Configuration Manual

MSc Research Project
Programme Name

Jagadeesh Komari
Student ID: 22150498

School of Computing
National College of Ireland

Supervisor: Shubham Subhnil

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Jagadeesh Komari		
Student ID:	22150498		
Programme:	Data Analytics	**Year:**	2023
Module:	MSc Research project		
Lecturer:	Shubham Subhnil		
Submission Due Date:	14/12/2023		
Project Title:	Exploration of Advanced Machine Learning Algorithms for Enhanced Fraud Detection in Financial Transactions		

Word Count: 412 **Page Count: 10**

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: K,Jagadeesh

Date: 14/12/2023

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

Jagadeesh Komari
Student ID: 22150498

1. Introduction

This manual demonstration about the well derived instructions for setting up and running the code for the process of research which i am working on Exploration of advanced machine learning algorithms for enhanced fraud detection in financial transactions. The project is implemented using the Python and utilizing the Google Colab for its powerful and collaborative computing environment in it. The next sections leads thorough required configurations and tools.

2. System Specification

The Financial transactions fraud detection system was developed and executed through the Google Colab, which is the cloud-based notebook environment. Where the System specifications are not applicable where Colab is an cloud based online service.

3. Software's Used:

The below mentioned following tools are necessary for code execution and running the financial fraud detection system for transactions:

- Google Account (To access the Google Colab)
- Google Colab (https://colab.research.google.com/)
- Python
- Numpy
- Matplotlib
- Seaborn
- Sklearn

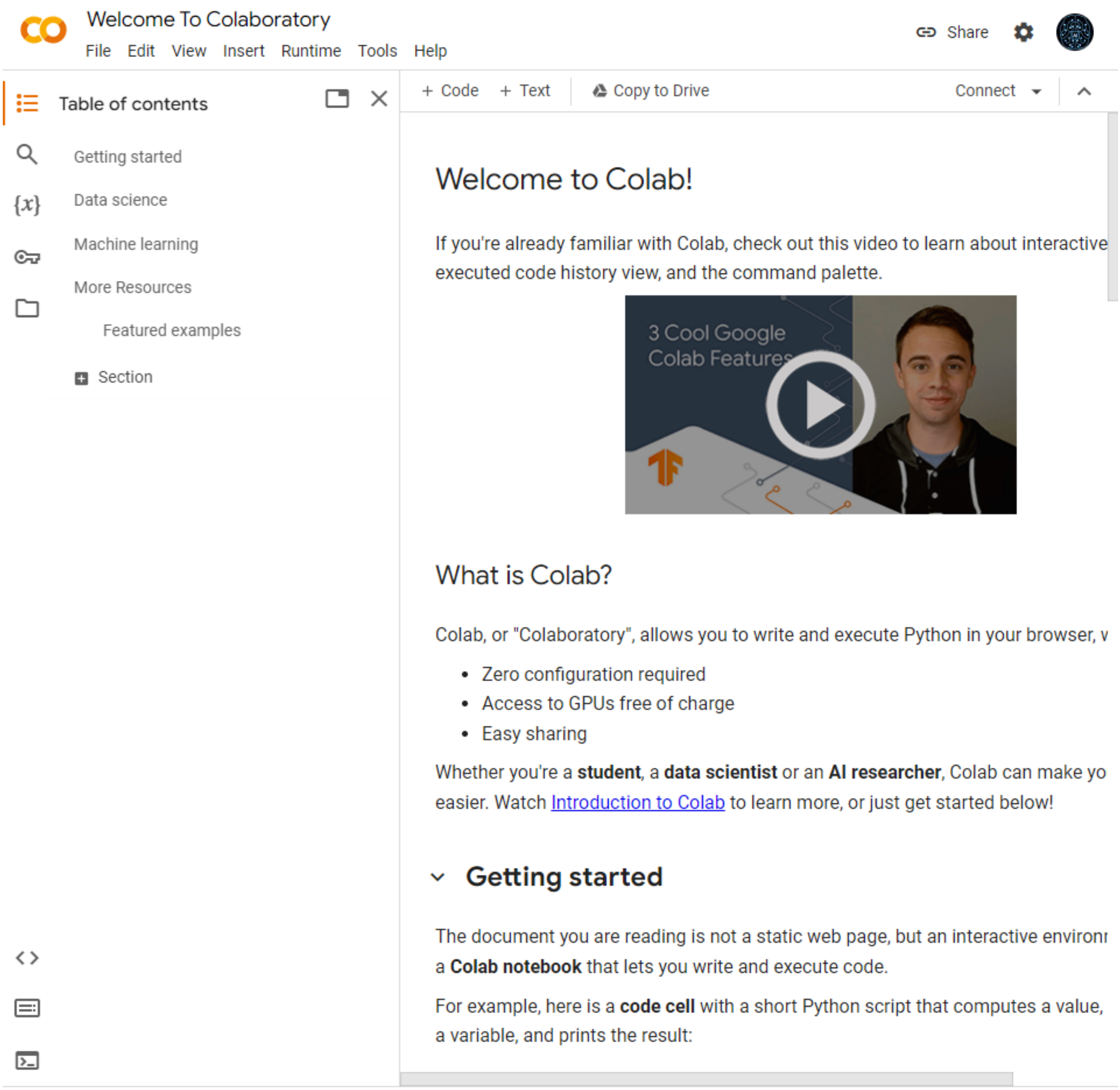
4. Setup the Software:

- First Ensures that have a google account or not.

- Then access to the Google Colab using this website: https://colab.research.google.com/

- Creates a new Colab notebook or upload the already written implementation notebook.

- ❖ Install the required packages or libraries which is necessary for the machine learning algorithms.



Welcome to Colab!

If you're already familiar with Colab, check out this video to learn about interactive executed code history view, and the command palette.

What is Colab?

Colab, or "Colaboratory", allows you to write and execute Python in your browser, v

- Zero configuration required
- Access to GPUs free of charge
- Easy sharing

Whether you're a **student**, a **data scientist** or an **AI researcher**, Colab can make yo easier. Watch Introduction to Colab to learn more, or just get started below!

Getting started

The document you are reading is not a static web page, but an interactive environı a **Colab notebook** that lets you write and execute code.

