**Insurance Claims- Fraud Detection**

Insurance fraud is any act committed to defraud an insurance process. Fraud occurs when someone knowingly lies to obtain a benefit or advantage to which they are not entitled or someone knowingly denies a benefit that is due and to which someone is entitled.

Fraud claims can be highly expensive. It is important to know which claims are correct and which are fraud. It is not easy to check all claims one by one as it may take too much time and costs expensive. Hence, in this project we use machine learning attributes to predict the fraud claim. Machine Learning techniques gives highly accurate result and is less time consuming.

Problem Definition

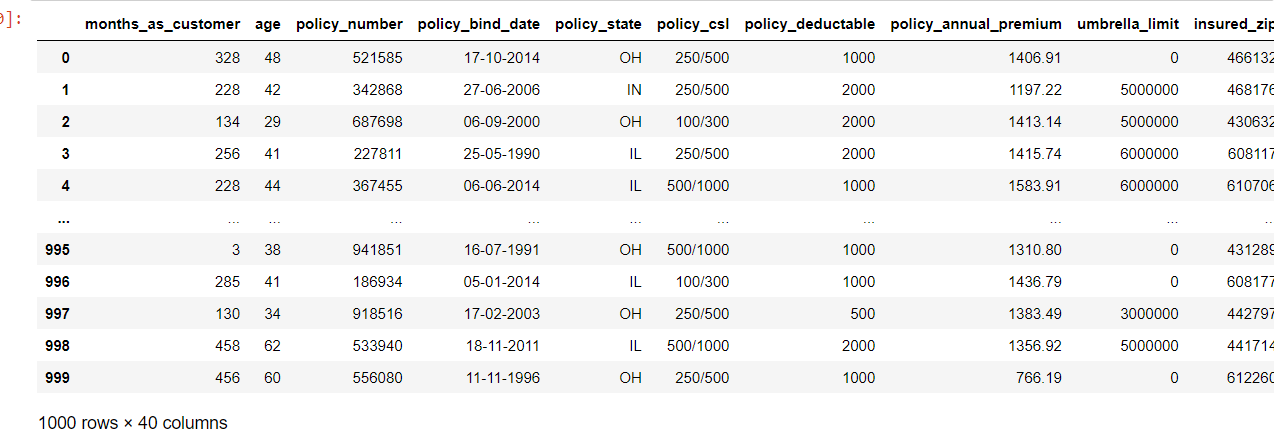
Insurance fraud is a huge problem in the industry. It is difficult to identify fraud claims.

The goal of this project is to build a model that can detect auto insurance fraud. Machine Learning is a unique solution for the auto insurance industry with this problem.

Fraud’s detection using machine learning is less compared to legit insurance claims. While building detection models, the savings from less prevention needs to be balanced. Machine Learning techniques allow for improving predictive accuracy, enabling loss control units to achieve higher coverage with low false positive rates.

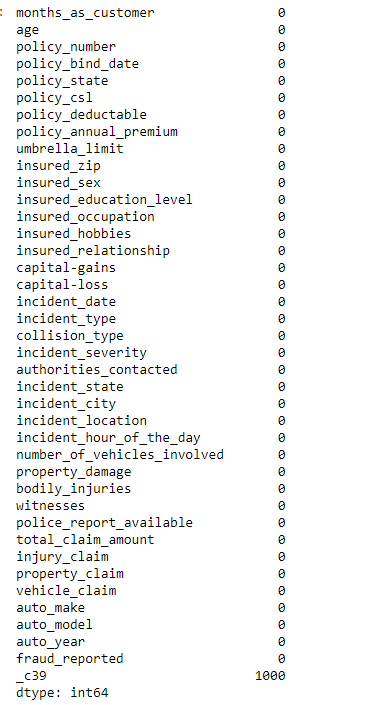
Data Analysis

In this project, we have a dataset which has auto insurance policy details along with the customer details. The given dataset contains 1000 rows and 40 columns. The column names like months\_as\_customer, age, policy\_number, policy\_bind\_date, policy\_state, policy\_csl, policy\_deductable etc.



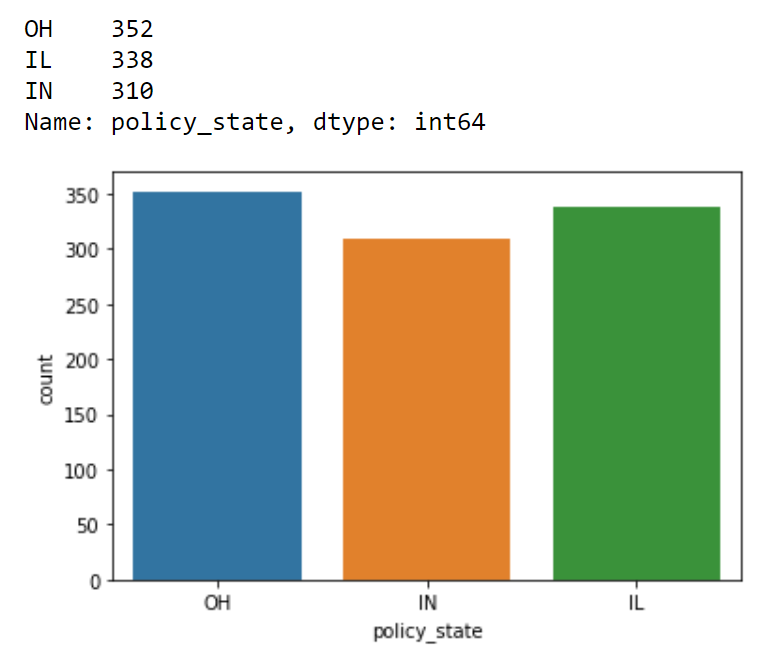
Companies which have small datasets is more likely to succeed in machine learning than the companies having huge datasets.

