segregated further after getting a leaf node.
- **Splitting** - Splitting is the process of dividing the decision node/root node into sub-nodes according to the given conditions.
- **Branch Tree** - A tree formed by splitting the tree.

- **Pruning** -Pruning is the process of removing the unwanted branches from the tree.

C. *Working of Decision Tree Algorithm*

In a decision tree, for predicting the class of the given data set, the algorithm starts from the root node of the tree. This algorithm compares the values of root attribute with the record (real data set) attribute and, based on the comparison, follows the branch and jumps to the next node.

For the next node, the algorithm again compares the attribute value with the other sub-nodes and move further. It continues the process until it reaches the leaf node of the tree.

Attribute Selection - While implementing a Decision tree, the main issue arises that how to select the best attribute for the root node and for sub-nodes. So, to solve such problems there is a technique which is called as Attribute selection measure or ASM. There are two techniques for this -

*1) **Gini Index**:* Gini Impurity is a measure of how often a randomly chosen element from the set would be incorrectly labeled if it was randomly labeled according to the distribution of labels in the subset. The Gini impurity can be computed by summing the probability of an item p_i with label i being chosen times the probability -

$$\sum_{k\neq i} p_k = 1 - p_i \tag{1}$$

, of a mistake in categorizing that item. It reaches its minimum (zero) when all cases in the node fall into a single target category.

To compute Gini impurity for a set of items with J classes, suppose $i \in \{1,2,...,J\}$,p_i be the fraction of items labeled with class *i* in the set.

$\mathrm{I}_G(p) = \sum_{i=1}^{J}\left(p_i \sum_{k\neq i} p_k\right)$

= $\sum_{i=1}^{J} p_i(1-p_i)$

= $\sum_{i=1}^{J}(p_i - p_i^2)$

= $\sum_{i=1}^{J} p_i - \sum_{i=1}^{J} p_i^2$

= 1 - $\sum_{i=1}^{J} p_i^2$

*2) **Information Gain**:* Information gain is the measurement of changes in entropy after the segmentation of a dataset based on an attribute. It calculates how much information a feature provides us about a class. According to the value of information gain, we split the node and build the decision tree. A decision tree algorithm always tries to maximize the value of information gain, and a node/attribute having the highest information gain is split first. It can be calculated using the below formula:

Information Gain = Entropy(S) - [(Weighted Avg)* Entropy(each feature)]

Entropy is defined as -

$$(T) = \mathrm{I}_E\left(p_1, p_2, \ldots, p_J\right) = -\sum_{i=1}^{J} p_i \log_2 p_i \tag{2}$$

where $p_1, p_2, \ldots$ are fractions that add up to 1 and represent the percentage of each class present in the child node that results from a split in the tree.

$$\overbrace{E_A(\mathrm{IG}(T,a))}^{\text{expected information gain}} = \overbrace{(T)}^{\text{entropy (parent)}} - \overbrace{(T \mid A)}^{\text{weighted sum of entropies (children)}}$$
$$= -\sum_{i=1}^{J} p_i \log_2 p_i - \sum_{a} p(a) \sum_{i=1}^{J} - \Pr(i \mid a) \log_2 \Pr(i \mid a)$$

That is, the expected information gain is the mutual information, meaning that on average, the reduction in the entropy of T is the mutual information.

D. *Bagging*

Bagging, also known as bootstrap aggregation, is the ensemble learning method that is commonly used to reduce variance within a noisy dataset. In bagging, a random sample of data in a training set is selected with replacement—meaning that the individual data points can be chosen more than once.

The training algorithm for random forests applies the general technique of bagging to tree learners. Given a training set $X = X_1, X_2, ...X_n$ with responses $Y = y_1, y_2, ..., y_n$, bagging repeatedly selects a random sample with replacement of the training set and fit trees to these samples.

For b = 1,...,B:
1. Sample, with replacement, n training examples from X, Y; call these X_b, Y_b.
2. Train a classification tree f_b on X_b, Y_b.

After training, predictions for unseen samples x can be made by averaging the predictions from all the individual regression trees on x:

$$\hat{f} = \frac{1}{B}\sum_{b=1}^{B} f_b(x) \tag{3}$$

or by taking the majority vote in the case of classification trees.

The number of samples/trees, B, is a free parameter. Typically, a few hundred to several thousand trees are used, depending on the size and nature of the training set. An optimal number of trees B can be found using cross-validation, or by observing the out-of-bag error: the mean prediction error on each training sample xi, using only the trees that did not have xi in their bootstrap sample. The training and test error tend to level off after some number of trees have been fit.

