**CENTRE OF DEVELOPMENT OF ADVANCED COMPUTING, MOHALI**



**ADVANCED COURSE ON DATA SCIENCE AND ANALYTICS (HP-KVN)**

**“PROJECT REPORT ON CREDIT CARD FRAUD DETECTION USING MACHINE LEARNING**

**CASE STUDY”**

## SUBMITTED TO: SUBMITTED BY:

MR. SANJAY MADAN Arpit shrivastav HARSH THAKUR

# ACKNOWLEDGEMENT

It is our proud privilege and duty to acknowledge the kind of help and guidance received from several people in preparation of this report. It would not have been possible to prepare this report in this form without their valuable help, cooperation and guidance.

First, and foremost, we wish to record our sincere gratitude to C-DAC Coordinators for their constant support and encouragement in preparation of this report and for making available videos and interface facilities needed to prepare this report.

Their contributions and technical support in preparing this report are greatly acknowledged.

# TABLE OF CONTENTS

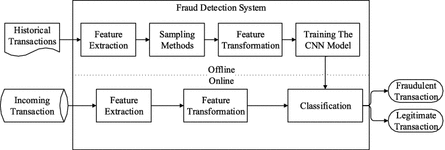
|  |  |  |
| --- | --- | --- |
| S.NO | TOPICS | PAGE NO. |
| 1. | Introduction | 4 |
| 2. | Problem Statement   * Project objectives | 5 |
| 3. | Dataset description | 6 |
| 4. | Libraries | 7-12 |
| 5. | Methodology   * Logistic regression | 13-20 |
| 6. | Implementation   * Preprocessing of data   + Feature scaling   + Train-Test split * Building the model and evaluation * Confusion matrix | 21-44 |
| 7. | Observation | 45 |
| 8. | References | 46 |

# 

# INTRODUCTION

Credit card generally refers to a card that is assigned to the customer (cardholder), usually allowing them to purchase goods and services within credit limit or withdraw cash in advance. Credit card provides the cardholder an advantage of the time, i.e., it provides time for their customers to repay later in a prescribed time, by carrying it to the next billing cycle.

Credit card frauds are easy targets. Without any risks, a significant amount can be withdrawn without the owner’s knowledge, in a short period. Fraudsters always try to make every fraudulent transaction legitimate, which makes fraud detection very challenging and difficult task to detect.



# Fraud Detection Process

# PROBLEM STATEMENT

The rapid growth in E-Commerce industry has lead to an exponential increase in the use of credit cards for online purchases and consequently they has been surge in the fraud related to it. In recent years, for banks has become very difficult for detecting the fraud in credit card system. Machine learning plays a vital role for detecting the credit card fraud in the transactions.

## To detect wether a credit card Transaction is Fraud or Not Using machine Learning

It is vital that credit card companies are able to identify fraudulent credit card transactions so that customers are not charged for items that they did not purchase. Such problems can be tackled with Data Science and its importance, along with Machine Learning, cannot be overstated. This project intends to illustrate the modelling of a data set using machine learning with Credit Card Fraud Detection. The Credit Card Fraud Detection Problem includes modelling past credit card transactions with the data of the ones that turned out to be fraud. This model is then used to recognize whether a new transaction is fraudulent or not. Our objective here is to detect 100% of the fraudulent transactions while minimizing the incorrect fraud classifications. Credit Card Fraud Detection is a typical sample of classification. In this process, we have focused on analysing and pre-processing data sets as well as the deployment of multiple anomaly detection algorithms such as Local Outlier Factor and Isolation Forest algorithm on the PCA transformed Credit Card Transaction data.

## OBJECTIVE

In this project we are going to predict whether a credit card is fraud or not using Machine Learning. \*\*identify suspicious events and report them to an analyst while letting normal transactions be automatically processed. \*\*

1. Applying ML model to detect Fraud
2. Check which algorithm Supervised or Unsupervised works better

# DATASET

## Dataset Link: <https://www.kaggle.com/mlg-ulb/creditcardfraud>

* The dataset contains transactions made by credit cards in September 2013 by European cardholders. This dataset presents transactions that occurred in two days, where we have 492 frauds out of 284,807 transactions.
* The dataset is highly unbalanced, the positive class (frauds) account for 0.172% of all transactions. It contains only numerical input variables which are the result of a PCA transformation.

## FEATURES

* V1, V2, ... V28 are the principal components obtained with PCA, the only features which have not been transformed with PCA are 'Time' and 'Amount'.
* Feature 'Time' contains the seconds elapsed between each transaction and the first transaction in the dataset.
* The feature 'Amount' is the transaction Amount, this feature can be used for example-dependant cost-sensitive learning.
* Feature 'Class' is the response variable and it takes value 1 in case of fraud and 0 otherwise.

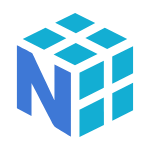
**Number of Rows**  : **284807**

**Number of Columns** : **31**

|  |  |
| --- | --- |
| Features | Description |
| **Time** | The second elapsed between Each Transactions. |
| **V1-V28** | The principal components obtained with PCA Transformations |
| **Amount** | The transaction Amount |
| **Class** | This is response variable and it takes value 1 in case of fraud and 0 otherwise. |

# LIBRARIES

### NUMPY



NumPy is the fundamental package for scientific computing in Python. It is a

Python library that provides a multidimensional array object, various derived

*import numpy as np*

objects (such as masked arrays and matrices), and an assortment of routines for fast operations on arrays, including mathematical, logical, shape manipulation, sorting, selecting, I/O, discrete Fourier transforms, basic linear algebra, basic statistical operations, random simulation and much more.

### PANDAS

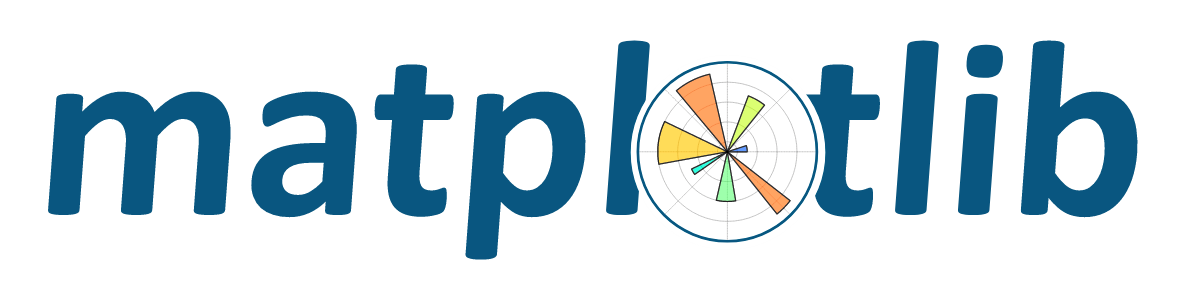


When working with tabular data, such as data stored in spreadsheets or databases, pandas is the right tool for you. pandas will help you to explore, clean, and process your data. In pandas, a data table is called a DataFrame.

*import pandas as pd*

To load the pandas package and start working with it, import the package. The community agreed alias for pandas is pd, so loading pandas as pd is assumed standard practice for all of the pandas documentation.

### MATPLOTLIB



Matplotlib is a library for making 2D plots of arrays in Python. Although it has its origins in emulating the MATLAB®1 graphics commands, it is independent of MATLAB, and can be used in a Pythonic, object oriented way.

import matplotlib.pyplot as plt

Although Matplotlib is written primarily in pure Python, it makes heavy use of NumPy and other extension code to provide good performance even for large arrays.

### SEABORN



Seaborn is built on top of Python’s core visualisation library Matplotlib. It is meant to serve as a complement, and not a replacement. However, Seaborn comes with some very important features. Let us see a few of them here .

import seaborn as sns

The features help in -

* Built in themes for styling matplotlib graphics
* Visualizing univariate and bivariate data
* Fitting in and visualising linear regression models
* Plotting statistical time series data
* Seaborn works well with Num Py and Pandas data structures
* It comes with built in themes for styling Matplotlib graphics

### SCIKIT-LEARN



### SKLEARN.METRICS

from sklearn.metrics import classification\_report , confusion\_matrix

The [**sklearn.metrics**](https://scikit-learn.org/stable/modules/classes.html#module-sklearn.metrics) module implements several loss, score, and utility functions to measure classification performance. Some metrics might require probability estimates of the positive class, conf idence values, or binary decisions values. Most implementations allow each sample to provide a weighted contribution to the overall score, through the sample\_weight parameter.

### SKLEARN.ENSEMBLE

*from sklearn.ensemble import RandomForestClassifier*

A random forest classifier.

A 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 with the max\_samples parameter if bootstrap=True (default), otherwise the whole dataset is used to build each tree.

### SKLEARN.NEIGHBORS

*from sklearn.neighbors KNeighborsClassifier*

scikit-learn implements two different nearest neighbors classifiers: [**KNeighborsClassifier**](https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html#sklearn.neighbors.KNeighborsClassifier) implements learning based on the k nearest neighbors of each query point, where k is an integer value specified by the user. [**RadiusNeighborsClassifier**](https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.RadiusNeighborsClassifier.html#sklearn.neighbors.RadiusNeighborsClassifier) implements learning based on the number of neighbors within a fixed radius r of each training point, where r is a floating-point value specified by the user.

The k-neighbors classification in [**KNeighborsClassifier**](https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html#sklearn.neighbors.KNeighborsClassifier) is the most commonly used technique. The optimal choice of the value k is highly data-dependent: in general a larger k suppresses the effects of noise, but makes the classification boundaries less distinct.

### SKLEARN.TREE

*from sklearn.tree import DecisionTreeClassifier*

**Decision Trees (DTs)** are a non-parametric supervised learning method used for [classification](https://scikit-learn.org/stable/modules/tree.html#tree-classification) and [regression](https://scikit-learn.org/stable/modules/tree.html#tree-regression). 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. A tree can be seen as a piecewise constant approximation.