There are some null values present in the given dataset. The number of null values present is shown below:

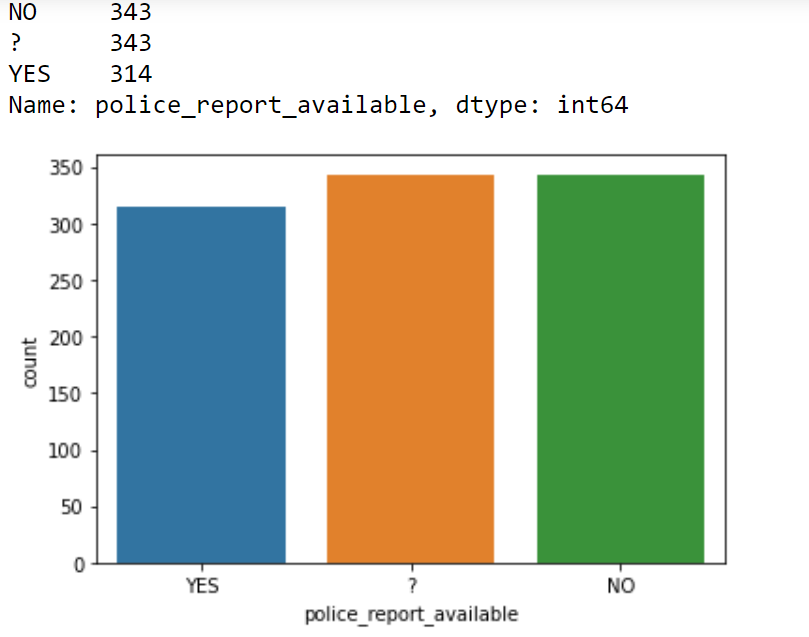


Exploratory Data Analysis

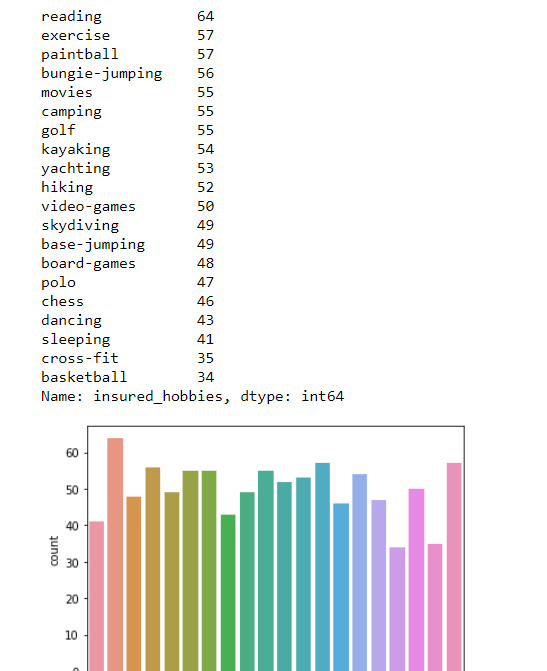
Visualization: The value of the fraud\_reported differs with the policy\_state. The customer having OH policy state has higher tendency to fraud.



The possibilities of having police report have less count compared to not available.

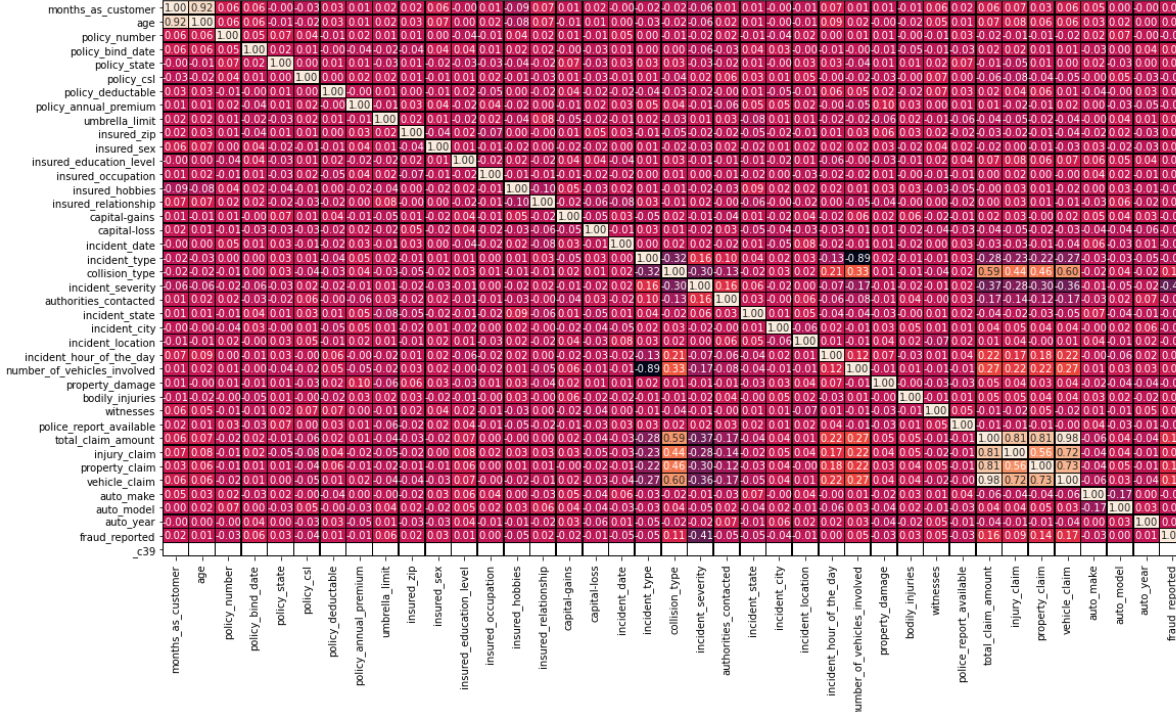


Customers having reading as hobby has high tendency to fraud.



