**CREDIT CARD FRAUD DETECTION WITH**

**APPLIED DATA SCIENCE**

**Phase 4: Development Part 2**

Credit card fraud is a prevalent concern in the modern financial landscape, with criminals continually finding new ways to engage in unauthorized and fraudulent transactions. To combat this issue, machine learning and data science techniques have become indispensable tools for financial institutions and businesses.

The provided Python program offers a comprehensive framework for credit card fraud detection, including essential steps such as data loading, data preprocessing, model training, and model evaluation.

**Program:**

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, roc\_auc\_score, confusion\_matrix

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

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

y = data['Class']

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

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

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

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

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)

roc\_auc = roc\_auc\_score(y\_test, y\_pred)

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

print("Accuracy:", accuracy)

print("Precision:", precision)

print("Recall:", recall)

print("F1 Score:", f1)

print("ROC AUC Score:", roc\_auc)

print("Confusion Matrix:\n", conf\_matrix)

* This code appears to be a python script that works with a dataset using the pandas library. Here’s a breakdown of the process:

**Program explanation:**

* **Import Necessary Libraries:**

This step imports the required Python libraries for working with data, building a random forest classifier, and evaluating its performance. These libraries include Pandas for data manipulation, scikit-learn for machine learning, and various metrics for evaluating the classifier.

**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, roc\_auc\_score, confusion\_matrix**

* **Load the Dataset:**

In this step, the program reads a dataset from a CSV file named 'credit\_card\_data.csv' using the Pandas library. This dataset is assumed to contain information about credit card transactions, including a 'Class' column that indicates whether each transaction is fraudulent (1) or not (0).

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

* **Split the Data into Features and Target**

In this step, the dataset is divided into two parts. The features (X) contain all columns except the 'Class' column, while the target variable (y) contains the 'Class' column, which represents whether a transaction is fraudulent or not.

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

**y = data['Class']**

* **Split the Data into Training and Testing Sets:**

The dataset is split into training and testing sets using the train test split function from scikit-learn. 80% of the data is used for training (X train and y train), and 20% is reserved for testing (X test and y test). The random state parameter is set to ensure reproducibility**.**

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

* **Model Training:**

A random forest classifier is created with 100 decision trees (n estimators=100) and trained using the training data. This model is used to predict whether credit card transactions are fraudulent or not**.**

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

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

* **Make Predictions:**

The model is used to make predictions on the test data, and the predicted labels are stored in the y pred variable.

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

* **Evaluation:**

In this step, various metrics are calculated to evaluate the model's performance, including accuracy, precision, recall, F1 score, ROC AUC score, and the confusion matrix. These metrics provide insights into how well the model is performing in terms of correctly identifying fraudulent and non-fraudulent transactions.

**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)**

**roc\_auc = roc\_auc\_score(y\_test, y\_pred)**

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

* **Print the Evaluation Metrics:**

Finally, the code prints out the calculated evaluation metrics to assess the model's performance in detecting credit card fraud. These metrics provide valuable information on how well the model is performing in terms of true positives, true negatives, false positives, and false negatives.

**print("Accuracy:", accuracy)**

**print("Precision:", precision)**

**print("Recall:", recall)**

**print("F1 Score:", f1)**

**print("ROC AUC Score:", roc\_auc)**

**print("Confusion Matrix:\n", conf\_matrix)**

**conclusion:**

This program can play a crucial role in enhancing the security of credit card transactions, enabling financial institutions to proactively identify and address fraudulent activities. Its adaptability and scalability make it a valuable asset in the ongoing battle against credit card fraud. Furthermore, continuous monitoring, periodic retraining, and ongoing refinement of the model are essential to ensure its effectiveness in an ever-evolving landscape of fraud**.**