**Credit Card Fraud Detection**

**Phase 3: Development Part 1**

Loading and preprocessing a dataset for credit card fraud detection typically involves handling imbalanced classes, scaling features, and potentially reducing dimensionality for model training. Below are the steps to load and preprocess a credit card fraud detection dataset using Python. We'll use the common credit card fraud dataset available on Kaggle as an example.

**1. Import Libraries:**

import pandas as pd

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

from sklearn.preprocessing import StandardScaler

from imblearn.over\_sampling import SMOTE

**2. Load the Dataset:**

# Load your credit card fraud dataset (replace with your actual dataset path)

df = pd.read\_csv("credit\_card\_fraud\_dataset.csv")

**3. Explore the Dataset:**

Take a quick look at the dataset to understand its structure:

print(df.head())

print(df.info())

print(df['Class'].value\_counts())

**4. Split Data into Features and Target:**

X = df.drop('Class', axis=1)

y = df['Class']

**5. Handle Imbalanced Classes:**

Credit card fraud datasets are often highly imbalanced, with a small number of fraudulent transactions. You can oversample the minority class using techniques like SMOTE (Synthetic Minority Over-sampling Technique):

smote = SMOTE(sampling\_strategy=0.5) # Adjust the sampling\_strategy as needed

X\_resampled, y\_resampled = smote.fit\_resample(X, y)

**6. Split Data into Training and Testing Sets:**

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X\_resampled, y\_resampled, test\_size=0.2, random\_state=42)

**7. Feature Scaling:**

Scale the features to have zero mean and unit variance, which is often necessary for machine learning algorithms:

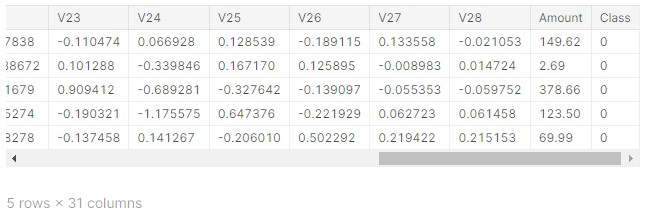
scaler = StandardScaler()

X\_train = scaler.fit\_transform(X\_train)

X\_test = scaler.transform(X\_test)

Now you have your dataset loaded, balanced, and preprocessed. You can proceed to build and train a machine learning model for credit card fraud detection using the X\_train and y\_train data. Keep in mind that the choice of model (e.g., logistic regression, random forest, XGBoost) and hyperparameter tuning is a significant part of this process. Additionally, you may want to perform dimensionality reduction (e.g., PCA) or other feature engineering based on your specific dataset and goals.

**Dataset Link:**[**https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud**](https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud)



**Importing Imortant Libraries:**

import numpy as np # linear algebra

import pandas as pd # data processing, CSV file I/O (e.g. pd.read\_csv)

import matplotlib.pyplot as plt

import seaborn as sns

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

from sklearn.preprocessing import PowerTransformer

from sklearn.preprocessing import StandardScaler

from sklearn.metrics import classification\_report

from sklearn.linear\_model import LogisticRegression

from sklearn.ensemble import RandomForestClassifier

import xgboost as xgb

Clean and preprocess the data, handle missing values, and normalize features. Data preprocessing is a crucial step in credit card fraud detection as it helps prepare the raw transaction data for analysis and model training. Here are the key data preprocessing steps involved in credit card fraud detection:

**A)Data Loading:**

Load the raw transaction data from your data source into your chosen data analysis tool or programming environment (e.g., Python with pandas).

**B)Data Exploration:**

Perform initial data exploration to understand the dataset's structure, features, and data types.Check for missing values in the dataset and decide how to handle them (e.g., imputation or removal).Examine summary statistics to gain insights into transaction amounts, timestamps, and other relevant features.

**C)Data Cleaning:**

Remove or handle duplicate records if they exist in the dataset.Address missing values by imputing them with appropriate values, such as the mean, median, or using advanced imputation methods like K-nearest neighbors or regression imputation.Correct any data format issues, such as inconsistent date/time formats.

**PYTHON PROGRAM:**

import matplotlib.pyplot as plt

import seaborn as sns

from matplotlib import gridspec

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

from sklearn.ensemble import RandomForestClassifier

from sklearn.metrics import (classification\_report, accuracy\_score,

precision\_score, recall\_score,

f1\_score, matthews\_corrcoef,

confusion\_matrix)

credit=pd.read\_csv('/kaggle/input/creditcardfraud/creditcard.csv')

credit.head()

credit.shape

credit.describe().T

fraud = credit[credit['Class'] == 1]

valid = credit[credit['Class'] == 0]

fraction = len(fraud)/float(len(valid))

print(fraction)

print("Fraud Cases: {}".format(len(credit[credit['Class'] == 1])))

print("Valid Cases: {}".format(len(credit[credit['Class'] == 0])))

print("Amount of details for the Fraudulent Transaction")

fraud.Amount.describe()

print("Amount of details for Normal Transaction")

valid.Amount.describe()

corrmat = credit.corr()

fig = plt.figure(figsize=(12, 9))

sns.heatmap(corrmat, vmax=.8, square=True)

plt.show()