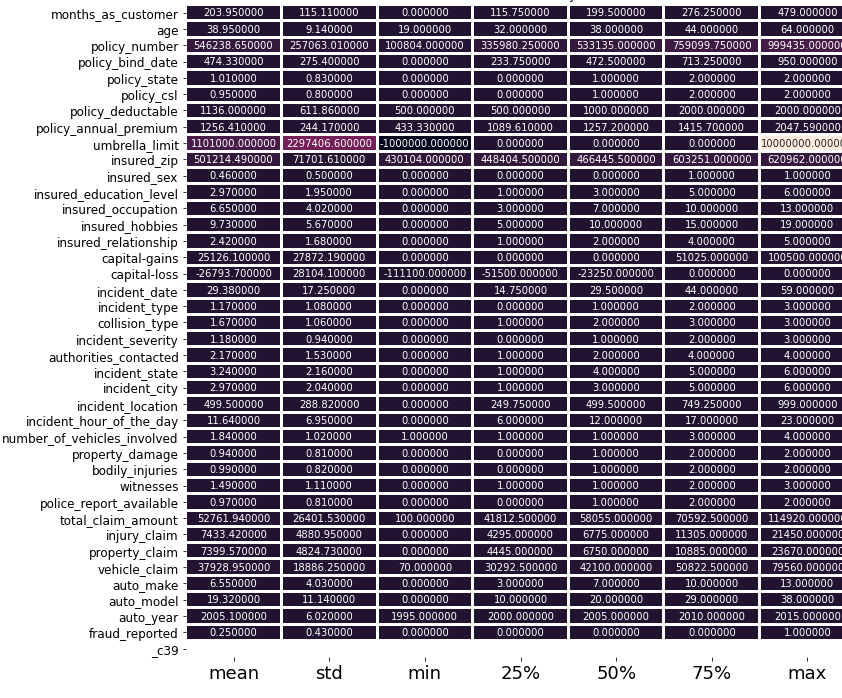
Dependent Variable: The dependent or output variable in the given dataset is the column ‘fraud\_reported’. There are 247 fraud claims and 753 non-fraud claims. 24.7% of the claims were frauds and 75.3% were non-fraudulent claims.

Correlation: Correlation between dependent and independent variables is plotted using heatmap. Column incident\_severity has maximum correlation with 41% correlation.



Describe of the dataset

Describe of the dataset is plotted using the heatmap which shows us the mean, standard deviation, maximum and minimum value of each column in the given dataset.



Pre-Processing Pipelines

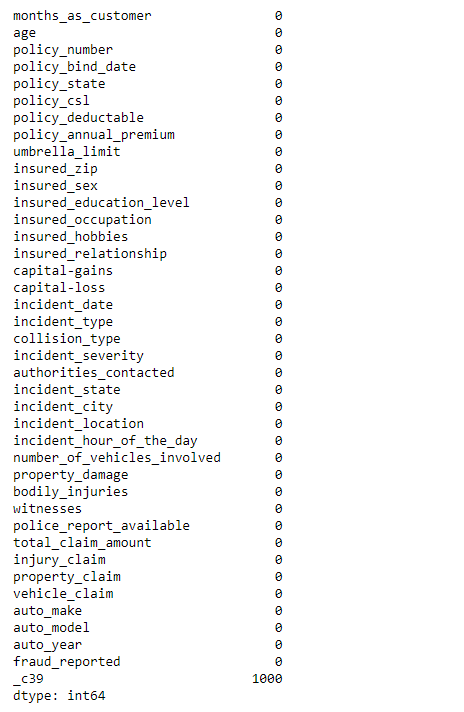
Data pre-processing is an important step in machine learning to get highly accurate and reliable result. Data pre-processing helps in increasing the quality of data by filling in missing data's(NaN values), removing outliers, scaling the data.

There are many steps involved in data pre-processing:

* Data Cleaning helps to impute the missed values and removing outliers from the dataset.
* Data Integration integrates data from multiple sources into single dataset.
* Data Transformation such as normalization helps in improving the accuracy and efficiency of algorithms involved in machine learning.
* Data Reduction reduces the data size by dropping out redundant features using feature selection and feature extraction techniques.

Treating null values

Sometimes there can be certain columns which may contain the null values used to indicate the missing or unknown values. In our dataset the null values are present in \_c39 column with 1000 null values.



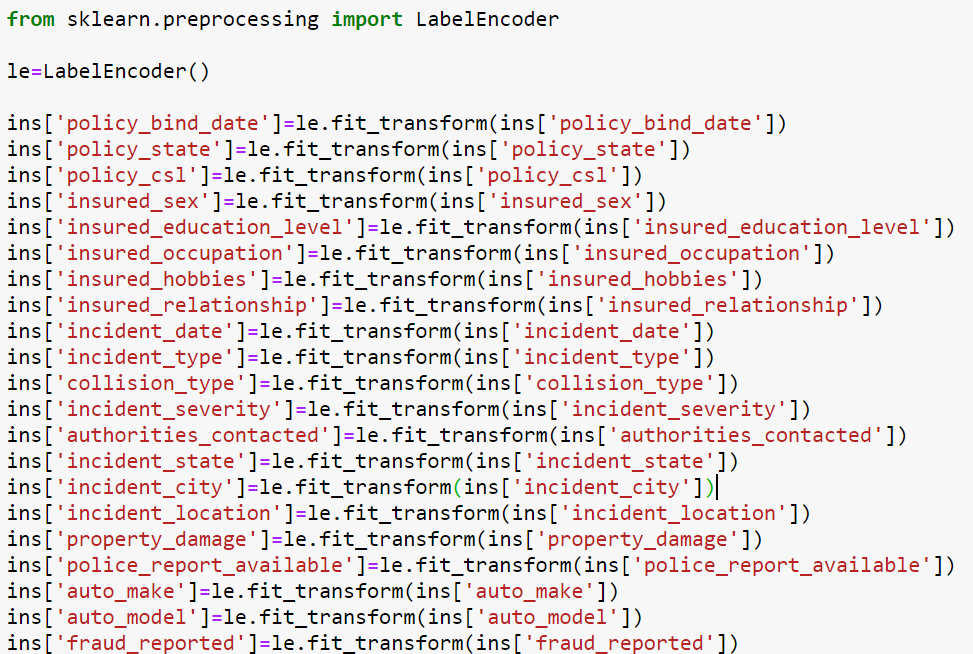
Converting labels into numeric

In machine learning, we usually deal with datasets which contain multiple labels in one or more column. These labels can be in the form of alphabets or numbers. To make the data understandable or in human readable form, the training data is often labelled in words.

