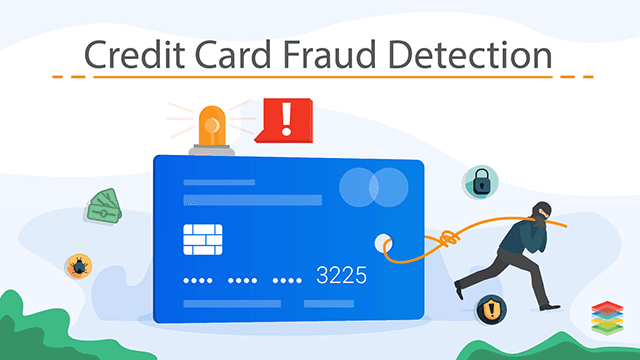
TITLE : CREDIT CARD FRAUD DETECTION

PHASE 4 : DEVELOPMENT PART 2

TEAM MEMBER NAME : ABINAYA K



CONTENT

Consider building the project by performing different activities like feature engineering, model training, evaluation etc as per the instructions in the project.

INTRODUCTION

Credit card fraud happens when a fraudster gets hold of someone else’s credit card details and makes a purchase with it. This is clear fraud, where the goal is to not pay for a good or service and still receive it.

Note that there is also another type of credit card fraud that happens when the cardholder is being dishonest. In that scenario, the payment looks legitimate, but the cardholder has already decided to return the item or ask for a refund.

The latter is called friendly fraud, and it can be challenging to detect. The cardholder may say that the card has been stolen whereas, in fact, they were the one who made the purchase but claim otherwise.

In both scenarios, however, the key ingredients are the same. For credit card fraud to work you need:

A credit card number (legitimate or stolen).

A CNP purchase (card not present), for instance at an online store.

A request for a refund. This will be made by the victim whose stolen card was used in the fraudulent purchase or the legitimate cardholder

GIVEN DATASET

In this section continue building the project by performing different activities like feature engineering, model training, evaluation etc as per the instructions in the project.



The dataset is made up of simulated credit card transactions for the period 01-Jan-2019 to 31-Dec-2020. It contains both legitimate and fraudulent transactions of 1000 customers with a pool of 800 merchants. The dataset is generated using Sparkov Data Generation | Github tool created by Brandon Harris.

**Dataset:** <https://www.kaggle.com/kartik2112/fraud-detection>

**Data Dictionary:**

index — Row ID

cc\_num — Credit Card Number of the customer

merchant — Merchant Name

amt — Transaction Amount (in USD)

is\_fraud — Target Variable. (0=genuine, 1=fraudulent)

**Library** **Requirements**:

networkx ==2.5

scikit-learn==0.24.0

pandas==1.1.3

node2vec==0.3.3

numpy=1.19.2

communities==2.2.0

**OVERVIEW FOR PROCESS OF CREDIT CARD FRAUD DETECTION**

Credit card fraud detection is a critical component of financial security, helping to safeguard both consumers and financial institutions from unauthorized and fraudulent transactions. This process involves the use of advanced technology and data analysis to identify and prevent fraudulent activities. Here is an overview of the key steps involved in credit card fraud detection:

1. **Data Collection:**
   * The process begins with the collection of vast amounts of data. This data includes transaction details, customer profiles, and historical transaction records.
2. **Data Preprocessing:**
   * Raw data is often noisy and requires preprocessing to make it suitable for analysis. This step involves data cleaning, normalization, and feature extraction.
3. **Feature Engineering:**
   * Feature engineering is essential to identify meaningful patterns. It involves selecting and creating relevant features from the data that can help in detecting fraud. These features may include transaction amounts, locations, time of day, and more.
4. **Machine Learning Models:**
   * Machine learning models play a central role in credit card fraud detection. Various algorithms, such as logistic regression, decision trees, random forests, and neural networks, are used to build predictive models. These models are trained on historical data to recognize patterns associated with legitimate and fraudulent transactions.
5. **Anomaly Detection:**
   * One of the primary methods for fraud detection is anomaly detection. This technique flags transactions that deviate significantly from the norm as potential fraud. Unusual patterns, such as high-value transactions, multiple transactions in a short time, or transactions from unusual locations, are often classified as anomalies.
6. **Rule-Based Systems:**
   * Rule-based systems are used to set predefined rules and thresholds. For example, if a cardholder suddenly makes a large transaction in a distant location, the system may flag this as suspicious and require further verification.
7. **Real-Time Monitoring:**
   * Credit card transactions are monitored in real-time, allowing for immediate responses to potential fraud. When a transaction is flagged as suspicious, the system can send an alert to the cardholder or block the transaction pending further verification.
8. **Machine Learning Updates**:
   * Machine learning models need continuous updates to adapt to evolving fraud patterns. As fraudsters change their tactics, models must be retrained with new data to stay effective.
9. **Behavioral Analysis:**
   * Analyzing the cardholder's behavior over time is crucial. This involves tracking spending patterns, transaction frequency, and preferred locations. Sudden deviations from this behavior can raise red flags.
10. **Collaboration and Information Sharing:**
    * Financial institutions often collaborate with other institutions and law enforcement agencies to share information about known fraud patterns and suspicious activities.
11. **Human Intervention:**
    * In some cases, human analysts play a role in reviewing flagged transactions and making the final decision on whether a transaction is fraudulent. This human element adds an additional layer of security.
12. **Fraud Prevention:**
    * Beyond detection, institutions also invest in fraud prevention measures, such as secure card issuance, two-factor authentication, and customer education, to reduce the likelihood of fraudulent activity.