E. *Random Forest Classifier*

Random forest, like its name implies, consists of a large number of individual decision trees that operate as an ensemble. Each individual tree in the random forest spits out a class prediction and the class with the most votes becomes our model's prediction.

Random Forest uses a modified tree learning algorithm that selects, at each candidate split in the learning process, a random subset of the features. This process is sometimes called feature bagging. The reason for doing this is the correlation of the trees in an ordinary bootstrap sample: if one or a few features are very strong predictors for the response variable (target output), these features will be selected in many of the B trees, causing them to become correlated.

F. *Advantages of Random Forest*

There are many advantages of random forest over decision trees -

- It reduces overfitting in decision trees and helps to improve the accuracy.
- It is flexible to both classification and regression problems.
- It works well with both categorical and continuous values.
- Normalising of data is not required as it uses a rule-based approach.

G. *Evaluating the model*

In model evaluation, we measure the extent to which the model fits the data.

*1) **Confusion Matrix**:* A confusion matrix is a table that shows the performance of a machine learning model. It displays how many of the model's predictions were right and how many were incorrect and which classes did the model succeed in and which did it fail in. It offers us an understanding of how the model works. If a model fails for a certain class, we may investigate it, figure out why, and try to improve the model.

There are various metrics based on confusion matrix. Some of them are listed below

*2) **Precision and Recall**:* Precision is the ratio between the True Positives and all the Positives. For our model, it will be the measure of number of cars for which we correctly identified its safety category.

$$\text{Precision} = \frac{\text{True Positive}}{\text{True Positive + False Positive}} \tag{4}$$

Recall is the measure of our model correctly identifying True Positives. For all the cars in the given safety label, recall tells us how many we correctly predicted.

$$\text{Recall} = \frac{\text{True Positive}}{\text{True Positive + False Negative}} \tag{5}$$

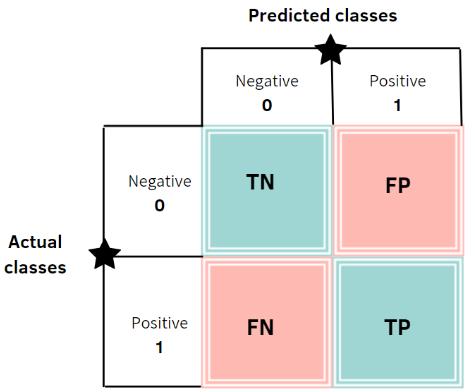


Fig. 1. Sample Confusion Matrix

*3) **F1 Score**:* F1 score is the measure which combines both precision and recall. It is calculated as the harmonic mean of precision and recall
-

$$F_1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{6}$$

III. The Problem

The Car Evaluation Database was derived from a simple hierarchical decision model. The prediction task is to classify a car based on its safety

A. *The Variables*

Buying - The price of the car.
Maintenance - The amount of maintenance required.
Doors - The number of doors.
Persons - The capacity of the car.
Lug boot - The size of the luggage boot in the car.
Safety - The estimated safety of the car.

B. *The Target Variable*

We have to predict the overall safety of the car. In the given data, there is an high imbalance and a lot of cars have unacceptable safety, denoted by 'unacc'. A countplot is shown below -

C. *Numerical Variables*

*1) **Doors**:* This variable indicates the number of doors in the car. To visualise the relationship of safety and number of doors, we plot the below given graph-

From the plot, we can see that there is not a significant difference between cars having 4 or more doors, but the cars having 2-3 doors are less safer.