In our dataset there are columns with categorical values. The columns like policy\_bind\_date, policy\_state, policy\_csl, insured\_sex, insured\_education\_level, insured\_occupation, insured\_hobbies, insured\_relationship, incident\_date, incident\_type, collision\_type, incident\_severity, authorities\_contacted, incident\_state, incident\_city, incident\_location, property\_damage, police\_report\_available, auto\_make, auto\_model, fraud\_reported. These columns are converted using Label Encoder.

Label Encoder refers to converting the labels into numeric form so as to convert it into the machine-readable form. It is an important step in data pre-processing.

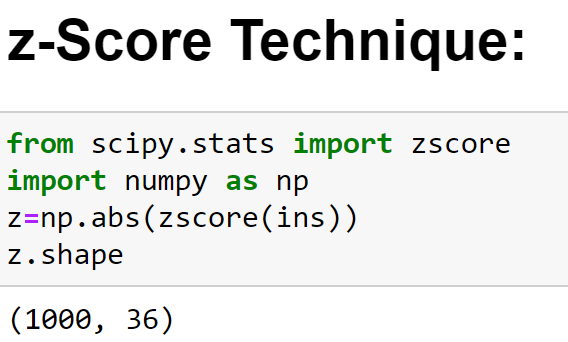
Label encoding in python can be imported from Sklearn library. Sklearn provides a very efficient tool for encoding.



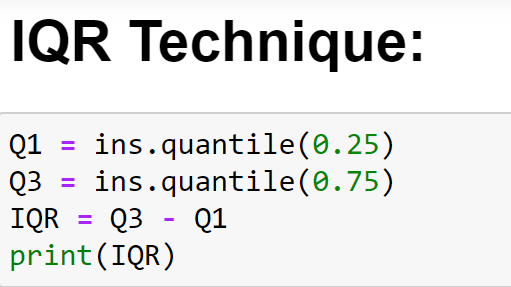
Outliers are datapoints that are far away from other similar points. They may be due to variability in the measurement or may be due to experimental errors. Outliers should be excluded from the dataset in order to get the efficient and accurate prediction.

Methods to remove outliers:

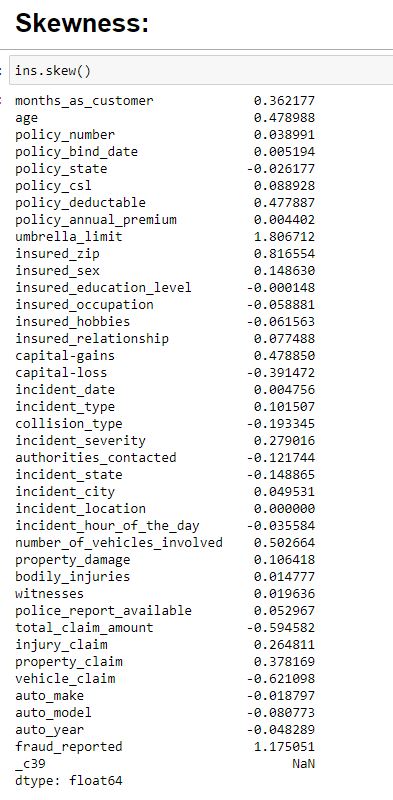
* Z-score: Call scipy.stats.zscore() with the given data-frame as its argument to get a numpy array containing the z-score of each value in a data-frame. Call numpy.abs with the previous result to convert each element in the data-frame to its absolute value.



* Interquartile(IQR) range: It can be used to remove outliers present in the dataframe. IQR can be calculated by using scipy.stats.iqr module



Skewness: Skewness is a measure of the asymmetry of the probability distribution of a real valued random variable about its mean.

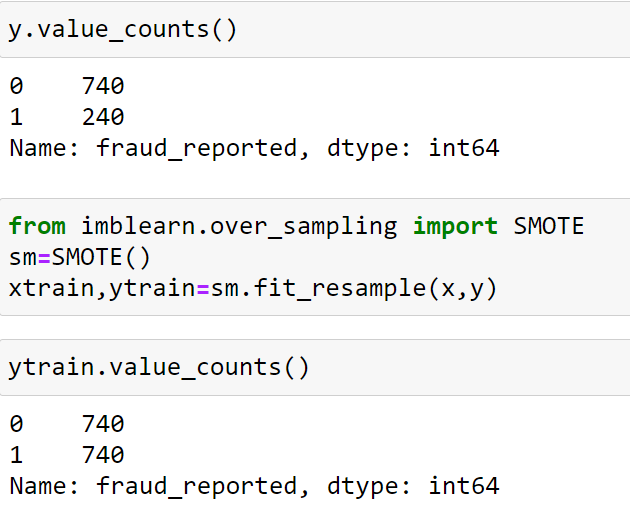


Balancing imbalanced data

There are different algorithms present to balance the data. We use SMOTE() algorithm to balance the data.

SMOTE algorithm works in 4 simple steps:

* Choose a minority class as a minority vector.
* Find its k-nearest neighbors.
* Choose one of its neighbors and place synthetic point anywhere on the line joining the point under consideration and its chosen neighbors.
* Repeat the steps until the data is balanced.



The original shape of our data was 753 for non-fraud and 247 for fraud claims. The SMOTE algorithm balances our data with the highest number of values present in it.

Building Machine Learning Models

For building machine learning models there are several models present inside the Sklearn module.

Sklearn module provides two types of models i.e regression and classification.Our datasets target is to predict whether fraud is reported or not. So, for this kind of problem, we use classification models.

Before fitting our dataset to its model first we have to separate the predictor variable and the target variable, then we pass this variable to the train\_test\_split method to create a random test and train subset.