Credit card fraud detection is an ongoing, dynamic process that combines technology, data analysis, and human oversight to protect cardholders and financial institutions from the ever-evolving tactics of fraudsters. It is an essential component of financial security in an increasingly digital and interconnected world.

FEATURE ENGINEERING :

Feature engineering is a critical step in building a credit card fraud detection system. While it's challenging to represent features in real-time pictorial format, you can create, extract, and engineer various features from credit card transaction data. These features will help your model identify fraudulent activities. Here are some feature engineering ideas:

**Transaction Amount Statistics**:

* + Mean and standard deviation of transaction amounts.
  + Scaling transaction amounts, e.g., by dividing by the mean or median

**Time-Related Features:**

* + Hour of the day and day of the week.
  + Time since the last transaction.

**Merchant Information**:

* + Number of transactions at the same merchant.
  + Frequency of transactions with a specific merchant.
  + The average transaction amount at the merchant.

**Geographic Features**:

* + The distance between the transaction location and the cardholder's home address.
  + IP address information to identify the location of online transactions.

**Cardholder Behavior**:

* + Historical transaction behavior, such as average transaction amount, frequency, and location.
  + Deviation from historical behavior.

**Categorical Variables**:

* + Encoding categorical variables like merchant category, transaction type, etc., using one-hot encoding or label encoding.

**Time Series Analysis**:

* + Detect patterns and anomalies in time series data.
  + Features like rolling averages, exponential moving averages, and autocorrelation can be helpful

**Anomaly Scores**:

* + Use anomaly detection algorithms to generate anomaly scores for each transaction and use these scores as features.

**Text Data Analysis**:

* + If transaction descriptions are available, you can analyze text data for patterns or keywords related to fraud.

**Historical Transactions**:

* + Features based on the cardholder's past transaction history, such as the number of recent transactions, the time since the last transaction, and total spend over a period.

**Aggregated Features**:

* + Features that aggregate information over time, e.g., sum of transaction amounts over the last 24 hours.

**Velocity Checks**:

* + Features related to the speed of transactions, such as the number of transactions within a short time frame.

**Social Network Analysis**:

* + Features that analyze the relationships between cardholders and merchants to detect suspicious connections.

**Cross-Validation Features**:

* + Features derived from cross-validation or rolling windows to detect changes in transaction patterns.

**Account and Cardholder Information**:

* + Features related to account age, credit limit, available credit, and cardholder demographics.

**Blacklist and Whitelist Checks**:

* + Features that flag transactions involving blacklisted or whitelisted merchants or locations.

**PCA or Dimensionality Reduction features**:

* + Using techniques like Principal Component Analysis (PCA) to reduce the dimensionality of the data while preserving important information.

Remember to visualize these features for a better understanding of their distribution and relationships. However, pictorial representations might be more useful for the final output or model evaluation rather than feature engineering.