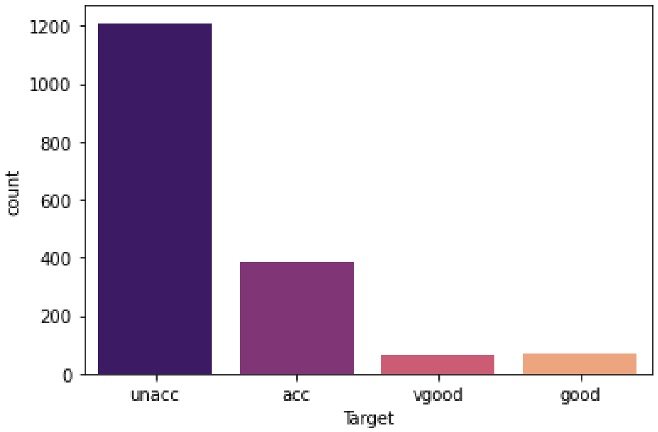


Fig. 2. Target Countplot

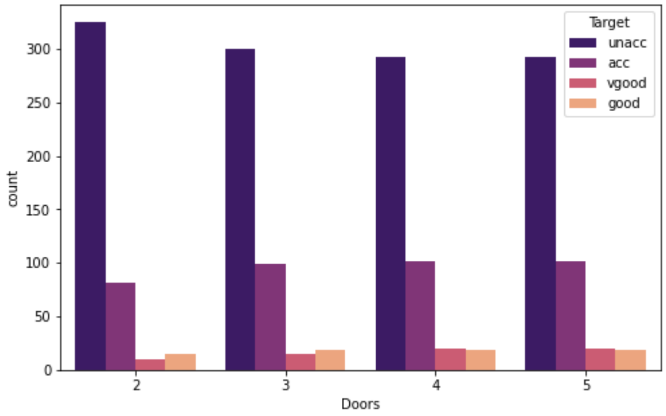


Fig. 3. Doors vs Safety

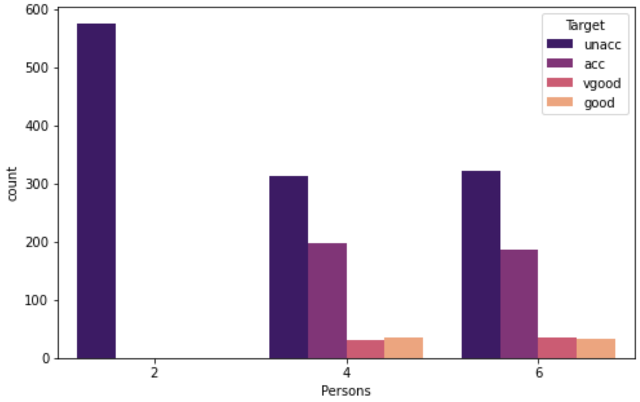


Fig. 4. Number of Persons vs Safety

*2) **Persons**:* This variable indicates capacity of the car. To visualise the relationship of safety and number of persons the car can hold, we plot the below given graph-

From the plot, we can see that the cars having higher capacity are on the safer side.

D. Categorical Variables

*1) **Buying**:* This variable indicates the price of the car. It classifies them as vhigh, high, med, and low. To visualise the relationship of safety and buying price, we plot the below given graph-

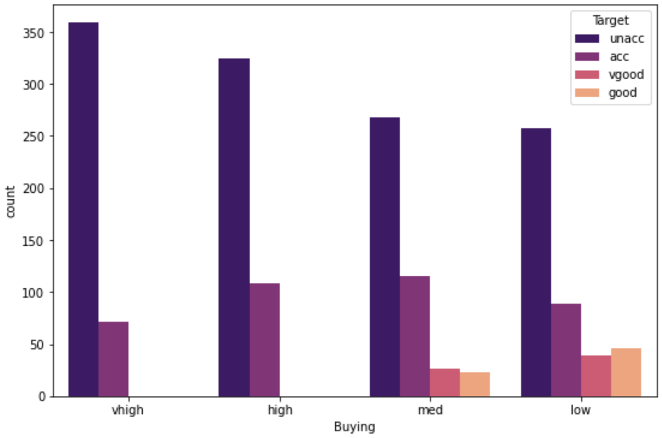


Fig. 5. Buying variable countplot

There seems no significant relationship between safety and buying price.

*2) **Maintenance**:* This variable indicates the amount of maintenance required for the car. It classifies them as vhigh, high, med, and low. To visualise the relationship of safety and required maintenance, we plot the below given graph-

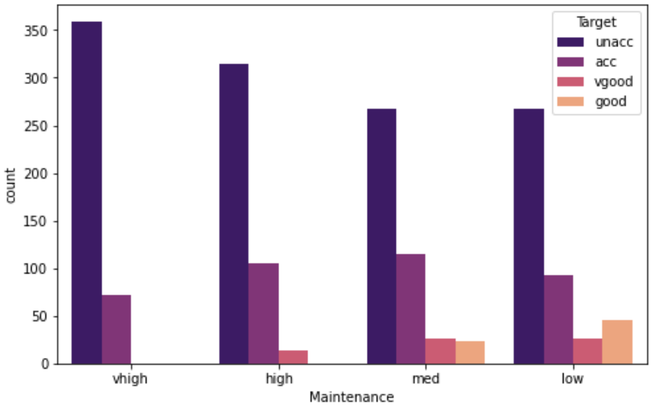


Fig. 6. Maintenance variable countplot

The plot indicates that lower maintenance cars are relatively safer.