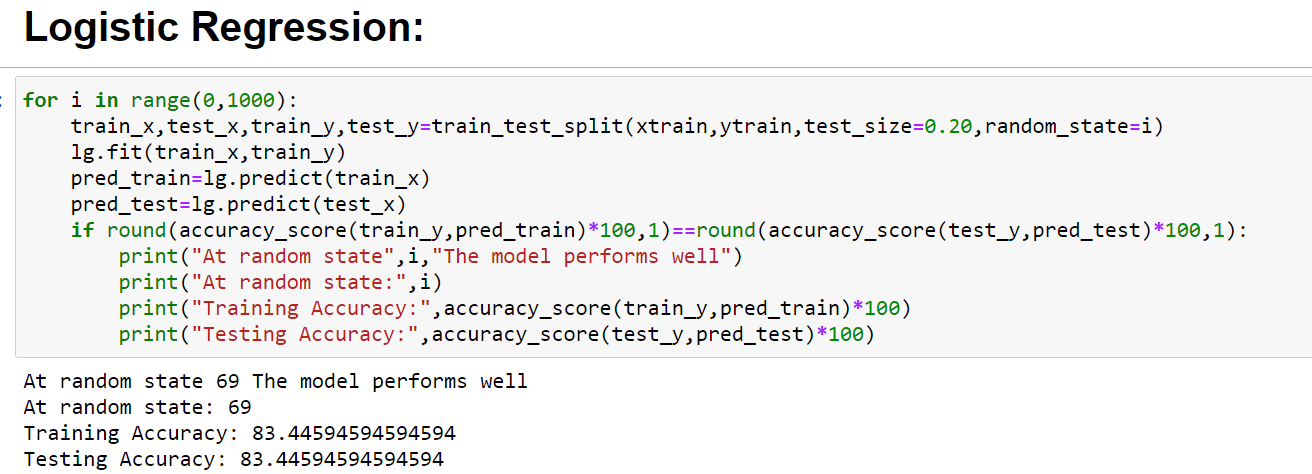
Train\_test\_split is a function in sklearn model selection for splitting data arrays into two subsets for training data and testing data. With this function, there is no need to divide the dataset manually. By default, Sklearn train\_test\_split will make random splitting for the two subsets. However, we can also specify a random state for the operation. It gives four outputs x\_train, x\_test, y\_train and y\_test. The x\_train and x\_test contains the training and testing predictor variables while y\_train and y\_test contains the training and testing target variable.

After performing train\_test\_split we have to choose the models to pass the training variable.

We can build as many models as we want to compare the accuracy given by these models and to select the best model among them.

In this project we have selected 8 models:

* Logistic Regression from sklearn.linear\_model: Logistic regression is a supervised learning classification algorithm used to predict the probability of a target or dependent variable. The type of target or dependent variable is binary, which means there would be only two possible cases 1(success/yes/true) or 0(fail/no/false). It is one of the simplest machine learning algorithms that can be used for various classification problems such as diabetes prediction, cancer detection etc.



* DecisionTreeClassifier from sklearn.tree: Decision Tree Classifier is a non\_parametric supervised learning method used for classification. The goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features. Decision trees can be constructed by an algorithmic approach that can split the dataset in different ways based on different conditions. The two main entities of a tree are decision nodes, where the data is split leaves, where we get the outcome.

Steps involved in this classification process:

1. Determine the root of the Tree.

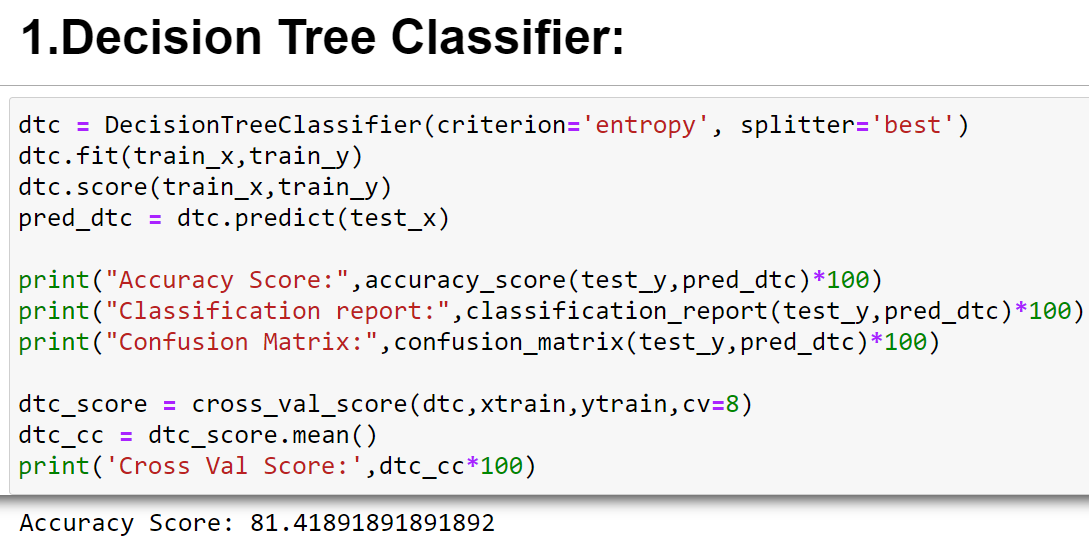
2. Calculate Entropy for the classes.

3. Calculate Entropy after split for each attribute.

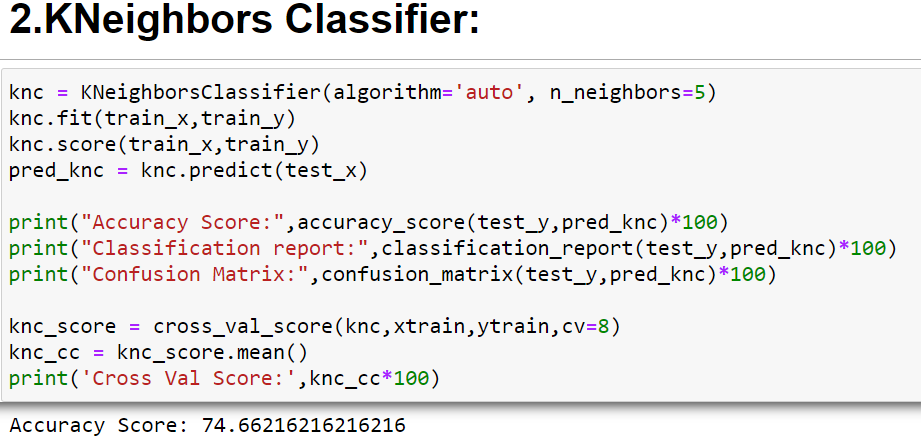
4. Calculate information gain for each split.

