**FRAUD DETECTION IN FINANCIAL TRANSACTION**

**TASK -2**

NAME: Muthu Karthi

Domain : Data Science Intern

Mail id : [muthukarthim373@gmail.com](mailto:muthukarthim373@gmail.com)

**Project Description :**

**Develop a machine learning model to detect fraudulent transactions in a financial dataset.**

**Steps:**

1. **Data Collection: Obtain historical transaction data, including features like transaction amount, timestamp, etc.**
2. **Data Preprocessing: Clean the data, handle missing values, and balance the dataset if needed.**
3. **Feature Engineering: Extract relevant features and engineer new ones that could aid in fraud detection.**
4. **Exploratory Data Analysis (EDA): Visualize patterns and anomalies in the data using graphs and statistics.**
5. **Model Selection: Choose classification algorithms like Random Forest, Support Vector Machines, or Neural Networks.**
6. **Model Training: Train the model on the preprocessed dataset.**
7. **Model Evaluation: Evaluate the model's performance using metrics like accuracy, precision, recall, and F1-score.**

**Tech Stack:**

* **Python**
* **Data manipulation libraries**
* **Machine learning libraries**
* **Deep learning libraries**

**Abstract**

With the exponential growth of digital financial transactions, the risk of fraudulent activities has become a significant concern for financial institutions and customers alike. Traditional rule-based fraud detection systems often struggle to keep pace with the evolving tactics of fraudsters. In response, this project proposes an advanced fraud detection system leveraging machine learning techniques.

The primary objective of this project is to design and implement a robust fraud detection system capable of effectively identifying and preventing fraudulent activities in real-time financial transactions. The proposed system will utilize a combination of supervised and unsupervised machine learning algorithms to analyze transactional data and detect anomalous patterns indicative of fraudulent behavior.

Key components of the proposed system include data preprocessing, feature engineering, model selection, and evaluation. Data preprocessing involves cleaning and transforming raw transactional data into a format suitable for analysis. Feature engineering aims to extract relevant features from the data that can help distinguish between legitimate and fraudulent transactions. Model selection involves choosing the most appropriate machine learning algorithms for the task, considering factors such as accuracy, scalability, and interpretability. Evaluation will be performed using metrics such as precision, recall, and F1-score to assess the effectiveness of the system in detecting fraud while minimizing false positives.

The project will be implemented using Python programming language and popular machine learning libraries such as scikit-learn and TensorFlow. The system will be trained and tested using a large dataset of historical financial transactions, encompassing various transaction types and scenarios.

The expected outcome of this project is a highly accurate and efficient fraud detection system capable of identifying fraudulent transactions in real-time, thereby minimizing financial losses and enhancing trust and security in digital transactions. The proposed system has the potential to benefit financial institutions, merchants, and consumers by reducing the incidence of fraud and improving overall transaction security.

**DATA COLLECTION**

**Define Your Objectives:** Clearly outline what types of fraud you want to detect and what constitutes normal transactions. This will guide your data collection efforts.

**Identify Data Sources: Determine where you can obtain relevant data. Potential sources include:**

* + Transaction logs: These contain details about each transaction, such as timestamp, amount, involved parties, and transaction type.
  + Customer information: Gather data on customers involved in transactions, including demographics and historical behavior.
  + Merchant data: Information about the entities receiving payments, including business type, location, and transaction history.
  + External data: Incorporate external sources like public records, social media, and industry reports to enrich your dataset.

**Data Collection Methodology:**

* + Internal sources: Work with your organization's IT or data team to access transaction logs and customer databases securely.
  + External sources: Utilize APIs, web scraping, or third-party data providers to gather additional information.
  + Ensure compliance: Adhere to data privacy regulations like GDPR or CCPA when collecting and handling customer data.

**Data Preprocessing:**

* + Clean the data: Remove duplicates, correct errors, and handle missing values.
  + Feature engineering: Create new features that might enhance fraud detection, such as transaction frequency, location-based features, or customer behavior patterns.
  + Normalize or scale numerical features: Ensure all features are on a similar scale to prevent model bias.

**Labeling Data:** Annotate your dataset to indicate which transactions are fraudulent and which are legitimate. This can be a time-consuming process but is essential for supervised learning**.**

**Data Splitting**: Divide your labeled dataset into training, validation, and test sets to evaluate model performance properly.

**Data Security**: Implement measures to protect sensitive information throughout the data collection, storage, and analysis process.

**Iterative Process:** Fraud patterns evolve over time, so continue to monitor and update your dataset regularly to keep your model effective.

**Importing the Dependencies**

import numpy as np

import pandas as pd

from sklearn.model\_selection import train\_test\_split

from sklearn.linear\_model import LogisticRegression

from sklearn.metrics import accuracy\_score

# loading the dataset to a Pandas DataFrame

credit\_card\_data = pd.read\_csv('/content/credit\_data.csv')

# first 5 rows of the dataset

credit\_card\_data.head()



credit\_card\_data.tail()



# dataset informations

credit\_card\_data.info()

class 'pandas.core.frame.DataFrame'>

RangeIndex: 284807 entries, 0 to 284806

Data columns (total 31 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 Time 284807 non-null float64