*3) **Lug boot**:* This variable indicates the size of the luggage boot of the car. It classifies the size as small, med and big. To visualise the relationship of safety and required maintenance, we plot the below given graph-

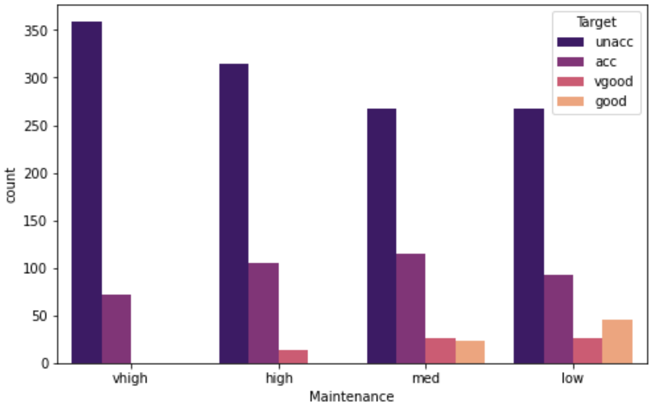


Fig. 7. Lug Boot variable countplot

The plot indicates a weak positive relationship between larger luggage boot and higher safety.

*4) **Safety**:* This variable indicates the estimated safety of the car. It classifies the size as low, med and big. To visualise the relationship of actual safety and estimated safety, we plot the below given graph-

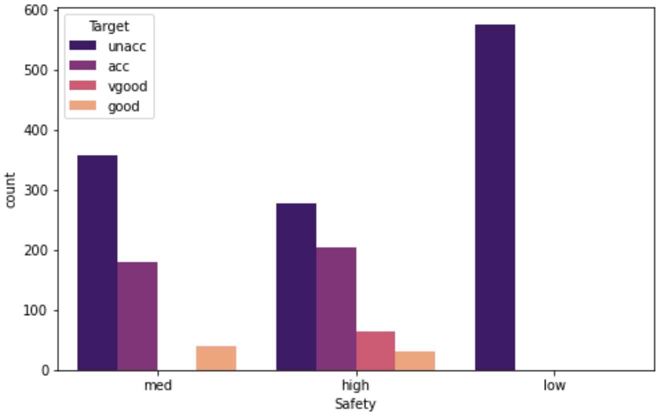


Fig. 8. Safety variable countplot

As expected, the plot indicates that the estimations are largely correct and cars with higher safety rating are relatively safer.

E. *Checking Correlation*

Through the below heatmap, we can see no high values of correlation among our numeric variables-

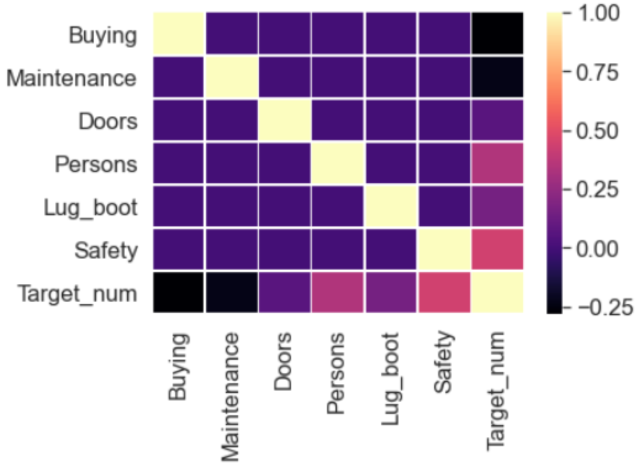


Fig. 9. Correlation Heatmap

We can see there is positive correlation of safety with number of persons and the estimated safety.

IV. The Model

A. *Train Test Split*

We have the following X and y for our final model, which we divide our data into training and testing data..
X = 'Buying', 'Maintenance', 'Doors', 'Persons', 'Lug boot', 'Safety' Y = Target

B. *Training the Model and Parameter Tuning*

Using scikit-learn.ensemble's RandomForestClassifier() function on our X and y variables, we develop a Tree model.

We have to decide the number of layers, or the Max depth parameter. So we plot a graph between max depth and cross validation score. We witness an elbow shaped graph, we choose maxdepth as 11.

C. *Predictions and Metrics*

The confusion matrix for our testing set in shown below

Based on the matrix, we can calculate the following (Fig 12) -

The f1 score is 0.98.