5. Perform the split.

6. Perform the further splits until the outcome is produced.



* KNeighborsClassifier from sklearn.neighbors: KNeighbors classifier is a predictive model where the K in the name of this classifier represents the k nearer neighbors, where k is an integer value specified by the user. Hence as the name suggests, this classifier implements learning based on the k nearest neighbors. The choice of the value of k is dependent on data.



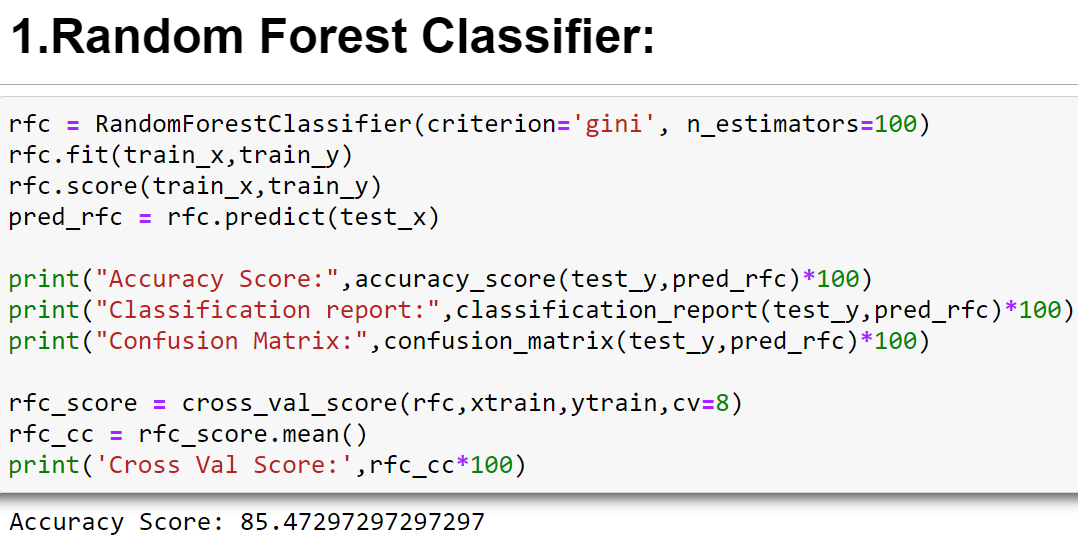
* SupportVectorClassifier(SVC) from sklearn.svm: Support vector classifier is a supervised machine learning algorithm which is used for classification problems. It uses a technique called the kernel trick to transform data and then based on these transformations it finds an optimal boundary between the possible outputs. After giving a svc model sets of labelled training data for each category, they are able to categorize new text.



* RandomForestClassifier from sklearn.ensemble: Random Forest classifier is a supervised learning algorithm used for classification problems. It is the most flexible and easy to use algorithm. Random forest is a meta estimator that fits a number of decision tree classifiers on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over fitting. The sub-sample size is controlled by max\_samples parameter.

Steps involved in this process are:

1. Select random samples from a given dataset.
2. Construct a decision tree for each sample and get a prediction result from each decision tree.
3. Perform a vote for each predicted result.
4. Select the prediction result with the most votes as the final prediction.

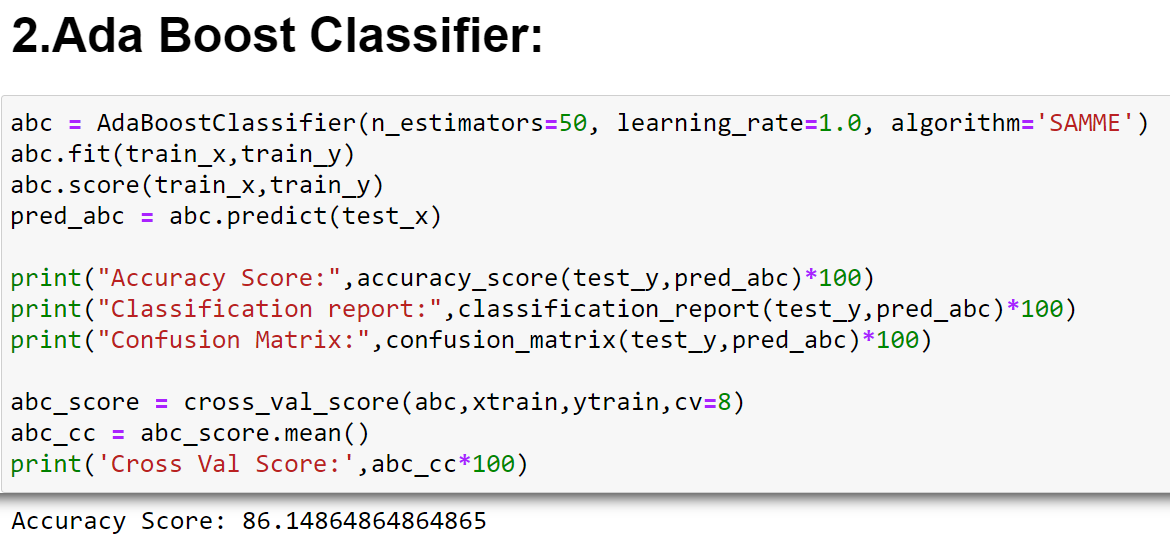


* AdaBoostClassifier from sklearn.ensemble: An Ada-boost or Adaptive Boosting is one of ensemble boosting classifier is a meta-estimator that begins by fitting a classifier on the original dataset and then fits additional copies of the classifier on the same dataset but where the weights of incorrectly classified instances are adjusted such that subsequent classifiers focus more on difficult cases. Adaboost should meet two conditions:

1. The classifier should be trained interactively on various weighed training examples.
2. In each iteration, it tries to provide an excellent fit for these examples by minimizing training error.