For example, here is a **code cell** with a short Python script that computes a value, a variable, and prints the result:

5. Setting up the Environment:

- ❖ For Google Colab, there is no need for any explicit environment setup we can access it from any web browser.

- ❖ Assures that you have a stable internet connection to access Colab notebooks and libraries to use them properly without any issues.

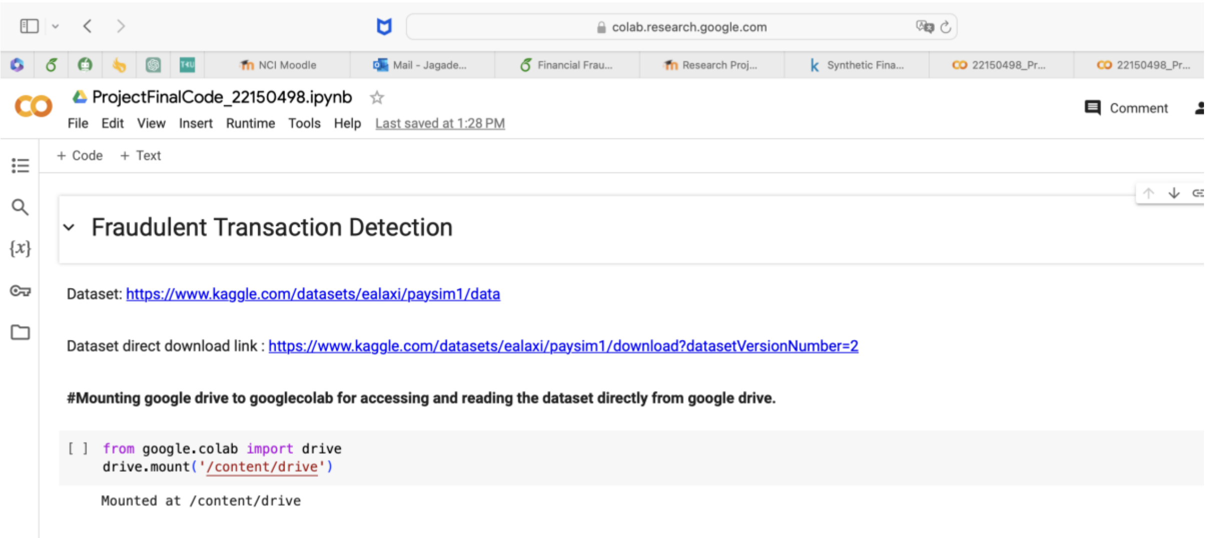
6. Source of Dataset

For Financial fraud detection I obtained the dataset form the Kaggle which was suitable for our research to topic of fraud detection.

7. Running of the Application (Financial Fraud Detection)

❖ **Open the Colab notebook through any web browser.**

Mount the Google drive to google drive and provide access to the drive to read and access the large dataset saved in the google drive.



Fraudulent Transaction Detection

Dataset: https://www.kaggle.com/datasets/ealaxi/paysim1/data

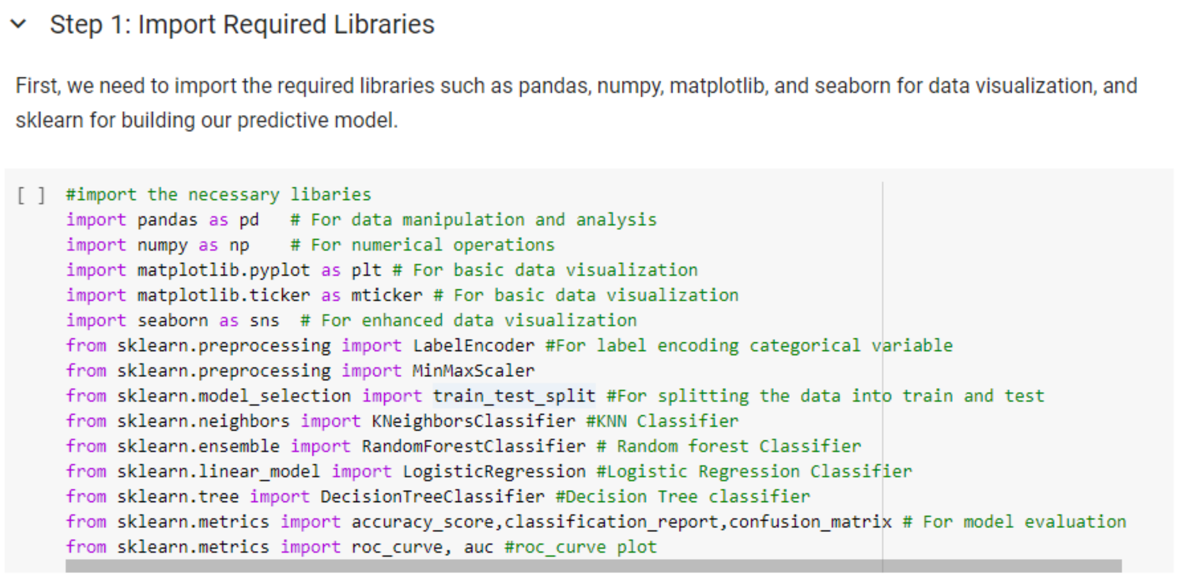
Dataset direct download link : https://www.kaggle.com/datasets/ealaxi/paysim1/download?datasetVersionNumber=2

#Mounting google drive to googlecolab for accessing and reading the dataset directly from google drive.