For instance, in the example below, decision trees learn from data to approximate a sine curve with a set of if-then-else decision rules. The deeper the tree, the more complex the decision rules and the fitter the model.

### SKLEARN. LINEAR\_MODEL

*from linear\_model import LogisticRegression*

Logistic Regression (aka logit, MaxEnt) classifier.

In the multiclass case, the training algorithm uses the one-vs-rest (OvR) scheme if the ‘multi\_class’ option is set to ‘ovr’, and uses the cross-entropy loss if the ‘multi\_class’ option is set to ‘multinomial’. (Currently the ‘multinomial’ option is supported only by the ‘lbfgs’, ‘sag’, ‘saga’ and ‘newton-cg’ solvers.)

This class implements regularized logistic regression using the ‘liblinear’ library, ‘newton-cg’, ‘sag’, ‘saga’ and ‘lbfgs’ solvers. **Note that regularization is applied by default**. It can handle both dense and sparse input. Use C-ordered arrays or CSR matrices containing 64-bit floats for optimal performance; any other input format will be converted

### SKLEARN.SVM

*from sklearn.svm import SVM*

**Support vector machines (SVMs)** are a set of supervised learning methods used for [classification](https://scikit-learn.org/stable/modules/svm.html#svm-classification), [regression](https://scikit-learn.org/stable/modules/svm.html#svm-regression) and [outliers detection](https://scikit-learn.org/stable/modules/svm.html#svm-outlier-detection).

The advantages of support vector machines are:

* Effective in high dimensional spaces.
* Still effective in cases where number of dimensions is greater than the number of samples.
* Uses a subset of training points in the decision function (called support vectors), so it is also memory efficient.
* Versatile: different [Kernel functions](https://scikit-learn.org/stable/modules/svm.html#svm-kernels) can be specified for the decision function. Common kernels are provided, but it is also possible to specify custom kernels.

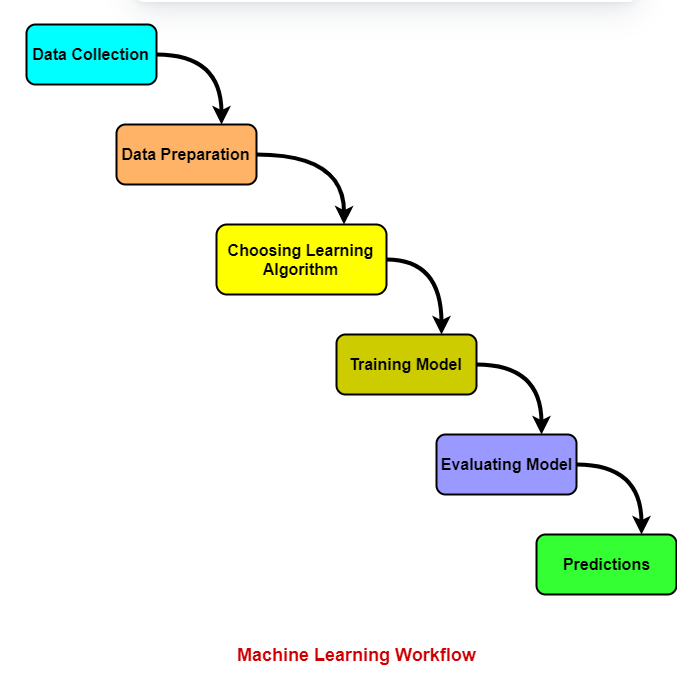
### XGBOOST

*from xgboost import XGBClassifier*

XGBoost is short for “eXtreme Gradient Boosting.” The “eXtreme” refers to speed enhancements such as parallel computing and cache awareness that makes XGBoost approximately 10 times faster than traditional Gradient Boosting. In addition, XGBoost includes a unique split-finding algorithm to optimize trees, along with built-in regularization that reduces overfitting. Generally speaking, XGBoost is a faster, more accurate version of Gradient Boosting.

Boosting performs better than bagging on average, and Gradient Boosting is arguably the best boosting ensemble. Since XGBoost is an advanced version of Gradient Boosting, and its results are unparalleled, it’s arguably the best machine learning ensemble that we have.

## MACHINE LEARNING WORKFLOW



# METHODOLOGY

## LOGISTIC REGRESSION

* Logistic regression is one of the most popular Machine Learning algorithms, which comes under the Supervised Learning technique. It is used for predicting the categorical dependent variable using a given set of independent variables.
* Logistic regression predicts the output of a categorical dependent variable. Therefore, the outcome must be a categorical or discrete value. It can be either Yes or No, 0 or 1, true or False, etc. but instead of giving the exact value as 0 and 1, it gives the probabilistic values which lie between 0 and 1.
* Logistic Regression is much similar to the Linear Regression except that how they are used. Linear Regression is used for solving Regression problems, whereas Logistic regression is used for solving the classification problems.
* In Logistic regression, instead of fitting a regression line, we fit an "S" shaped logistic function, which predicts two maximum values (0 or 1).
* The curve from the logistic function indicates the likelihood of something such as whether the cells are cancerous or not, a mouse is obese or not based on its weight, etc.
* Logistic Regression is a significant machine learning algorithm because it has the ability to provide probabilities and classify new data using continuous and discrete datasets.
* Logistic Regression can be used to classify the observations using different types of data and can easily determine the most effective variables used for the classification.



## Logistic Regression Equation:

The Logistic regression equation can be obtained from the Linear Regression equation. The mathematical steps to get Logistic Regression equations are given below:

* We know the equation of the straight line can be written as:

Logistic Regression in Machine Learning

* In Logistic Regression y can be between 0 and 1 only, so for this let's divide the above equation by (1-y):

Logistic Regression in Machine Learning

* But we need range between -[infinity] to +[infinity], then take logarithm of the equation it will become:

Logistic Regression in Machine Learning

The above equation is the final equation for Logistic Regression.

## KNN ALGORITHM

* K-Nearest Neighbour is one of the simplest Machine Learning algorithms based on Supervised Learning technique.
* K-NN algorithm assumes the similarity between the new case/data and available cases and put the new case into the category that is most similar to the available categories.
* K-NN algorithm stores all the available data and classifies a new data point based on the similarity. This means when new data appears then it can be easily classified into a well suite category by using K- NN algorithm.
* K-NN algorithm can be used for Regression as well as for Classification but mostly it is used for the Classification problems.
* K-NN is a **non-parametric algorithm**, which means it does not make any assumption on underlying data.
* It is also called a **lazy learner algorithm** because it does not learn from the training set immediately instead it stores the dataset and at the time of classification, it performs an action on the dataset.
* KNN algorithm at the training phase just stores the dataset and when it gets new data, then it classifies that data into a category that is much similar to the new data.



## DECISION TREE

* Decision Tree is a **Supervised learning technique**that can be used for both classification and Regression problems, but mostly it is preferred for solving Classification problems. It is a tree-structured classifier, where**internal nodes represent the features of a dataset, branches represent the decision rules** and **each leaf node represents the outcome.**
* In a Decision tree, there are two nodes, which are the **Decision Node** and**Leaf Node.** Decision nodes are used to make any decision and have multiple branches, whereas Leaf nodes are the output of those decisions and do not contain any further branches.
* The decisions or the test are performed on the basis of features of the given dataset.
* ***It is a graphical representation for getting all the possible solutions to a problem/decision based on given conditions.***
* It is called a decision tree because, similar to a tree, it starts with the root node, which expands on further branches and constructs a tree-like structure.
* In order to build a tree, we use the **CART algorithm,** which stands for **Classification and Regression Tree algorithm.**
* A decision tree simply asks a question, and based on the answer (Yes/No), it further split the tree into subtrees.



**IMPORTANT DEFINITIONS :**

**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)**

**GINI INDEX:**

* Gini index is a measure of impurity or purity used while creating a decision tree in the CART (Classification and Regression Tree) algorithm.
* An attribute with the low Gini index should be preferred as compared to the high Gini index.
* It only creates binary splits, and the CART algorithm uses the Gini index to create binary splits.
* Gini index can be calculated using the below formula:

**Gini Index= 1- ∑jPj2**

## RANDOM FOREST

* Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique. It can be used for both Classification and Regression problems in ML. 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 the name suggests, ***"Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset."*** Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, and it predicts the final output.
* **The greater number of trees in the forest leads to higher accuracy and prevents the problem of overfitting.**



## SVM (SUPPORT VECTOR MACHINE)

* Support Vector Machine or SVM is one of the most popular Supervised Learning algorithms, which is used for Classification as well as Regression problems. However, primarily, it is used for Classification problems in Machine Learning.
* The goal of the SVM algorithm is to create the best line or decision boundary that can segregate n-dimensional space into classes so that we can easily put the new data point in the correct category in the future. This best decision boundary is called a hyperplane.
* SVM chooses the extreme points/vectors that help in creating the hyperplane. These extreme cases are called as support vectors, and hence algorithm is termed as Support Vector Machine.



## TYPES OF SVM

* **Linear SVM:** Linear SVM is used for linearly separable data, which means if a dataset can be classified into two classes by using a single straight line, then such data is termed as linearly separable data, and classifier is used called as Linear SVM classifier.
* **Non-linear SVM:** Non-Linear SVM is used for non-linearly separated data, which means if a dataset cannot be classified by using a straight line, then such data is termed as non-linear data and classifier used is called as Non-linear SVM classifier.

**HYPERPLANE AND SUPPORT VECTORS IN THE SVM ALGORITHM:**

* **Hyperplane:** There can be multiple lines/decision boundaries to segregate the classes in n-dimensional space, but we need to find out the best decision boundary that helps to classify the data points. This best boundary is known as the hyperplane of SVM.

The dimensions of the hyperplane depend on the features present in the dataset, which means if there are 2 features (as shown in image), then hyperplane will be a straight line. And if there are 3 features, then hyperplane will be a 2-dimension plane.

We always create a hyperplane that has a maximum margin, which means the maximum distance between the data points.