For real-time monitoring, continuously update and compute these features as new data arrives. Monitor the distribution of these features over time to detect any significant deviations that could signal fraudulent activity. Visualizations like line plots, scatter plots, histograms, and heatmaps can help in understanding these distributions and anomalies

Table 1

**Classification confusion matrix**

|  |  |  |
| --- | --- | --- |
|  | Actual positive  Y=1 | Actual negative  Y=0 |
| Predicted positive | True positive (TP) | False positive (FP) |
| Predicted negative  c = 0 | False negative (FN) | True positive (TN) |

Table 2

**Cost matrix**

|  |  |  |
| --- | --- | --- |
|  | Actual positive | Actual negative |
| Predicted positive  ci = 1 | CT Pi | CF Pi |
| Predicted negative  ci = 0 | CFNi | CT Ni |

From this table, several statistics are extracted. In particular:

• Accuracy = T P+T N T P+T N+F P+FN

• Recall = T P T P+FN

• Precision = T P T P+F P

• F1Score = 2 Precision·Recall Precision+Recall

Table 3

**Credit card fraud cost matrix**

|  |  |  |
| --- | --- | --- |
|  | Actualpositive  yi = 1 | Actual negative  yi = 0 |
| Predicted positive  CI=1 | CT Pi = Ca | CF Pi = C |
| Predicted negative  CI=0 | CFN = Amti | CT Ni = 0 |

## **Objective — Fraudulent Transactions as Edge Classification Problem:**

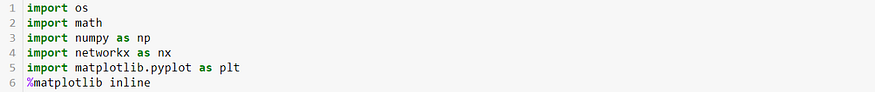
We will be representing the transactions as graphs with -

* Customers & Merchants as Nodes
* Transactions as Edges
* Weight of the Edges in line with Amount of Transaction carried out
* Label of the Edges as ‘Genuine’ or ‘Fraudulent’.

Hence, our objective of detecting fraudulent credit card transactions can be translated into a ‘Node Classification’ task in Graph Machine Learning parlance.

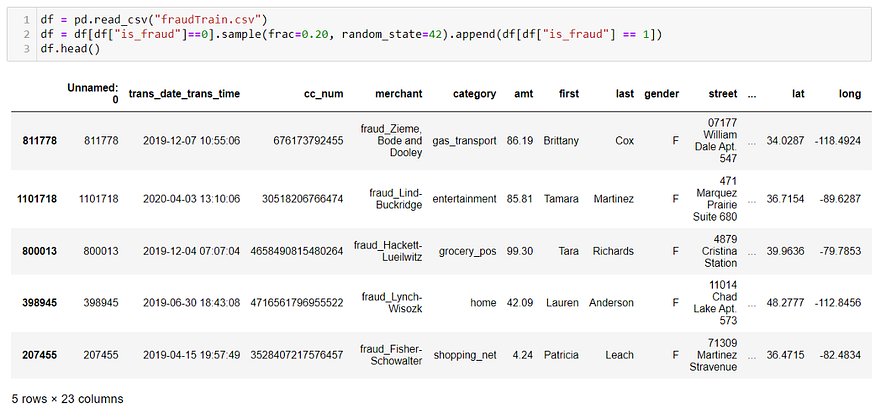
**Exploratory Data Analysis**

**Step 1: Load the libraries**



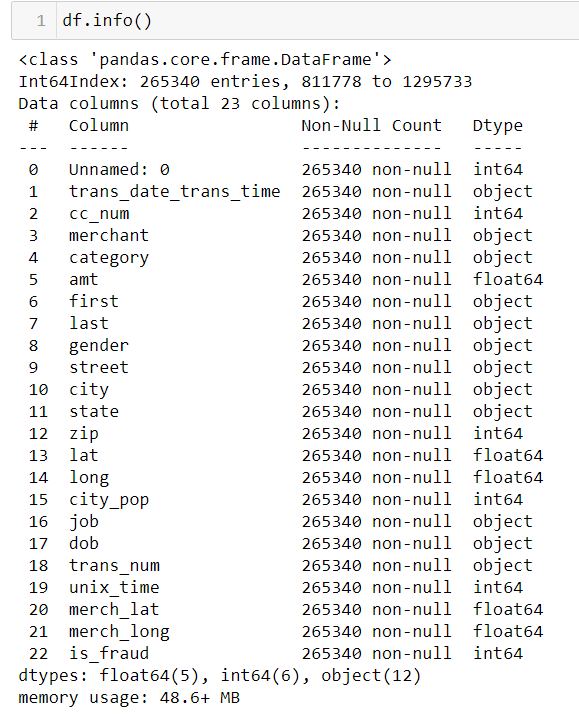
**Step 2: Load the dataset**

From the original dataset, we have selected 20% of the genuine transactions and all of the fraudulent transactions for demonstration purposes.