AdaBoostClassifier works in the following steps:

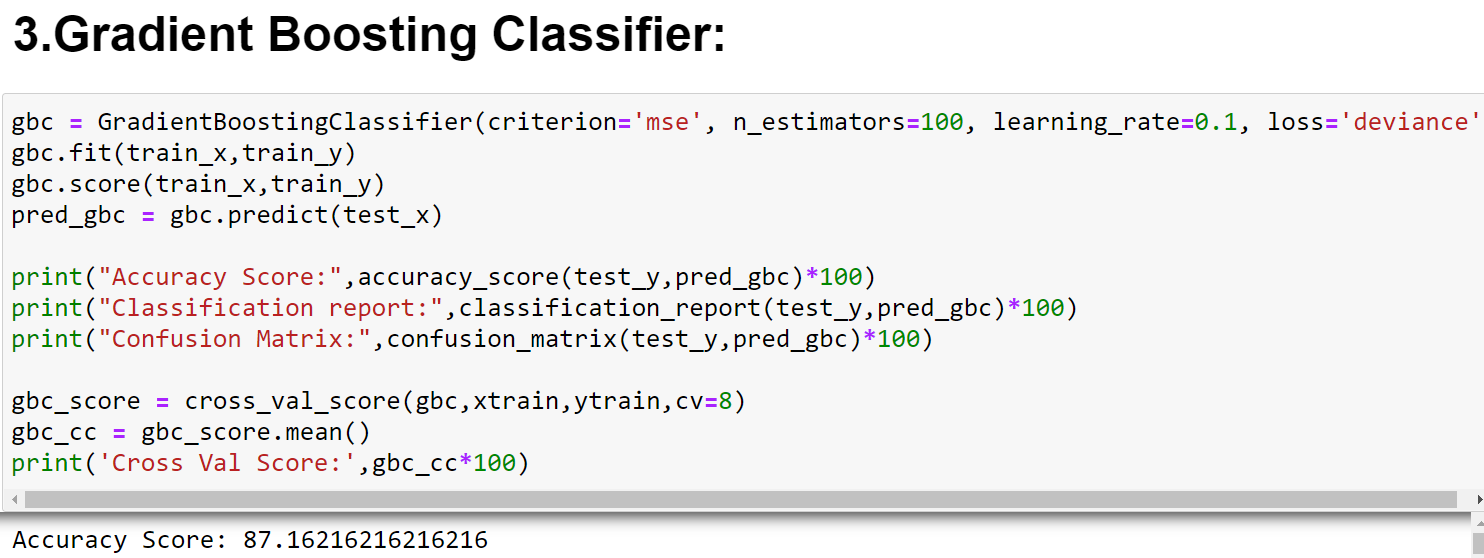
1. Initially, Adaboost selects a training subset randomly.
2. It iteratively trains the AdaBoost machine learning model by selecting the training set based on the accurate prediction of the last training.
3. It assigns the higher weight to wrong classified observations so that in the next iteration these observations will get the high probability for classification.
4. It assigns the weight to the trained classifier in each iteration according to the accuracy of the classifier. The more accurate classifier will get high weight.
5. This process iterates until the complete training data fits without any error or until reached to the specified maximum number of estimators.
6. To classify, perform a vote across all of the learning algorithms you built.



* GradientBoostingClassifier from sklearn.ensemble: Gradient boosting classifiers are a group of machine learning algorithms that combine many weak learning models together to create a strong predictive model. Decision trees are usually used when doing gradient boosting.

Steps followed while performing gradient boosting classification process:

1. Fit the model.
2. Tune the model's parameters and hyperparameters.
3. Make predictions.
4. Interpret the results.



* VotingClassifier from sklearn.ensemble: A voting classifier is a machine learning model that trains on an ensemble of numerous models and predicts an output based on their highest probability of chosen class as the output. It simply aggregates the findings of each classifier passed into voting classifier and predicts the output class based on the highest majority of voting. Instead of creating separate models and finding the accuracy for each of them, we create a single model which trains by these models and predicts output based on their combined majority of voting for each output class.

Voting Classifier supports two types of voting:

Hard Voting: In hard voting, the predicted output class is a class with the highest majority of votes.

Soft Voting: In soft voting, the output class is the prediction based on the average of probability given to that class.

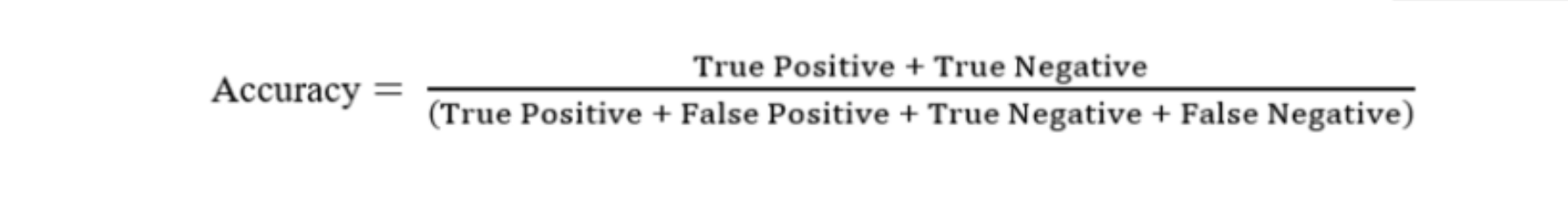
In practical the output accuracy will be more for soft voting as it is the average probability of all the estimators combined.

CONCLUDING REMARKS

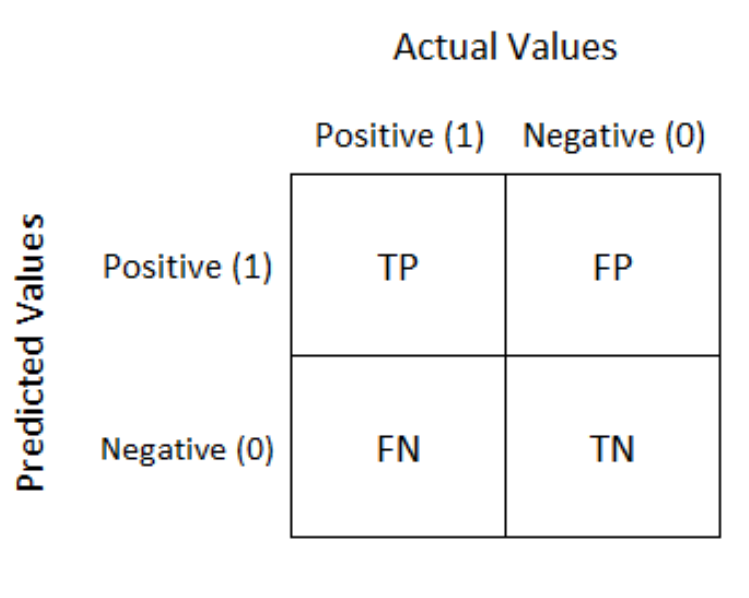
We got our best model i.e. Random Forest Classifier with the accuracy score of 85.47%.