* **Support Vectors:**

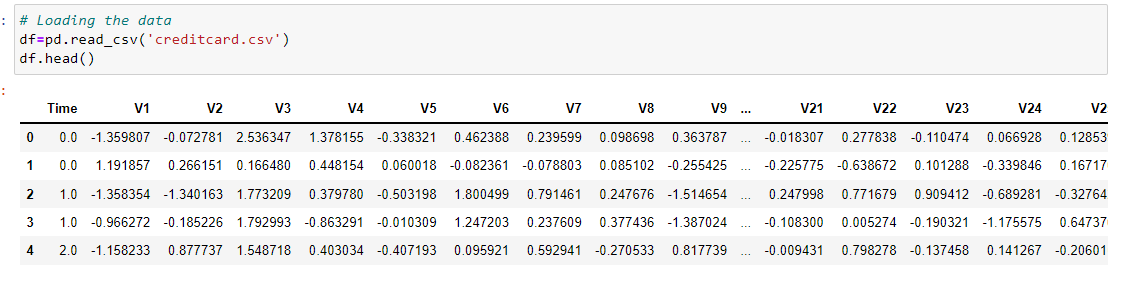
The data points or vectors that are the closest to the hyperplane and which affect the position of the hyperplane are termed as Support Vector. Since these vectors support the hyperplane, hence called a Support vector.

## XGBOOST

* Gradient boosted decision trees are implemented by the XGBoost library of Python, intended for speed and execution, which is the most important aspect of ML (machine learning).
* **XgBoost**: XgBoost (Extreme Gradient Boosting) library of Python was introduced at the University of Washington by scholars. It is a module of Python written in C++, which helps ML model algorithms by the training for Gradient Boosting.
* **Gradient boosting:** This is an AI method utilized in classification and regression assignments, among others. It gives an expectation model as a troupe of feeble forecast models, commonly called decision trees.

# IMPLEMENTATION

## READING DATASET



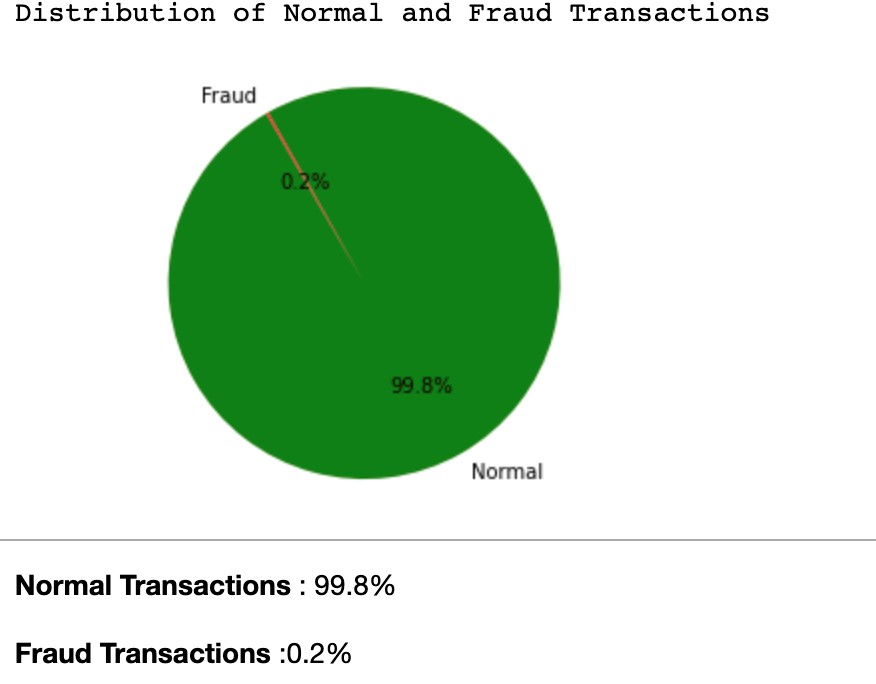
## EXPLORATORY DATA ANALYSIS

Checking null values

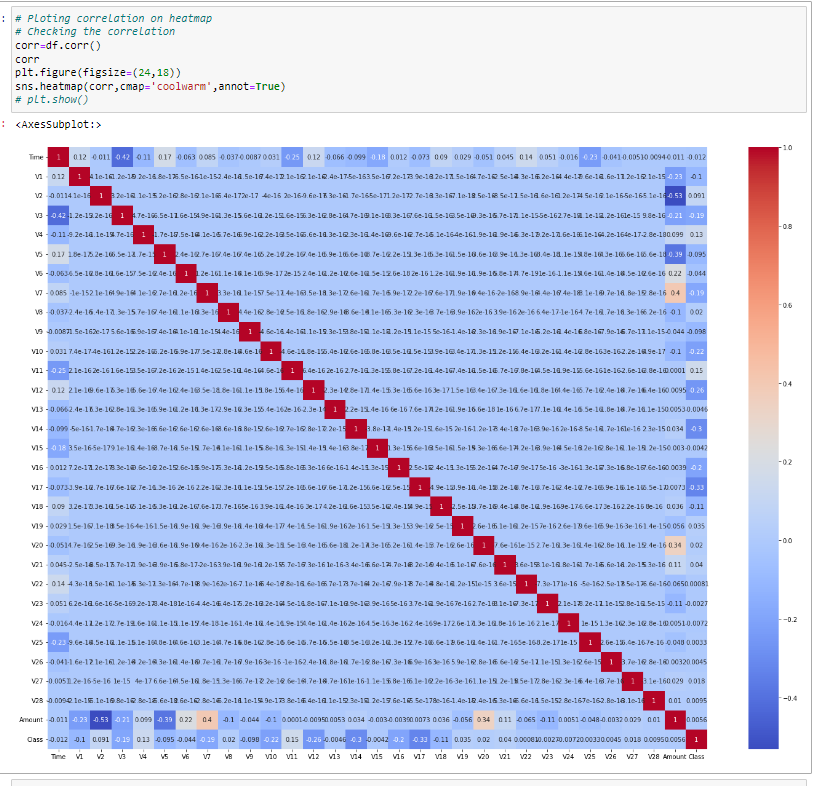


Return False As there are No Null values

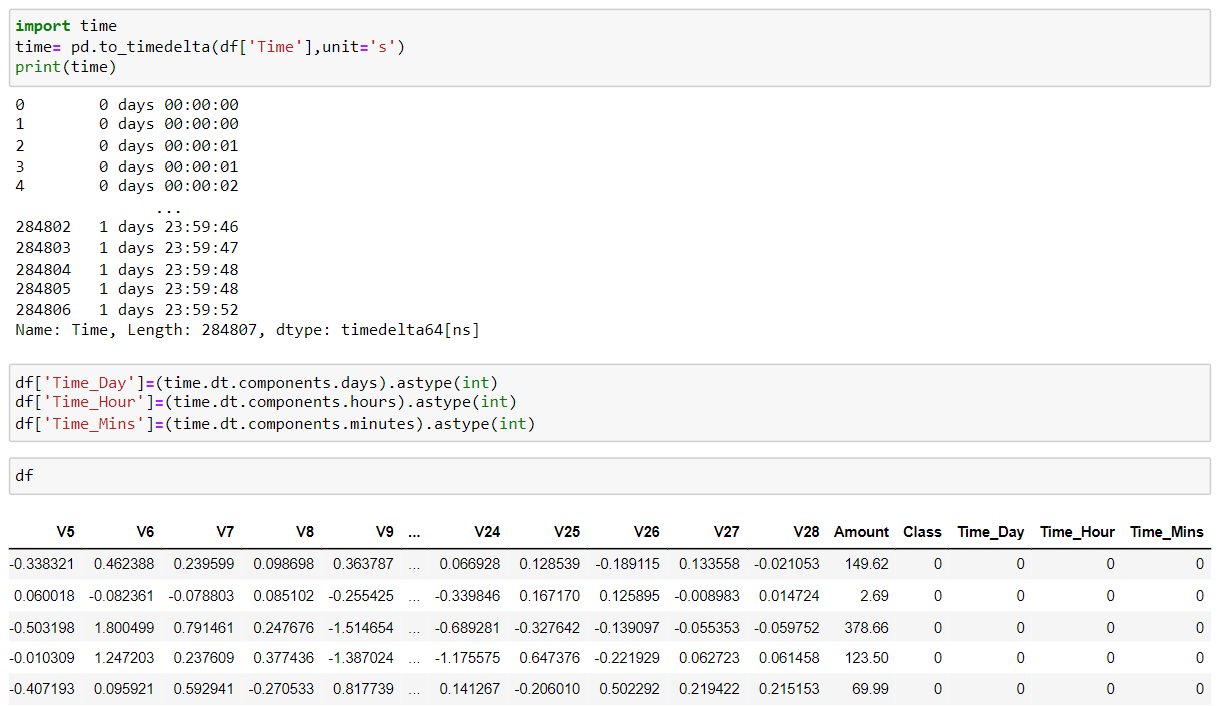
• The given dataset is PCA Transformed and there is no need to check outliers as it is already transformed.

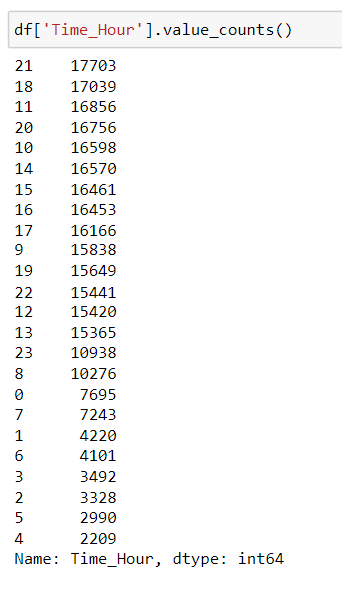


So, the dataset is highly Unbalanced.