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

❖ **Import the Required libaries**

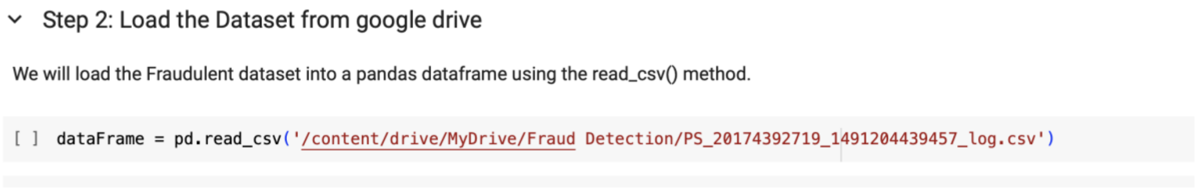


Step 1: Import Required Libraries

First, we need to import the required libraries such as pandas, numpy, matplotlib, and seaborn for data visualization, and sklearn for building our predictive model.

```
#import the necessary libaries
import pandas as pd   # For data manipulation and analysis
import numpy as np    # For numerical operations
import matplotlib.pyplot as plt # For basic data visualization
import matplotlib.ticker as mticker # For basic data visualization
import seaborn as sns  # For enhanced data visualization
from sklearn.preprocessing import LabelEncoder #For label encoding categorical variable
from sklearn.preprocessing import MinMaxScaler
from sklearn.model_selection import train_test_split #For splitting the data into train and test
from sklearn.neighbors import KNeighborsClassifier #KNN Classifier
from sklearn.ensemble import RandomForestClassifier # Random forest Classifier
from sklearn.linear_model import LogisticRegression #Logistic Regression Classifier
from sklearn.tree import DecisionTreeClassifier #Decision Tree classifier
from sklearn.metrics import accuracy_score,classification_report,confusion_matrix # For model evaluation
from sklearn.metrics import roc_curve, auc #roc_curve plot
```

❖ Load the Financial Fraud Dataset

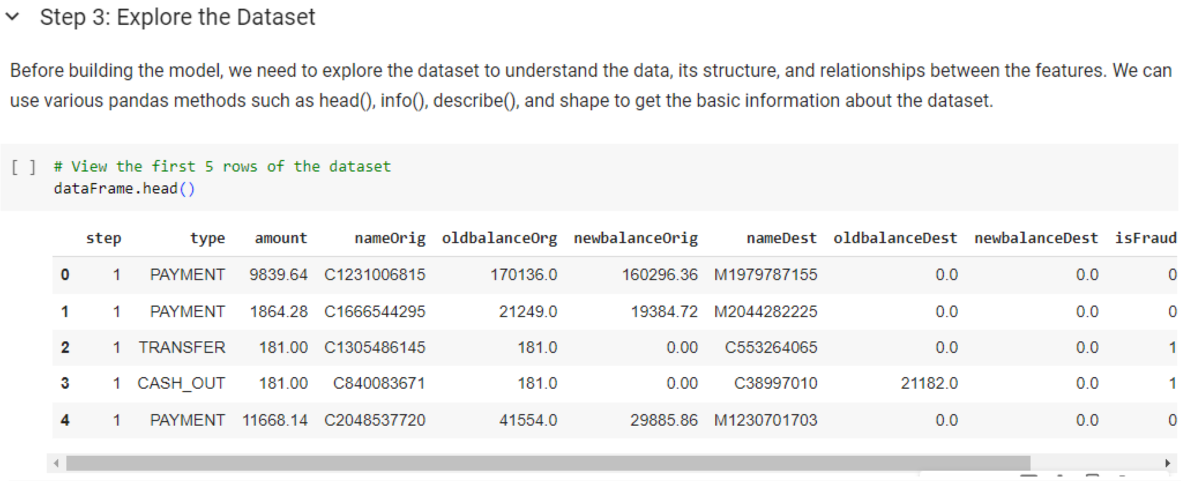


Step 2: Load the Dataset from google drive

We will load the Fraudulent dataset into a pandas dataframe using the read_csv() method.

```
[ ]  dataFrame = pd.read_csv('/content/drive/MyDrive/Fraud Detection/PS_20174392719_1491204439457_log.csv')
```

❖ Explore the Financial Fraud Dataset

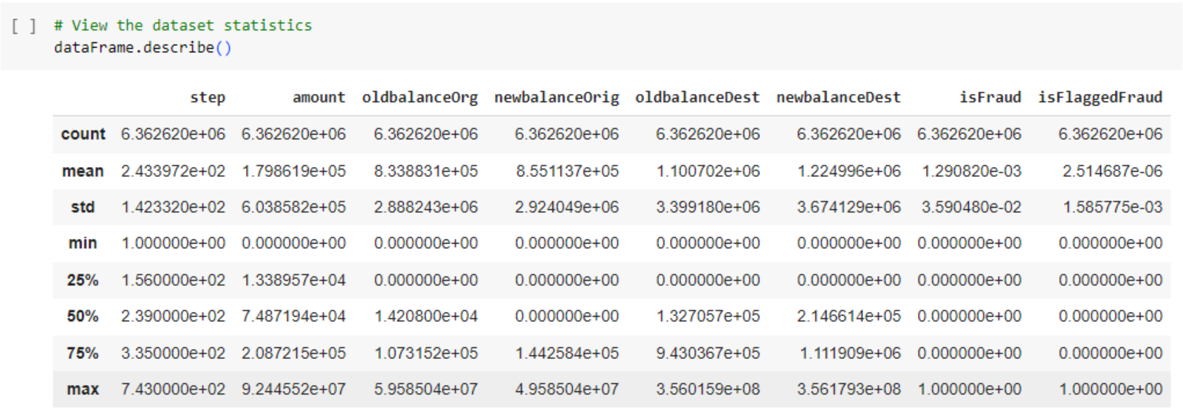


Step 3: Explore the Dataset

Before building the model, we need to explore the dataset to understand the data, its structure, and relationships between the features. We can use various pandas methods such as head(), info(), describe(), and shape to get the basic information about the dataset.

```
[ ]  # View the first 5 rows of the dataset
     dataFrame.head()
```