Accuracy: It is one of the most efficient metrics, it is the measure of all the correctly identified cases. It is most used when all the cases are equally important.

The formula used to calculate the accuracy is



Confusion Matrix: A table that is often used to describe the performance of a classification model on a set of data for which the true values are known.



Where, TP (True Positive): The cases were positive and predicted positive.

TN (True Negative): The cases were negative and predicted negative.

FP (False Positive): The cases were positive but predicted negative.

FN (False Negative): The cases were negative but predicted positive.

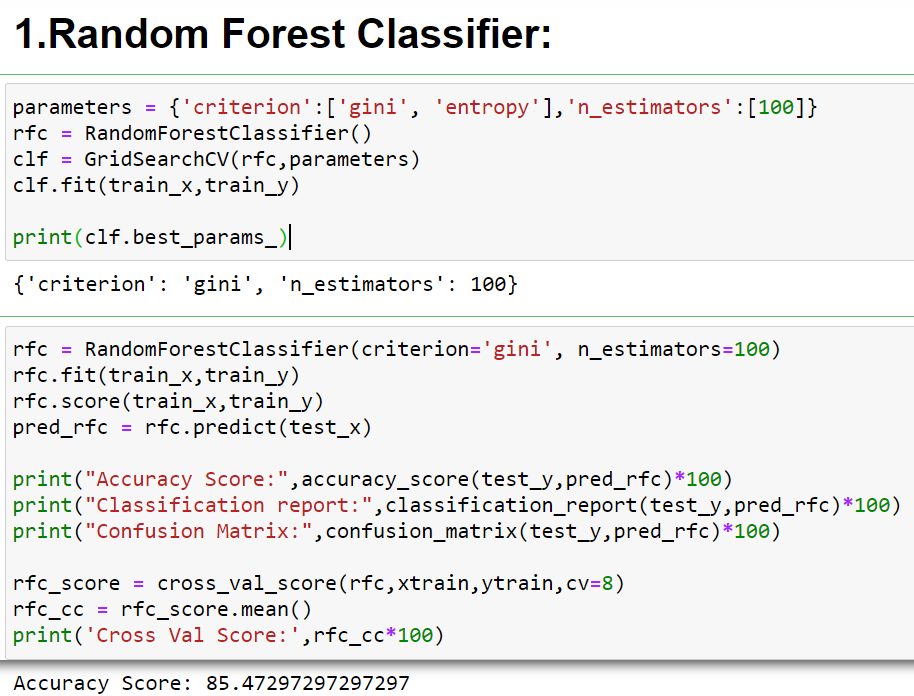
Hyper Parameter Tuning

Hyper parameter optimisation in machine learning intends to deliver the best parameters as measured on a validation set.

We used Grid Search CV for the hyper parameter tuning.

GridSearchCV

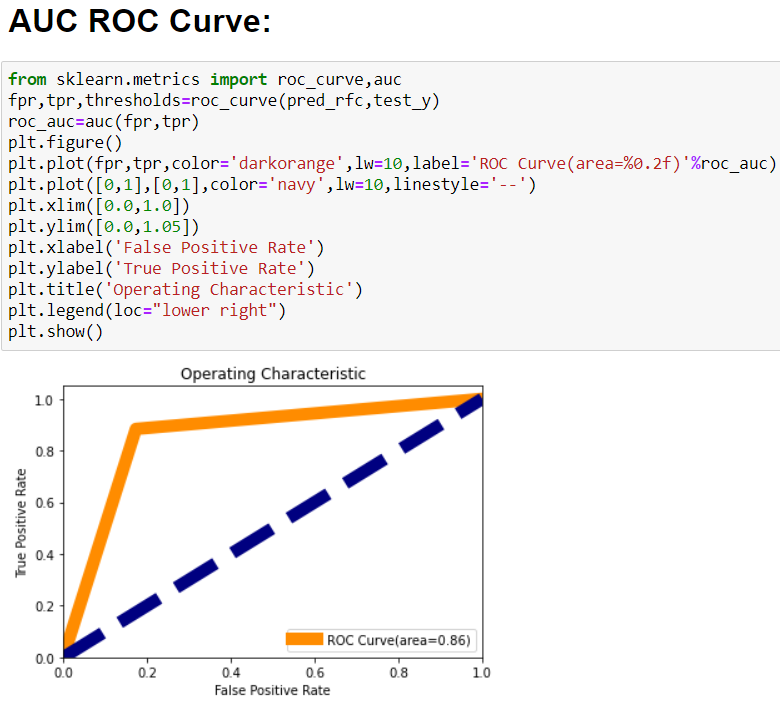
In the GridSearchCV approach, the machine learning model is evaluated for a range of hyper parameter values. This approach is called GridSearchCV because it searches for best set of hyper parameters from a grid of hyper parameter values.



AUC-ROC Curve

It is the performance measurement for the classification problems at various threshold settings. ROC is a probability curve and AUC represents the degree or measure of separability. Higher the AUC, the better the model is at predicting.

The AUC-ROC curve is plotted with TPR against FPR where TPR is on the y-axis and FPR is on x-axis.



Remarks

This project has built a model that can detect auto insurance fraud. This helps in reducing losses for insurance companies. The challenge behind fraud detection in machine learning is that frauds are far less compared to legit insurance claims.

Eight different classifiers were used in this project: Logistic Regression, Decision Tree Classifier, KNeighbors Classifier, Support Vector Classifier, Random Forest Classifier, Ada Boost Classifier, Gradient Boosting Classifier, Voting Classifier.

The best and final model was Random Forest Classifier that yield an accuracy score of 0.85 and an AUC-ROC curve of 0.86. The model performed excellent.