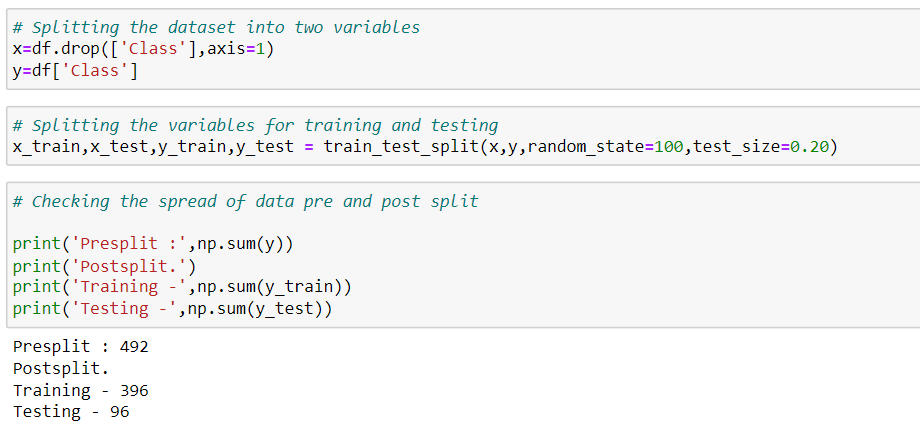


This shows the correlation matrix of the dataset. This matrix explains that attribute class is independent of both the amount and time of the transaction was made. It is even clear from the matrix, the class of the transaction is depending on PCA applied attributes.