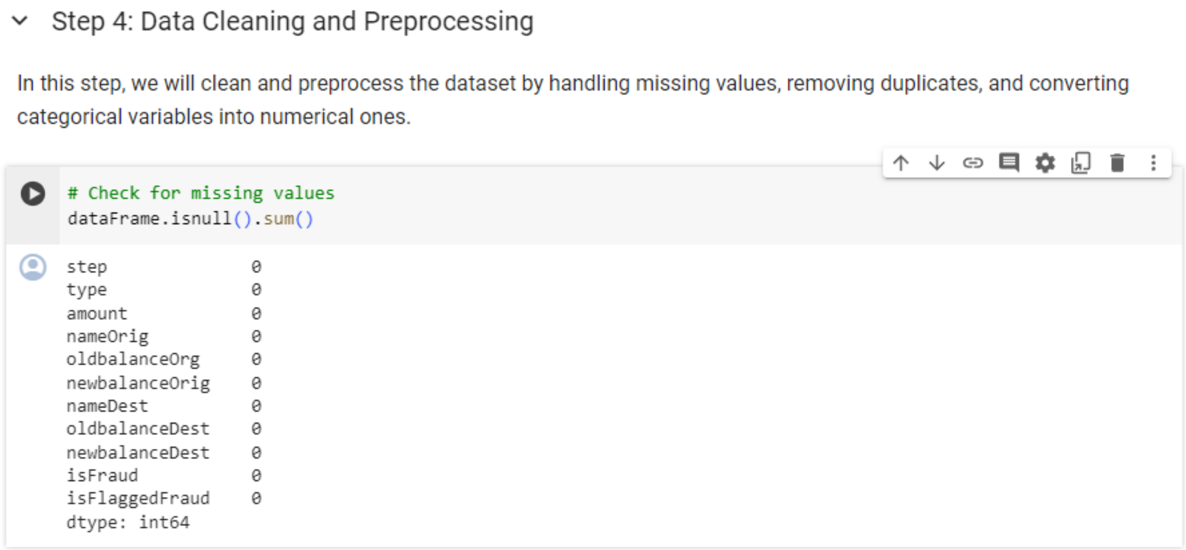
	step	type	amount	nameOrig	oldbalanceOrg	newbalanceOrig	nameDest	oldbalanceDest	newbalanceDest	isFraud
0	1	PAYMENT	9839.64	C1231006815	170136.0	160296.36	M1979787155	0.0	0.0	0
1	1	PAYMENT	1864.28	C1666544295	21249.0	19384.72	M2044282225	0.0	0.0	0
2	1	TRANSFER	181.00	C1305486145	181.0	0.00	C553264065	0.0	0.0	1
3	1	CASH_OUT	181.00	C840083671	181.0	0.00	C38997010	21182.0	0.0	1
4	1	PAYMENT	11668.14	C2048537720	41554.0	29885.86	M1230701703	0.0	0.0	0



```
[ ]  # View the dataset statistics
     dataFrame.describe()
```

	step	amount	oldbalanceOrg	newbalanceOrig	oldbalanceDest	newbalanceDest	isFraud	isFlaggedFraud
count	6.362620e+06	6.362620e+06	6.362620e+06	6.362620e+06	6.362620e+06	6.362620e+06	6.362620e+06	6.362620e+06
mean	2.433972e+02	1.798619e+05	8.338831e+05	8.551137e+05	1.100702e+06	1.224996e+06	1.290820e-03	2.514687e-06
std	1.423320e+02	6.038582e+05	2.888243e+06	2.924049e+06	3.399180e+06	3.674129e+06	3.590480e-02	1.585775e-03
min	1.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00
25%	1.560000e+02	1.338957e+04	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00
50%	2.390000e+02	7.487194e+04	1.420800e+04	0.000000e+00	1.327057e+05	2.146614e+05	0.000000e+00	0.000000e+00
75%	3.350000e+02	2.087215e+05	1.073152e+05	1.442584e+05	9.430367e+05	1.111909e+06	0.000000e+00	0.000000e+00
max	7.430000e+02	9.244552e+07	5.958504e+07	4.958504e+07	3.560159e+08	3.561793e+08	1.000000e+00	1.000000e+00

❖ Data Cleaning and Preprocessing of the Financial Fraud Dataset



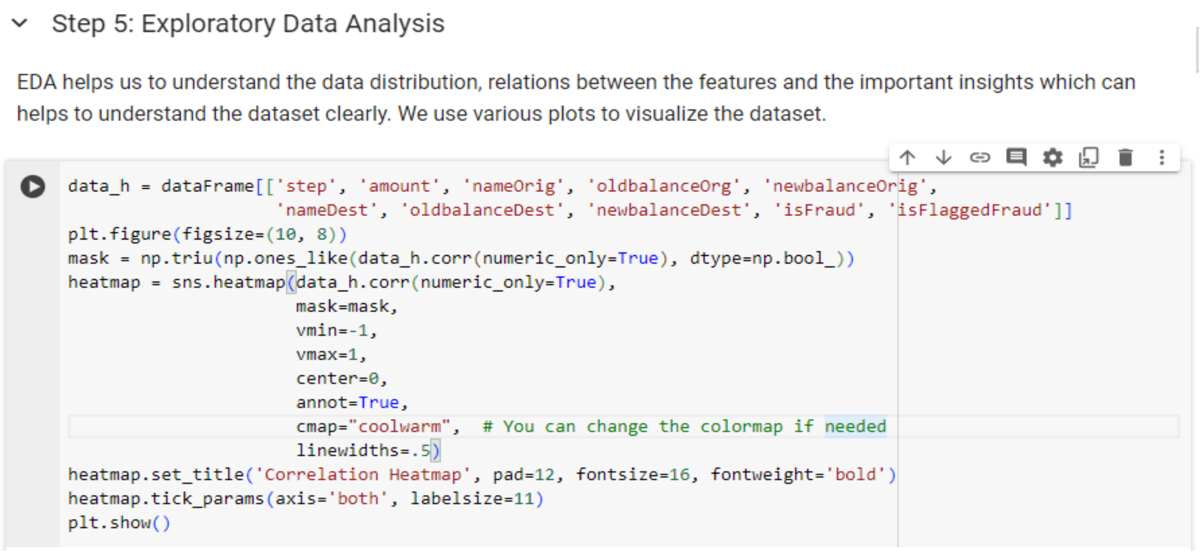
Step 4: Data Cleaning and Preprocessing

In this step, we will clean and preprocess the dataset by handling missing values, removing duplicates, and converting categorical variables into numerical ones.

```
# Check for missing values
dataFrame.isnull().sum()
```

```
step              0
type              0
amount            0
nameOrig          0
oldbalanceOrg     0
newbalanceOrig    0
nameDest          0
oldbalanceDest    0
newbalanceDest    0
isFraud           0
isFlaggedFraud    0
dtype: int64
```