V. Conclusions

We were able to achieve a F1 score of 0.98 with a max depth of 11 by fitting Random Forest Classifier on our data set. From our observations, we can conclude the following

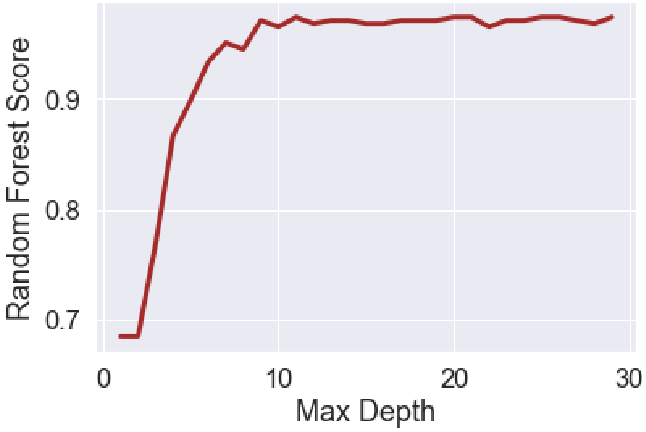


Fig. 10. Max Depth vs Score

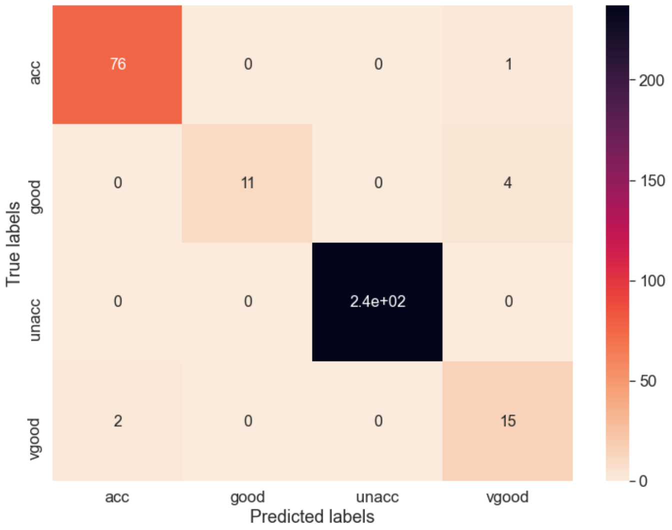
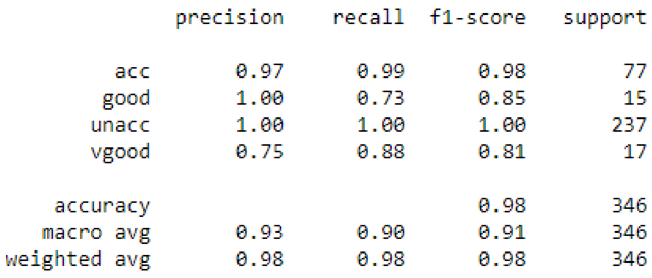


Fig. 11. Confusion Matrix for testing data



```
              precision    recall  f1-score   support

         acc       0.97      0.99      0.98        77
        good       1.00      0.73      0.85        15
       unacc       1.00      1.00      1.00       237
       vgood       0.75      0.88      0.81        17

    accuracy                           0.98       346
   macro avg       0.93      0.90      0.91       346
weighted avg       0.98      0.98      0.98       346
```

Fig. 12. Scores for testing data

about the significant features-

- **Doors and Passengers** - Cars with higher number of doors and higher passenger capacity are safer than the smaller cars .
- **Price** - The price of the car does not affect its safety.
- **Luggage Boot Size** - Cars with bigger luggage boot tend to have better safety.
- **Estimated safety** - Quite intuitively, cars with higher estimated safety are more safer.
- **Maintenance** - Cars which require low maintenance are more safe than their high maintenance counterparts.

Thus, as shown by both the qualitative and the quantitative analysis, the safety of the car depends on various factors.

References

[1] G. James, D. Witten, T. Hastie, and R. Tibshirani, An introduction to statistical learning: With applications in r. New York, NY: Springer, 2021.

[2] T. Hastie, J. Friedman, and R. Tisbshirani, The elements of Statistical Learning: Data Mining, Inference, and prediction. New York: Springer, 2017.

[3] "Random Forest," Wikipedia, 02-Nov-2021. [Online]. Available: https://en.wikipedia.org/wiki/Randomforest. [Accessed: 11-Nov-2021].

[4] T. Yiu, "Understanding random forest," Medium, 29-Sep-2021. [Online]. Available: https://towardsdatascience.com/understanding-random-forest-58381e0602d2. [Accessed: 11-Nov-2021].

[5] "Sklearn.ensemble.randomforestclassifier," Available: https://scikit-learn.org/stable/module/sklearn.ensemble.RandomForestClassifier.html. [Accessed: 11-Nov-2021].