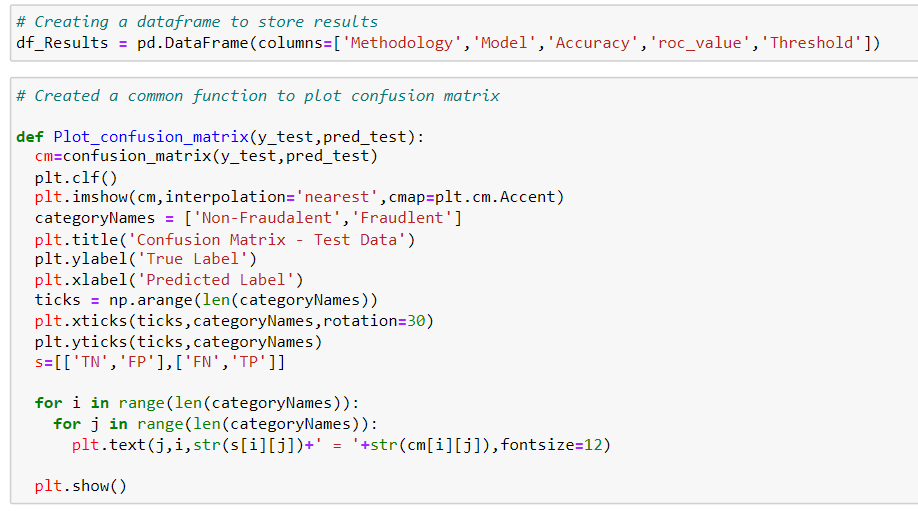


## SPLITTING THE DATA FOR TRAINING AND TESTING

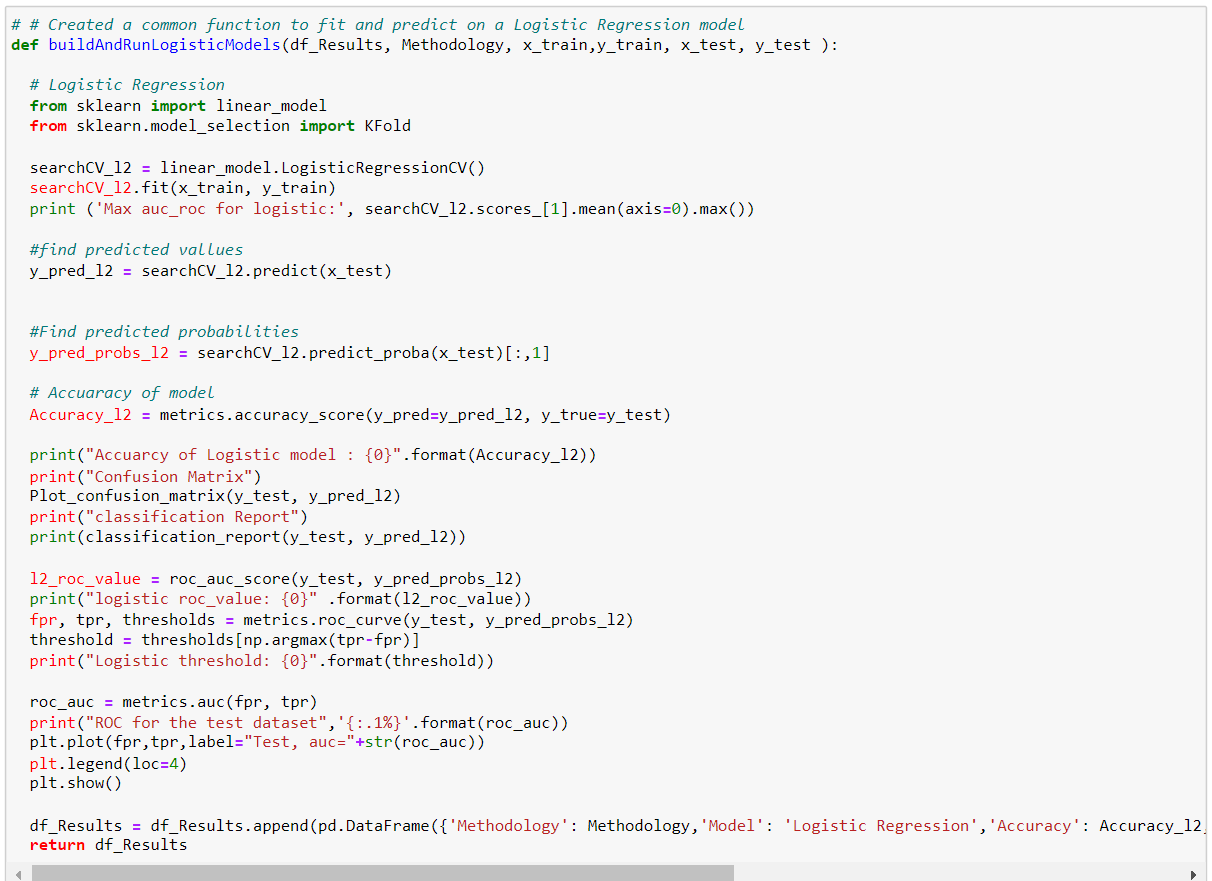


## MODEL BUILDING

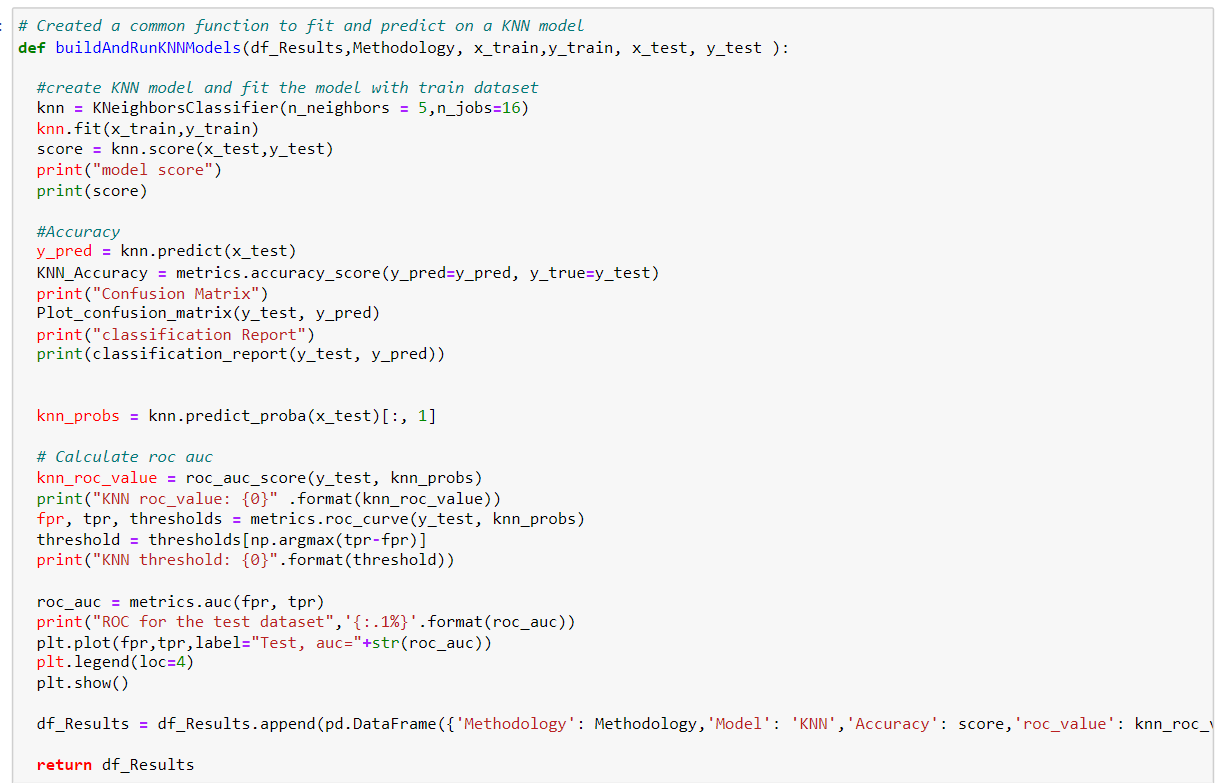
* CONFUSION MATRIX



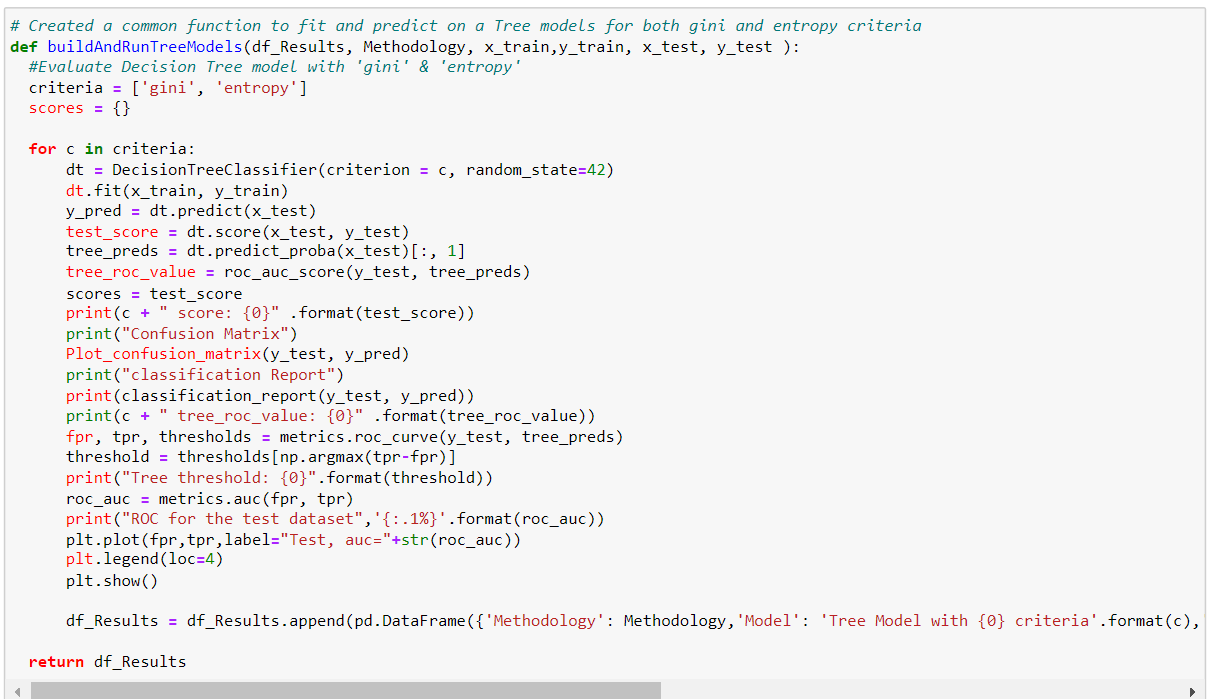
* LOGISTIC MODEL



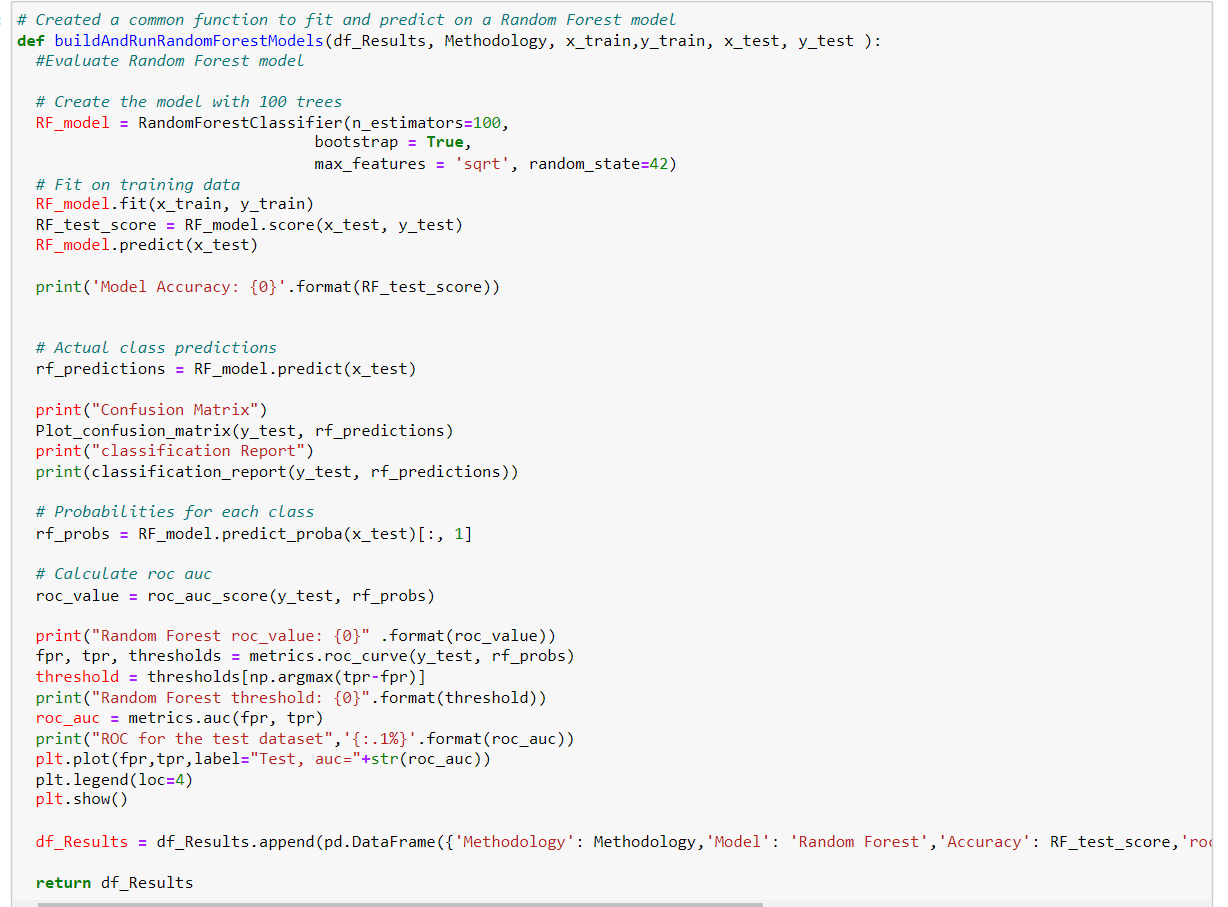
* KNN MODEL



* DECISION TREE MODEL



* RANDOM FOREST MODEL



* XGBOOST MODEL



* SVM MODEL

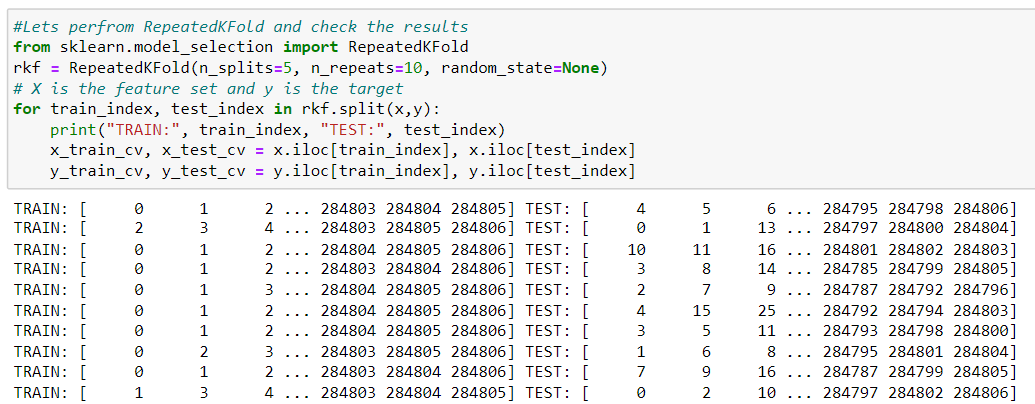


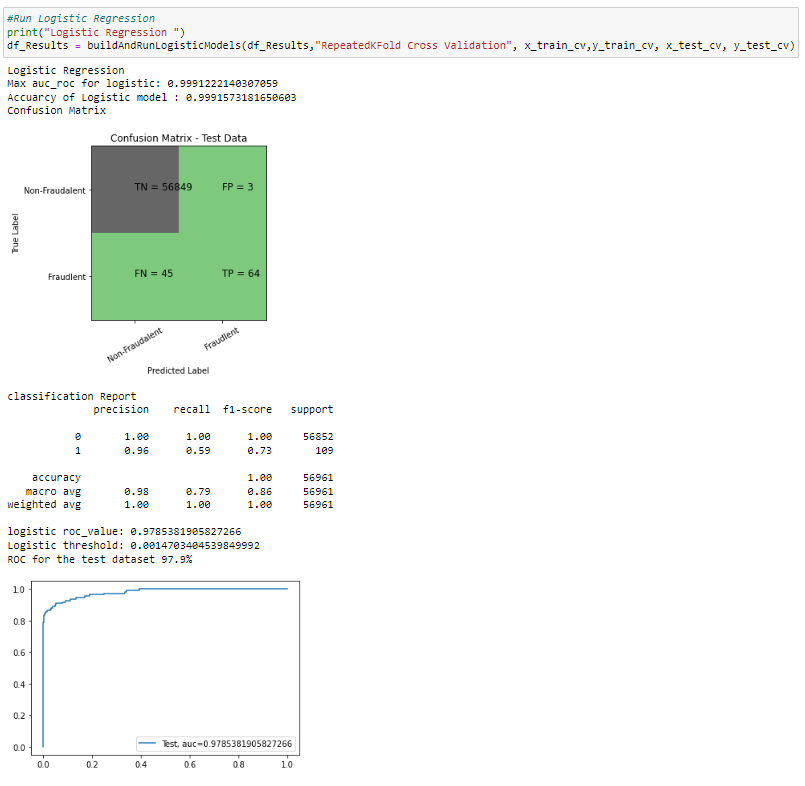
* **K-FOLD CROSS VALIDATION**

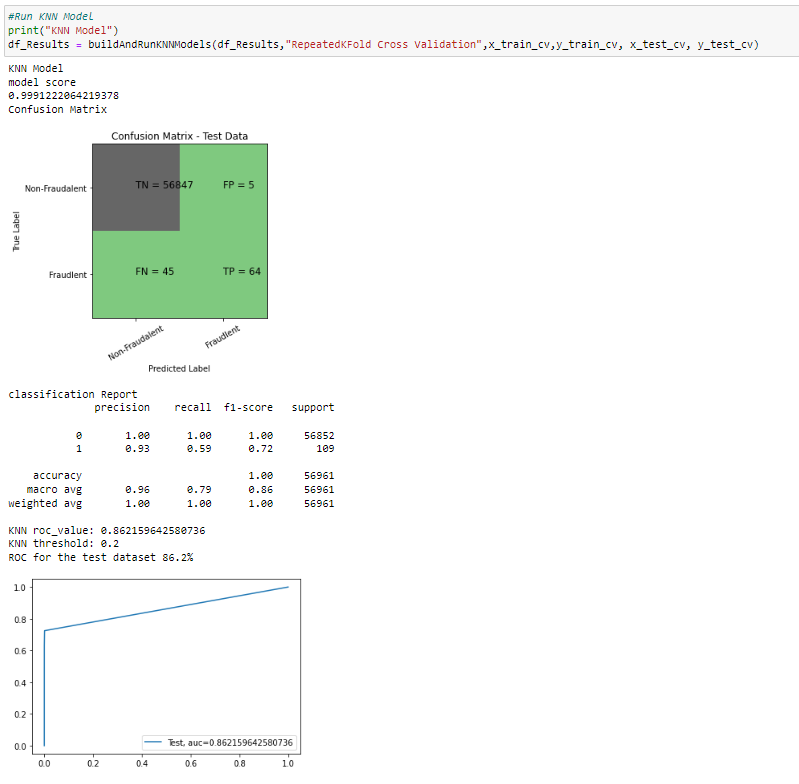