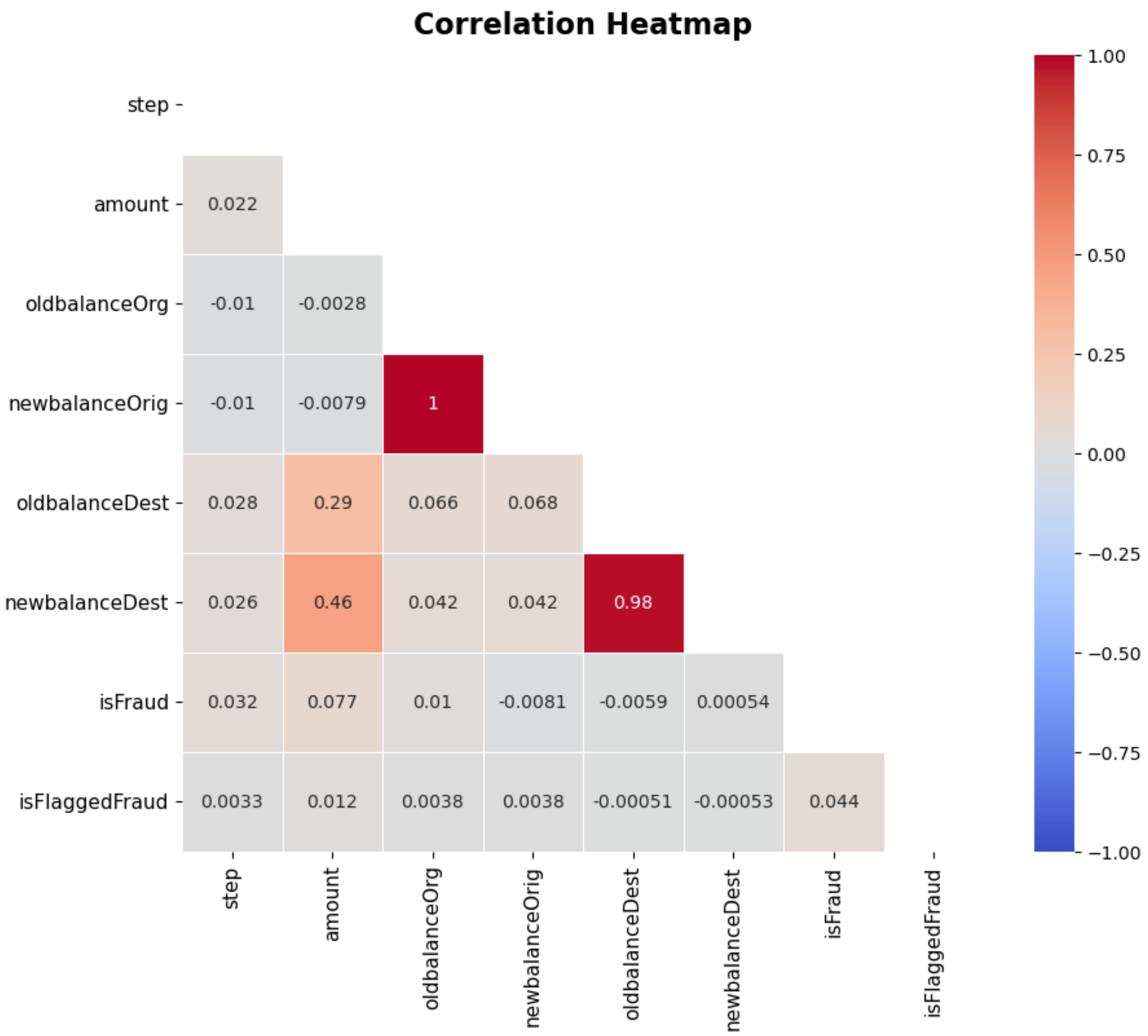
❖ EDA of the Financial Fraud Dataset

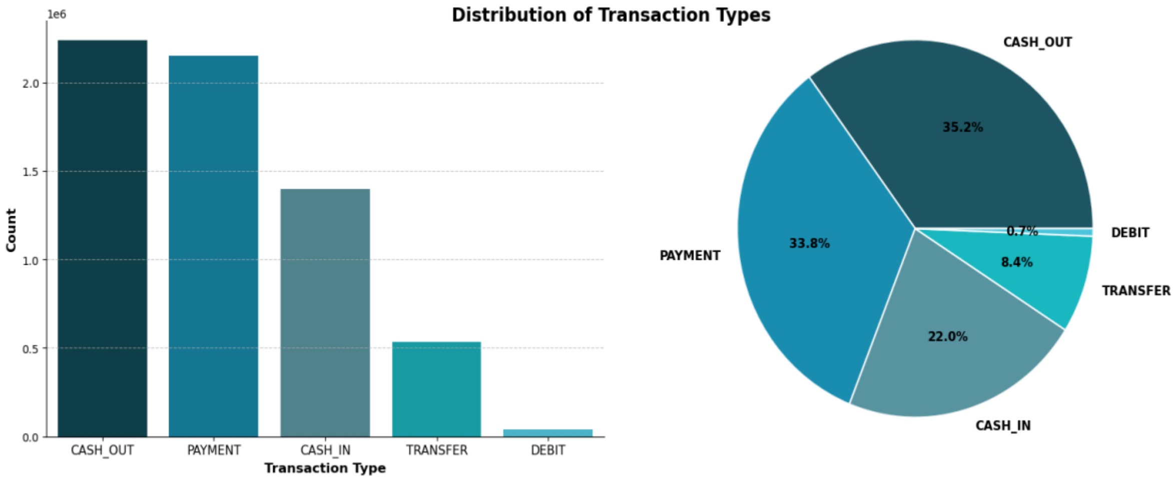


Step 5: Exploratory Data Analysis

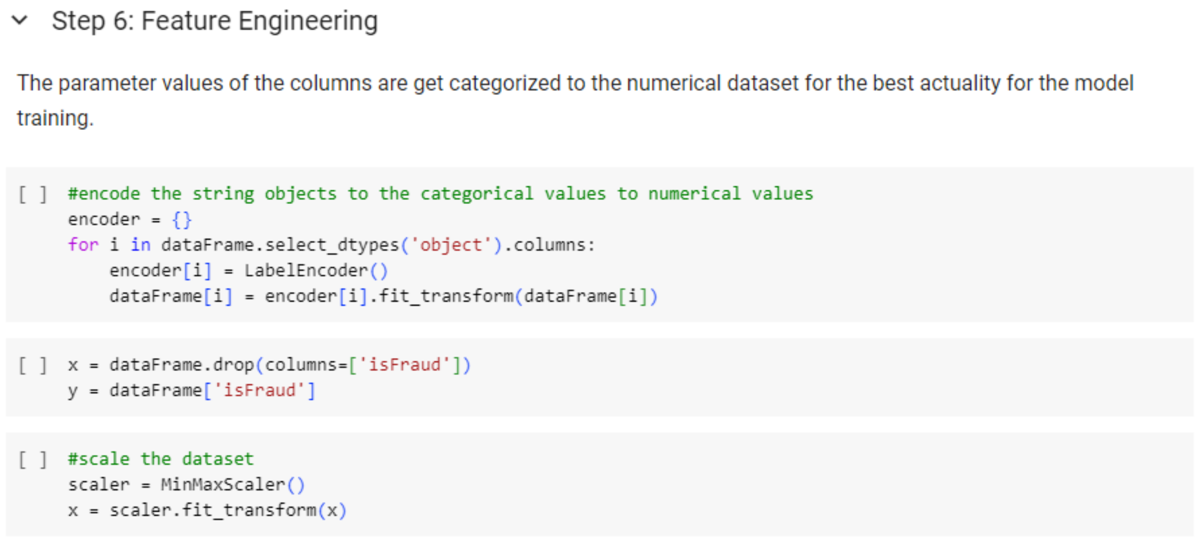
EDA helps us to understand the data distribution, relations between the features and the important insights which can helps to understand the dataset clearly. We use various plots to visualize the dataset.

```
data_h = dataFrame[['step', 'amount', 'nameOrig', 'oldbalanceOrg', 'newbalanceOrig',
                    'nameDest', 'oldbalanceDest', 'newbalanceDest', 'isFraud', 'isFlaggedFraud']]
plt.figure(figsize=(10, 8))
mask = np.triu(np.ones_like(data_h.corr(numeric_only=True), dtype=np.bool_))
heatmap = sns.heatmap(data_h.corr(numeric_only=True),
                      mask=mask,
                      vmin=-1,
                      vmax=1,
                      center=0,
                      annot=True,
                      cmap="coolwarm",  # You can change the colormap if needed
                      linewidths=.5)
heatmap.set_title('Correlation Heatmap', pad=12, fontsize=16, fontweight='bold')
heatmap.tick_params(axis='both', labelsize=11)
plt.show()
```


Correlation Heatmap
step
amount
oldbalanceOrg
newbalanceOrig