1 V1 284807 non-null float64

2 V2 284807 non-null float64

3 V3 284807 non-null float64

4 V4 284807 non-null float64

5 V5 284807 non-null float64

6 V6 284807 non-null float64

7 V7 284807 non-null float64

8 V8 284807 non-null float64

9 V9 284807 non-null float64

10 V10 284807 non-null float64

11 V11 284807 non-null float64

12 V12 284807 non-null float64

13 V13 284807 non-null float64

14 V14 284807 non-null float64

15 V15 284807 non-null float64

16 V16 284807 non-null float64

17 V17 284807 non-null float64

18 V18 284807 non-null float64

19 V19 284807 non-null float64

20 V20 284807 non-null float64

21 V21 284807 non-null float64

22 V22 284807 non-null float64

23 V23 284807 non-null float64

24 V24 284807 non-null float64

25 V25 284807 non-null float64

26 V26 284807 non-null float64

27 V27 284807 non-null float64

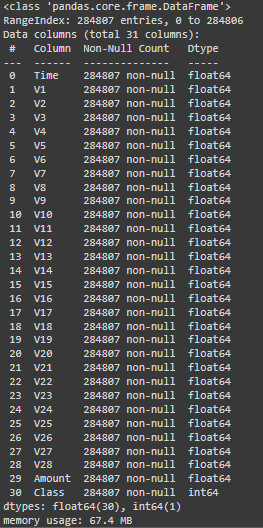
28 V28 284807 non-null float64

29 Amount 284807 non-null float64

30 Class 284807 non-null int64

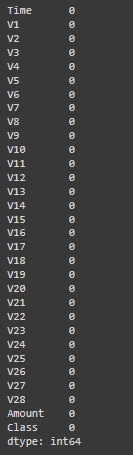
dtypes: float64(30), int64(1)

memory usage: 67.4 MB



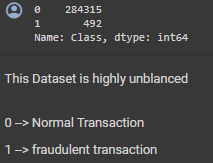
# checking the number of missing values in each column

credit\_card\_data.isnull().sum()



# distribution of legit transactions & fraudulent transactions

credit\_card\_data['Class'].value\_counts()



# separating the data for analysis

legit = credit\_card\_data[credit\_card\_data.Class == 0]

fraud = credit\_card\_data[credit\_card\_data.Class == 1]

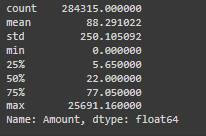
print(legit.shape)

print(fraud.shape)

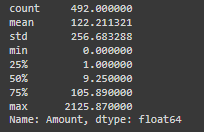


# statistical measures of the data

legit.Amount.describe()

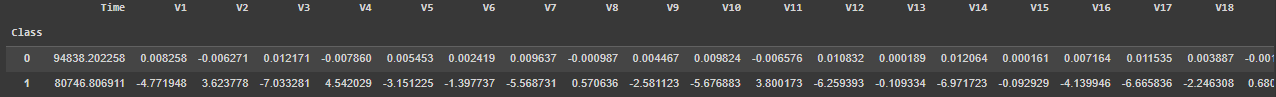


fraud.Amount.describe()



# compare the values for both transactions

credit\_card\_data.groupby('Class').mean()



**Under-Sampling**

**Build a sample dataset containing similar distribution of normal transactions and Fraudulent Transactions**

**Number of Fraudulent Transactions --> 492**

legit\_sample = legit.sample(n=492)

**Concatenating two Data Frames**

new\_dataset = pd.concat([legit\_sample, fraud], axis=0)

new\_dataset.head()



new\_dataset.tail()



new\_dataset['Class'].value\_counts()



new\_dataset.groupby('Class').mean()

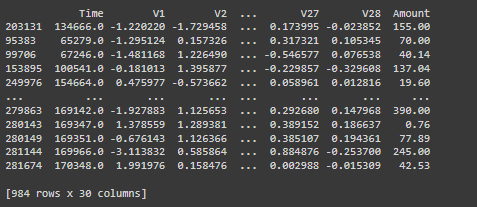


**Splitting the data into Features & Targets**

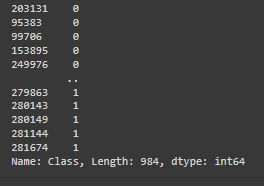
X = new\_dataset.drop(columns='Class', axis=1)

Y = new\_dataset['Class']

**print(X)**

****

**print(Y)**

****

**Split the data into Training data & Testing Data**

X\_train, X\_test, Y\_train, Y\_test = train\_test\_split(X, Y, test\_size=0.2, stratify=Y, random\_state=2)

print(X.shape, X\_train.shape, X\_test.shape)



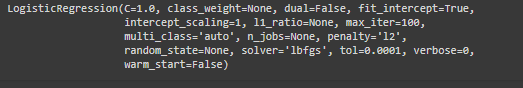
**Model Training**

**Logistic Regression**

model = LogisticRegression()

# training the Logistic Regression Model with Training Data

model.fit(X\_train, Y\_train)



**Model Evaluation**

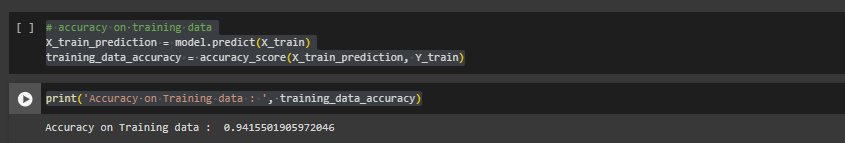
**Accuracy Score**

# accuracy on training data

X\_train\_prediction = model.predict(X\_train)

training\_data\_accuracy = accuracy\_score(X\_train\_prediction, Y\_train)

print('Accuracy on Training data : ', training\_data\_accuracy)



# accuracy on test data

X\_test\_prediction = model.predict(X\_test)

test\_data\_accuracy = accuracy\_score(X\_test\_prediction, Y\_test)

print('Accuracy score on Test Data : ', test\_data\_accuracy)