K-fold cross-validation approach divides the input dataset into K groups of samples of equal sizes. These samples are called **folds**. For each learning set, the prediction function uses k-1 folds, and the rest of the folds are used for the test set. This approach is a very popular CV approach because it is easy to understand, and the output is less biased than other methods.

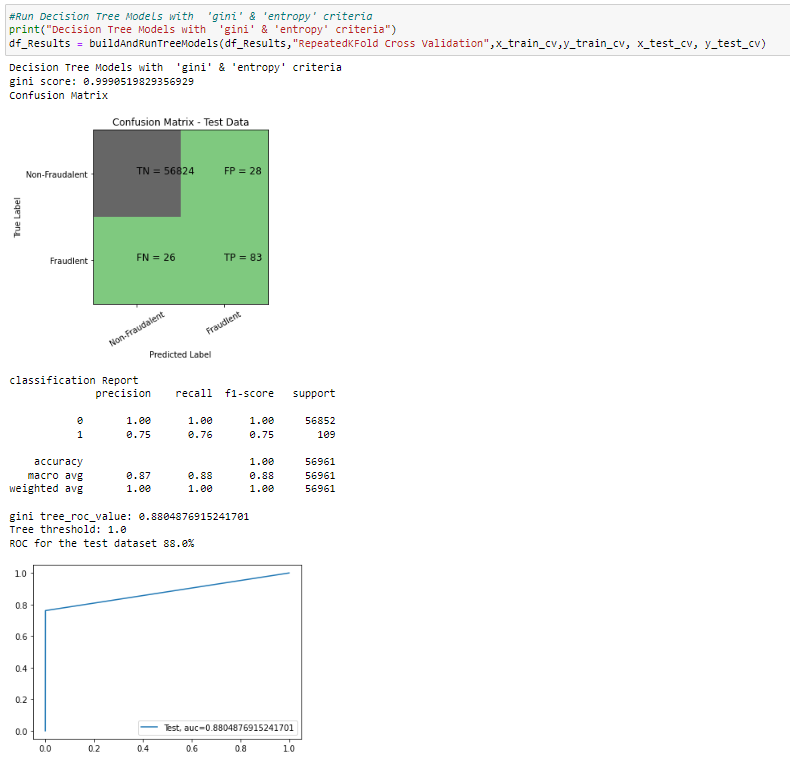
The steps for k-fold cross-validation are:

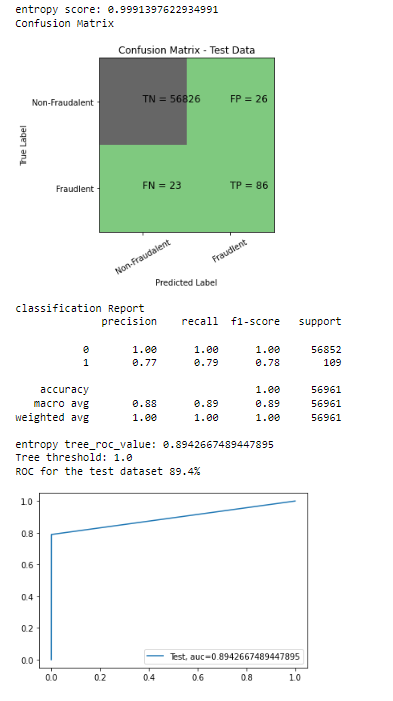
* Split the input dataset into K groups
* For each group:
  + Take one group as the reserve or test data set.
  + Use remaining groups as the training dataset
  + Fit the model on the training set and evaluate the performance of the model using the test set.