oldbalanceDest
newbalanceDest
isFraud
isFlaggedFraud
0.022
-0.01
-0.0028
-0.01
-0.0079
1
0.028
0.29
0.066
0.068
0.026
0.46
0.042
0.042
0.98
0.032
0.077
0.01
-0.0081
-0.0059
0.00054
0.0033
0.012
0.0038
0.0038
-0.00051
-0.00053
0.044
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00


Distribution of Transaction Types
1e6
2.0
1.5
1.0
0.5
0.0
Count
CASH_OUT
PAYMENT
CASH_IN
TRANSFER
DEBIT
Transaction Type
CASH_OUT
35.2%
DEBIT
0.7%
8.4%
TRANSFER
22.0%
CASH_IN
33.8%
PAYMENT

- ## Feature engineering to select the appropriate features for the Financial Fraud detection from the dataframe



Step 6: Feature Engineering

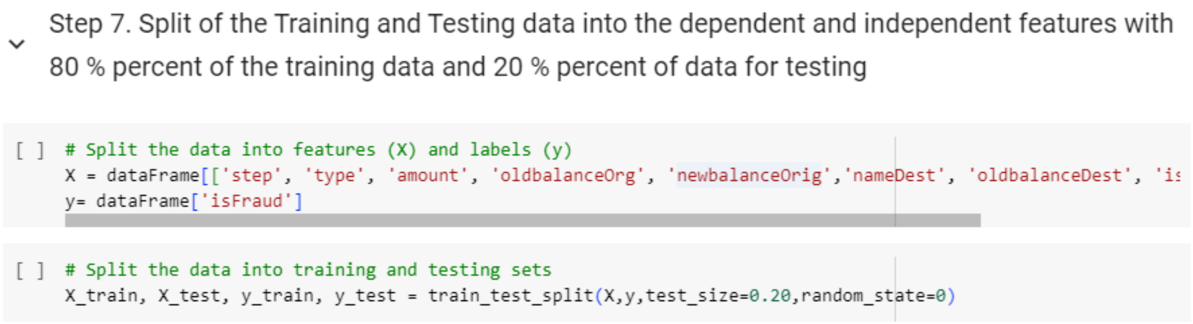
The parameter values of the columns are get categorized to the numerical dataset for the best actuality for the model training.

```
[ ] #encode the string objects to the categorical values to numerical values
    encoder = {}
    for i in dataFrame.select_dtypes('object').columns:
        encoder[i] = LabelEncoder()
        dataFrame[i] = encoder[i].fit_transform(dataFrame[i])
```

```
[ ] x = dataFrame.drop(columns=['isFraud'])
    y = dataFrame['isFraud']
```

```
[ ] #scale the dataset
    scaler = MinMaxScaler()
    x = scaler.fit_transform(x)
```