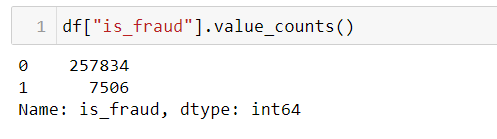
**Step 3: Data Intuition**

The dataset has 23 features and 265,340 records. No null values are present in the dataset.



**Step 4: Target Variable Distribution**

Out of the total 265,340 records, 257,834 are genuine and 7506 are fraudulent. Hence, the dataset exhibits class imbalance with 2.83% of fraudulent transactions.



The dataset can be converted into a graph network using the networkx python library.

MODEL TRAINING

Training a credit card fraud detection model is a complex task that involves a combination of data preprocessing, feature engineering, and machine learning. Below is a simplified example of how you can approach this task using Python and some popular libraries. Please note that a real-world credit card fraud detection model would require more extensive data preprocessing, feature engineering, and a larger, more diverse dataset.

**Data Preparation** :

Start by importing the necessary libraries and loading your dataset.

Make sure you have a dataset with labeled examples (fraudulent and non- fraudulent transactions).

Split the data into training and testing sets.

import pandas as pd

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

# Load your dataset

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

# Split data into features (X) and labels (y)

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

y = data['Class']

# Split the data into training and testing sets

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

**Data Preprocessing**:

* Perform data preprocessing steps, such as normalization and handling missing values.

from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()

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

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

**Model training:**

Choose a machine learning model to use. In this example, we'll use a simple logistic regression model.

from sklearn.linear\_model import LogisticRegression

# Create and train the model

model = LogisticRegression()

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

**MODEL EVALUATION**

Evaluate the model's performance on the test dataset using appropriate metrics like accuracy, precision, recall, and F1-score.

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

# Make predictions on the test set

y\_pred = model.predict(X\_test)

# Evaluate the model

print("Confusion Matrix:\n", confusion\_matrix(y\_test, y\_pred))

print("\nClassification Report:\n", classification\_report(y\_test, y\_pred))

print("Accuracy:", accuracy\_score(y\_test, y\_pred))

1.Tune the Model:

You can improve the model's performance by hyperparameter tuning or trying different algorithms like Random Forest, XGBoost, or neural networks.

**2.Cross-Validation**:

It's a good practice to perform cross-validation to ensure your model's generalization capability.

**3.Handling Imbalanced Data**:

Credit card fraud datasets are usually highly imbalanced. You may need to use techniques like oversampling (SMOTE), undersampling, or anomaly detection methods to address this issue.

**4.Deployment**:

Once you are satisfied with your model's performance, you can deploy it for real-time or batch processing of credit card transactions.

Evaluating a credit card fraud detection model typically involves assessing its performance in terms of accuracy, precision, recall, F1-score, and other relevant metrics. Below is a step-by-step guide on how to evaluate a credit card fraud detection model using Python and scikit-learn:

# Import necessary libraries

import pandas as pd

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

from sklearn.ensemble import RandomForestClassifier

from sklearn.metrics import accuracy\_score, precision\_score, recall\_score, f1\_score, confusion\_matrix

# Load your dataset (replace 'data.csv' with your dataset)

data = pd.read\_csv('data.csv')

# Split the data into features (X) and labels (y)

X = data.drop('Class', axis=1) # Features

y = data['Class'] # Labels

# Split the data into training and testing sets (e.g., 80% training and 20% testing)

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

# Initialize and train a machine learning model (Random Forest in this example)

model = RandomForestClassifier(n\_estimators=100, random\_state=42)

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

# Make predictions on the test data

y\_pred = model.predict(X\_test)

# Evaluate the model

accuracy = accuracy\_score(y\_test, y\_pred)

precision = precision\_score(y\_test, y\_pred)

recall = recall\_score(y\_test, y\_pred)

f1 = f1\_score(y\_test, y\_pred)

confusion = confusion\_matrix(y\_test, y\_pred)

# Print the evaluation metrics

print(f"Accuracy: {accuracy:.2f}")

print(f"Precision: {precision:.2f}")

print(f"Recall: {recall:.2f}")

print(f"F1 Score: {f1:.2f}")

print("Confusion Matrix:\n", confusion)

CONCLUSION

In conclusion, credit card fraud detection is a critical and evolving aspect of financial security. Utilizing advanced technology, data analysis, and machine learning, businesses and financial institutions can proactively identify and prevent fraudulent transactions, safeguarding the interests of both customers and the industry as a whole.