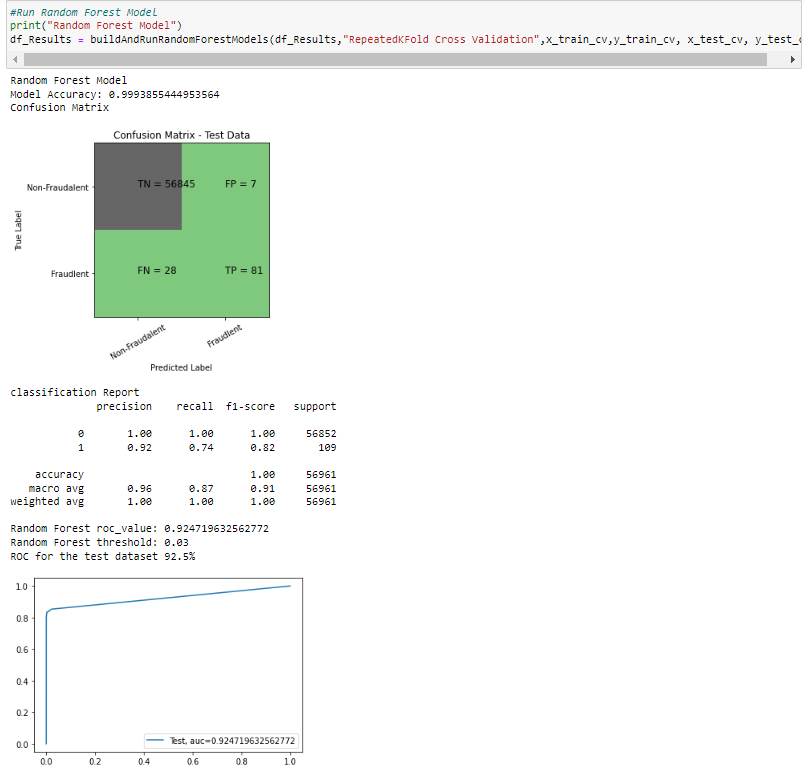


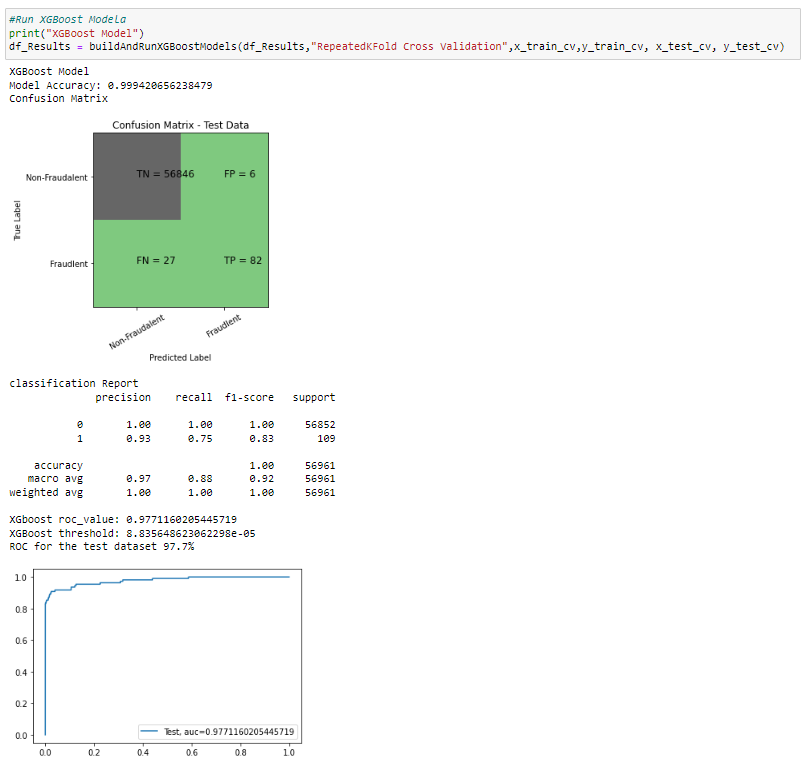


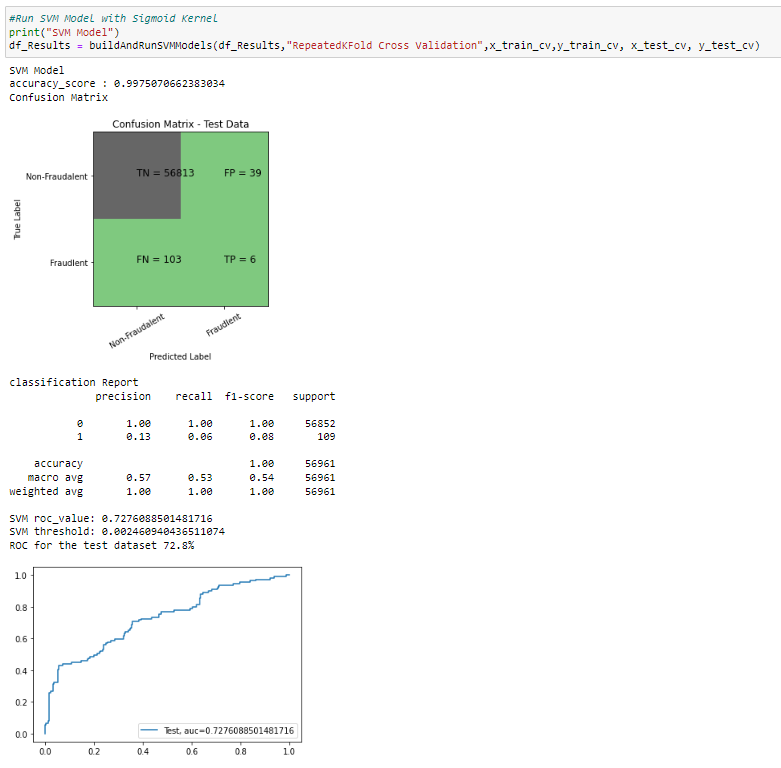


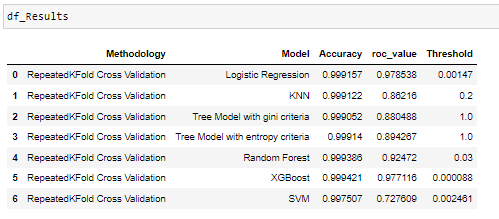


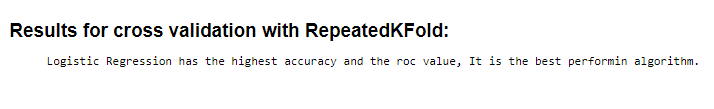








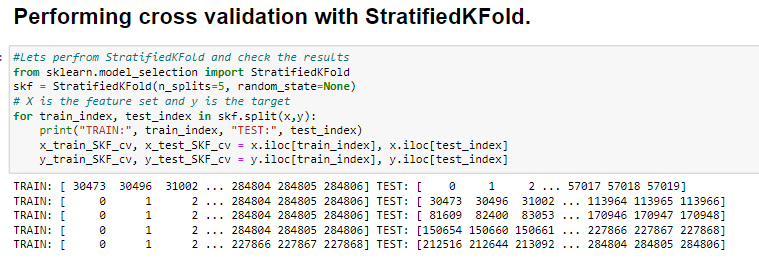


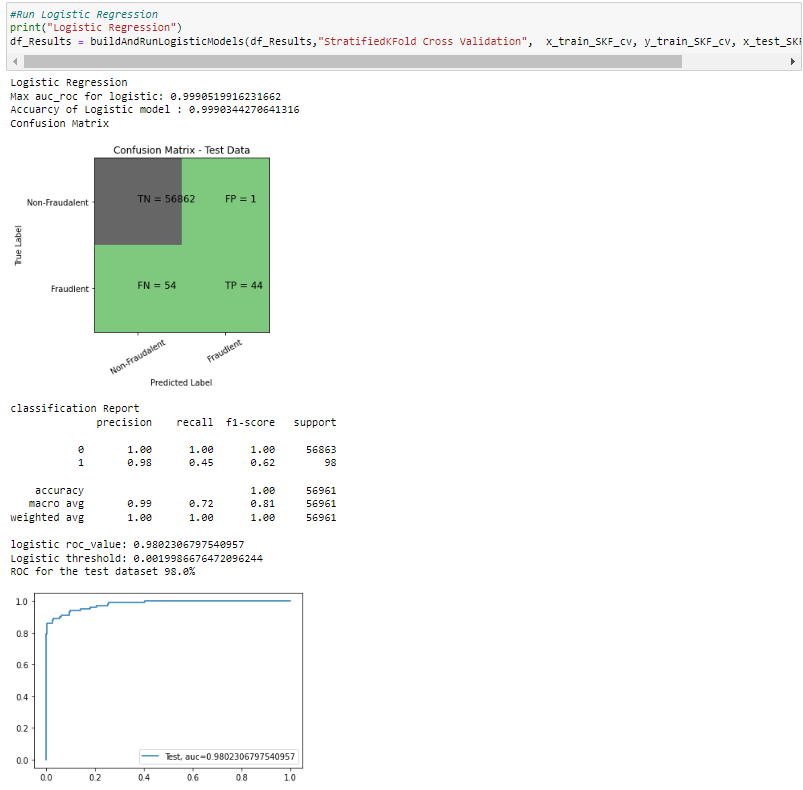


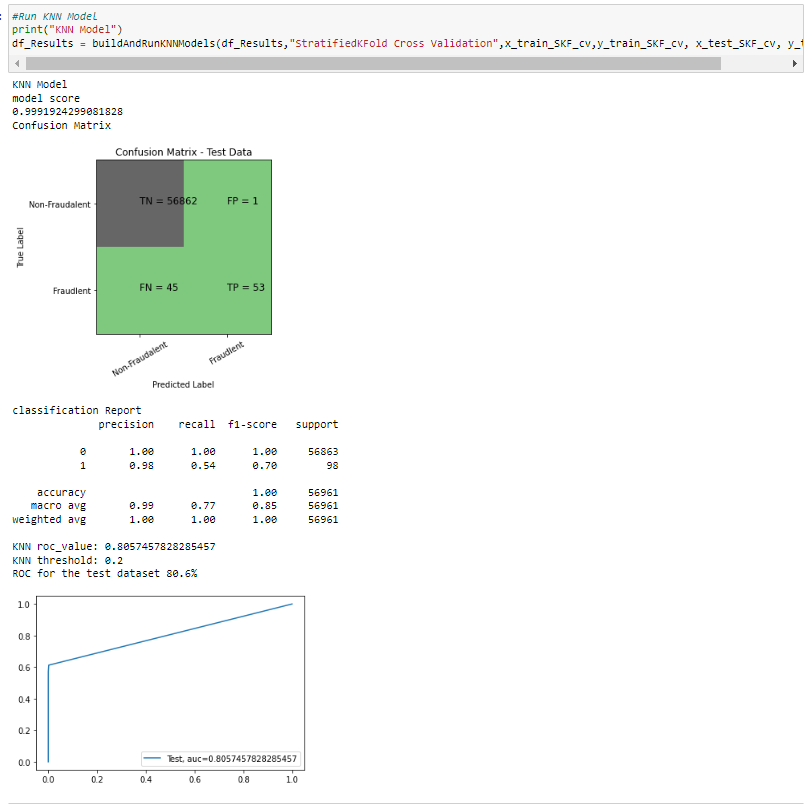
* **STARTIFIED K-FOLD CROSS-VALIDATION**

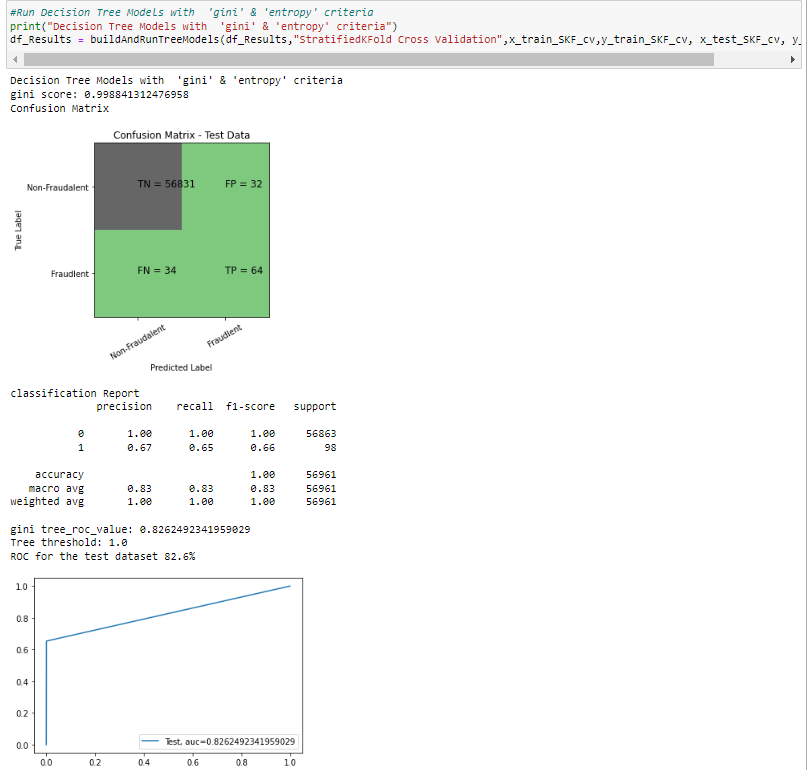
This technique is similar to k-fold cross-validation with some little changes. This approach works on stratification concept, it is a process of rearranging the data to ensure that each fold or group is a good representative of the complete dataset. To deal with the bias and variance, it is one of the best approaches.

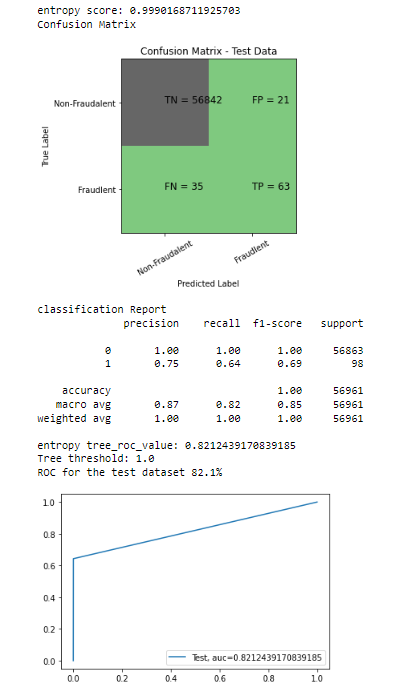
It can be understood with an example of housing prices, such that the price of some houses can be much high than other houses. To tackle such situations, a stratified k-fold cross-validation technique is useful.