- ## Splitting of the data into the training and testing

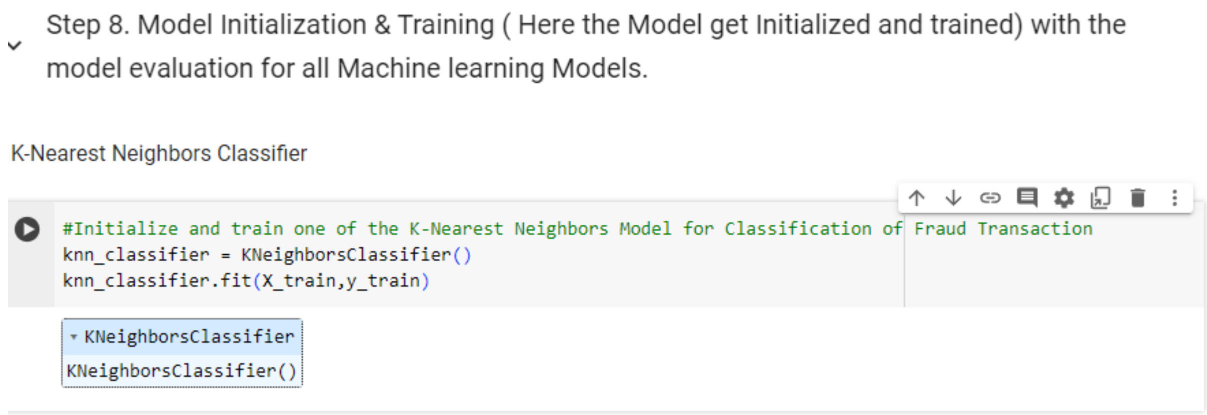


Step 7. Split of the Training and Testing data into the dependent and independent features with 80 % percent of the training data and 20 % percent of data for testing

```
[ ] # Split the data into features (X) and labels (y)
    X = dataFrame[['step', 'type', 'amount', 'oldbalanceOrg', 'newbalanceOrig','nameDest', 'oldbalanceDest', 'is
    y= dataFrame['isFraud']
```

```
[ ] # Split the data into training and testing sets
    X_train, X_test, y_train, y_test = train_test_split(X,y,test_size=0.20,random_state=0)
```

- ## Model Initialization and Training



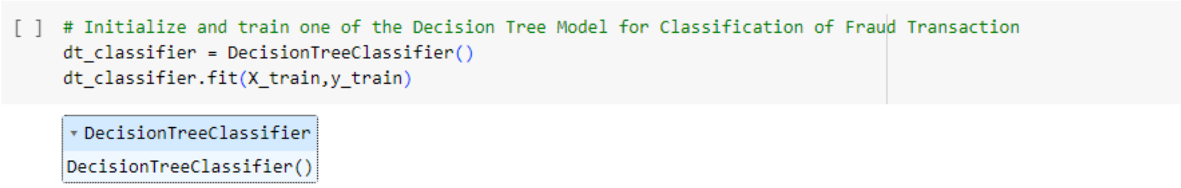
Step 8. Model Initialization & Training (Here the Model get Initialized and trained) with the model evaluation for all Machine learning Models.

K-Nearest Neighbors Classifier

```
#Initialize and train one of the K-Nearest Neighbors Model for Classification of Fraud Transaction
knn_classifier = KNeighborsClassifier()
knn_classifier.fit(X_train,y_train)
```

```
▾ KNeighborsClassifier
KNeighborsClassifier()
```

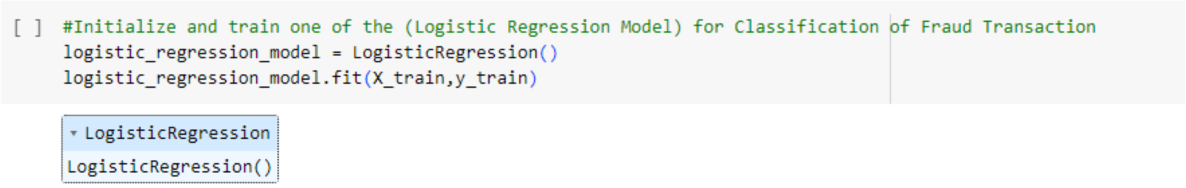
Decision Tree Classifier



```
[ ] # Initialize and train one of the Decision Tree Model for Classification of Fraud Transaction
    dt_classifier = DecisionTreeClassifier()
    dt_classifier.fit(X_train,y_train)
```

```
▾ DecisionTreeClassifier
DecisionTreeClassifier()
```

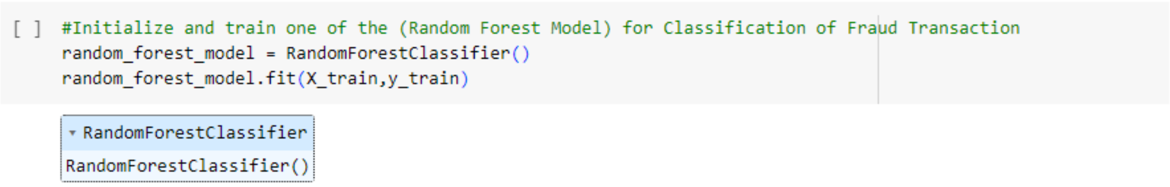
Logistic Regression Classifier



```
[ ] #Initialize and train one of the (Logistic Regression Model) for Classification of Fraud Transaction
    logistic_regression_model = LogisticRegression()
    logistic_regression_model.fit(X_train,y_train)
```

```
▾ LogisticRegression
LogisticRegression()
```

Random Forest Classifier

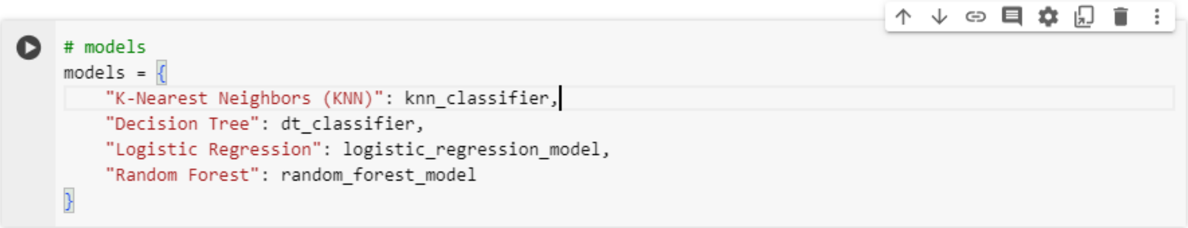


```
[ ] #Initialize and train one of the (Random Forest Model) for Classification of Fraud Transaction
    random_forest_model = RandomForestClassifier()
    random_forest_model.fit(X_train,y_train)
```

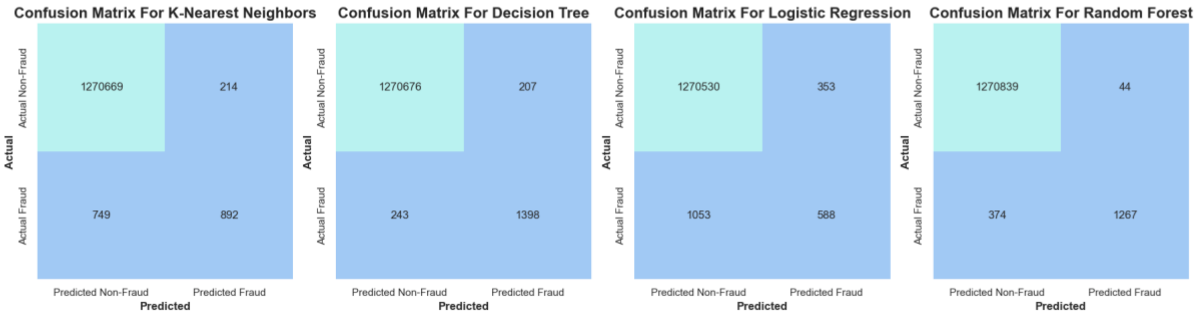
```
▾ RandomForestClassifier
RandomForestClassifier()
```

❖ Model Evaluation

Step 9. Comparision of Metrics Results for all Machine learning Models.



```
# models
models = {
    "K-Nearest Neighbors (KNN)": knn_classifier,
    "Decision Tree": dt_classifier,
    "Logistic Regression": logistic_regression_model,
    "Random Forest": random_forest_model
}
```



Confusion Matrix For K-Nearest Neighbors

Actual \ Predicted	Predicted Non-Fraud	Predicted Fraud
Actual Non-Fraud	1270669	214
Actual Fraud	749	892

Confusion Matrix For Decision Tree

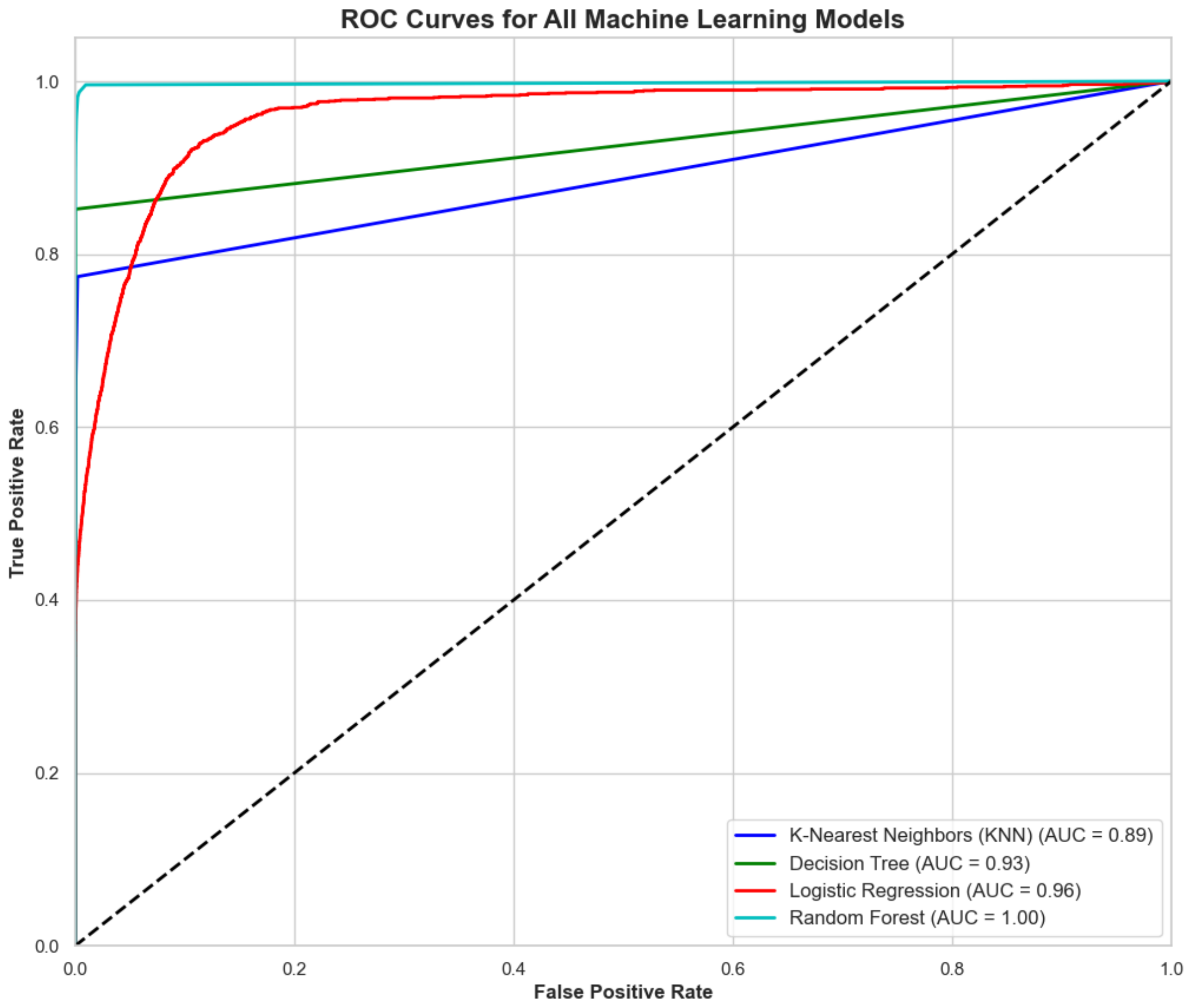
Actual \ Predicted	Predicted Non-Fraud	Predicted Fraud
Actual Non-Fraud	1270676	207
Actual Fraud	243	1398

Confusion Matrix For Logistic Regression

Actual \ Predicted	Predicted Non-Fraud	Predicted Fraud
Actual Non-Fraud	1270530	353
Actual Fraud	1053	588

Confusion Matrix For Random Forest

Actual \ Predicted	Predicted Non-Fraud	Predicted Fraud
Actual Non-Fraud	1270839	44
Actual Fraud	374	1267



This manual describes as a detailed manual for configuration, code running, for the exploration of advanced machine learning algorithms for the application enhanced fraud detection in financial transactions using the Google Colab.

References

Google Account: https:/gmail.com/
Google Colab: https://colab.research.google.com/
Kaggle: https://www.kaggle.com/