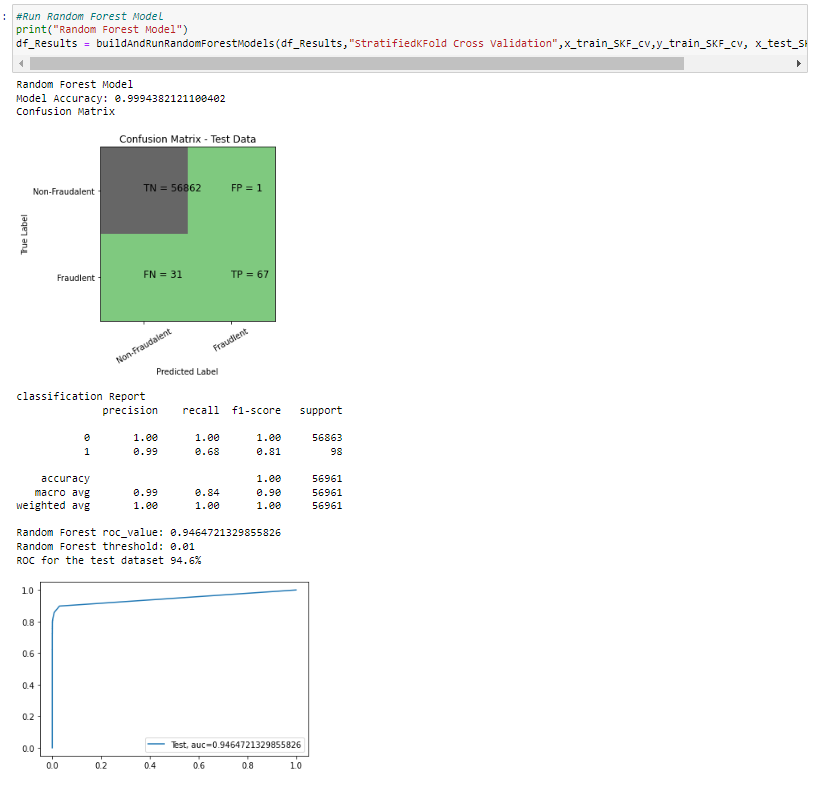


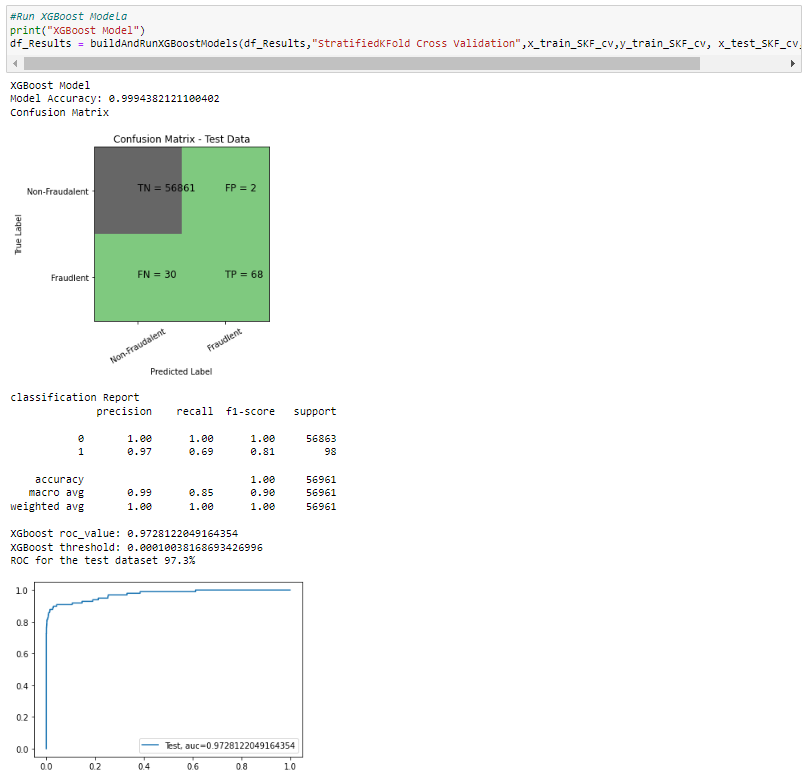


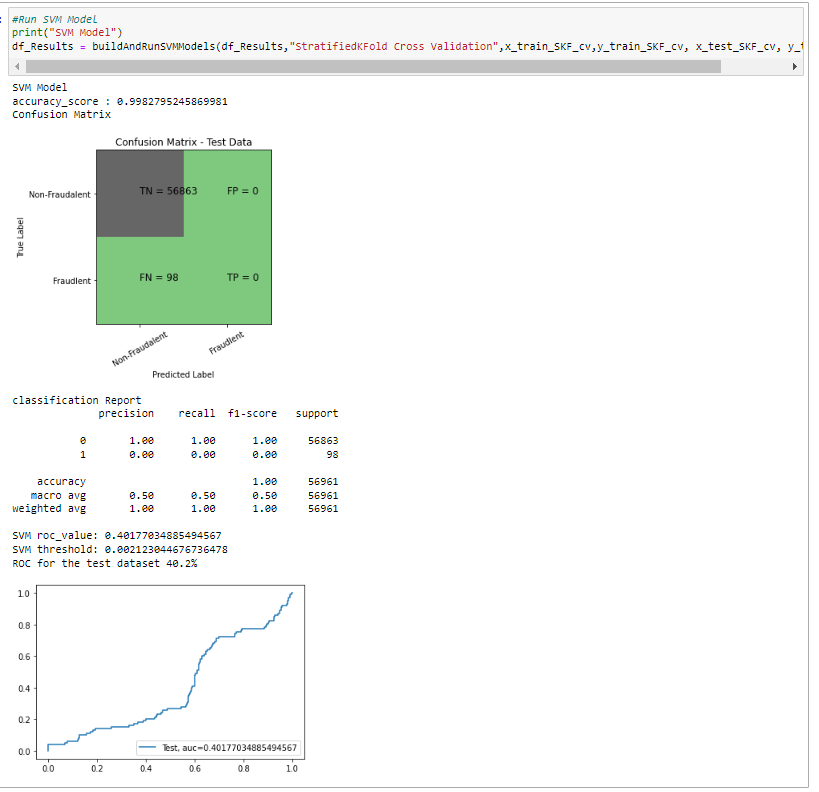


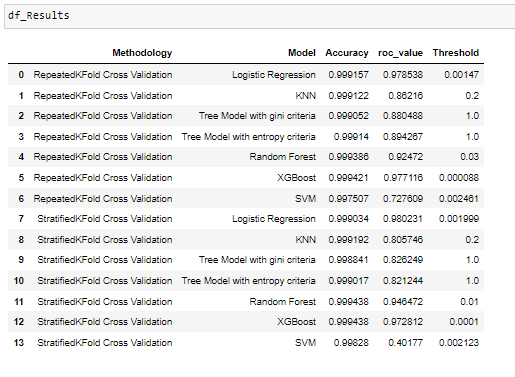


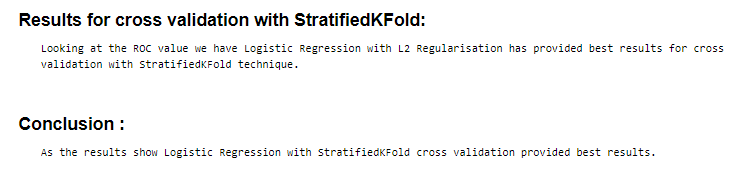












# OBSERVATIONS

1. So, this project is used for predicting the fraud related to the credit cards.
2. In this project we have used various machine learning algorithms such as KNN, Random Forest, Decision Tree and Logistic Regression.
3. But the best one algorithm whose accuracy is highest is: - Logistic Regression
4. Logistic Regression performed the best among all the used models.
5. Logistic Regression had the highest accuracy of 0.99034 and roc\_value of 0.980231.
6. We can also improve on this accuracy by increasing the sample size or use deep learning algorithms however at the cost of computational expense.We can also use complex anomaly detection models to get better accuracy in determining more fraudulent cases**.**

# REFERENCE

1. <https://scikit-learn.org/>
2. <https://seaborn.pydata.org/>
3. <https://matplotlib.org/>
4. <https://pandas.pydata.org/>
5. https://numpy.org/
6. [https://cs.nju.edu.cn/zhouzh/zhouzh.files/publication/icdm08b.pdf? q=isolation-forest](https://cs.nju.edu.cn/zhouzh/zhouzh.files/publication/icdm08b.pdf?q=isolation-forest)
7. https://www.researchgate.net/publication/224384174\_Isolation\_Forest
8. [https://www.datatechnotes.com/2020/04/anomaly-detection-with-one-classsvm.html](https://www.datatechnotes.com/2020/04/anomaly-detection-with-one-class-svm.html)
9. http://cucis.ece.northwestern.edu/projects/DMS/publications/ AnomalyDetection.pdf
10. https://indico.cern.ch/event/871578/contributions/3676247/attachments/ 1962862/3263071/Introduction\_IF\_ML.pdf
11. [https://machinelearningmastery.com/roc-curves-and-precision-recall-curvesf o r - c l a s s i f i c a t i o n - i n - p y t h o n](https://machinelearningmastery.com/roc-curves-and-precision-recall-curves-for-classification-in-python/#:~:text=A%20useful) [/](https://machinelearningmastery.com/roc-curves-and-precision-recall-curves-for-classification-in-python/#:~:text=A%20useful)

[#:~:text=A%20useful](https://machinelearningmastery.com/roc-curves-and-precision-recall-curves-for-classification-in-python/#:~:text=A%20useful)%20tool%20when%20predicting,values%20between%2 00.0%20and%201.0.

1. https://www.geeksforgeeks.org/introduction-to-support-vector-machines-svm/