APPLIED DATA SCIENCE

Vaishnavivelusamy

VVCET

CREDIT CARD FRAUD DEDECTION

INTRODUCTION

PROBLEM DEFINITION

DEEP LEARING

REAL-TIME MONITORING

IMPLANCED DATA HANDLING

DISCRIPTIVE ANALYSIST

ANOMALY DETECTION

PREDICTIVE ANALYSIST

NETWORK ANALYSIS

TIME SERIES ANALYSIS

CONCLUSION

INTRODUCTION

Machine learning algorithms play a cru cial role identifying fraudulent transactions b y analyzing patterns and anomalies in credit card data.

Commonly used machine learning algori thms for credit card fraud detection include l ogistic regression, decision trees, random for ests, and neural networks.

PROBLEM DEFINITION

The problem is to develop a machine learning-based system for

real-time credit card fraud detection.

The goal is to create a solution that can accurately identify fraudulent transactions

while minimizing false positives.

This project involves data preprocessing, feature engineering, model selection, training, and

evaluation to create a robust fraud detection systems

What the Credit Card Fraud Dedection?

- Credit card fraud is the act of using another person’s credit card to make purchases or request cash advances without the cardholder’s knowledge or consent.
- These criminals may obtain the card itself through physical theft, though increasingly fraudsters are leveraging digital means to steal the credit card number and accompanying personal information to make illicit transactions

Types of credit card fraud

Credit card fraud falls into two basic categories:

1) Card present fraud

2) Card-not-present fraud

DIFFERENT ANALYSIS

Deep Learning for Fraud Detection

- Explore advanced neural network architectures like convolutional neural
- networks (CNNs) and recurrent neural networks (RNNs) for more
- sophisticated feature extraction and time series analysis.

- You can use deep learning models, such as a neural network,
- to capture complex patterns in the data.

PROGRAM

```
from tensorflow.keras.models import Sequential
 from tensorflow.keras.layers import Dense
model = Sequential([
                    Dense(128,
input_dim=X_train.shape[1], activation='relu'),
                    Dense(64, activation='relu'),
                    Dense(1, activation='sigmoid') ])
model.compile(optimizer='adam',
loss='binary_crossentropy', metrics['accuracy'])
 model.fit(X_train, y_train, epochs=10,
batch_size=32)
```

Real-time Monitoring

To detect fraud in real-time, you can set up an automated system that continuously monitors incoming transactions and applies your trained model to identify anomalies.

program

from skmultiflow.classification

import HoeffdingTree

from skmultiflow.data import FileStream

Define a stream data source (replace 'data_stream.arff' with your data source)

stream = FileStream('data_stream.arff', n_targets=2)

Create an online learning model (Hoeffding Tree)

model = HoeffdingTree()

IMPLANCED DATA HANDLNG

Credit card fraud datasets are often imbalanced, with a majority of non-fraudulent transactions. You may need to use techniques

Like oversampling, under sampling, or Synthetic Minority Over-sampling Technique (SMOTE) to handle this imbalance .

PROGRAM

from imblearn.over_sampling import SMOTE sm = SMOTE(sampling_strategy=0.5) X_resampled, y_resampled = sm.fit_resample(X_train, y_train) # Train the model with the resampled data model.fit(X_resampled, y_resampled)

Descriptive Analytics

Data Exploration:

Initial analysis to understand data characteristics.

Data Visualization:

Using charts to spot trends and patterns.

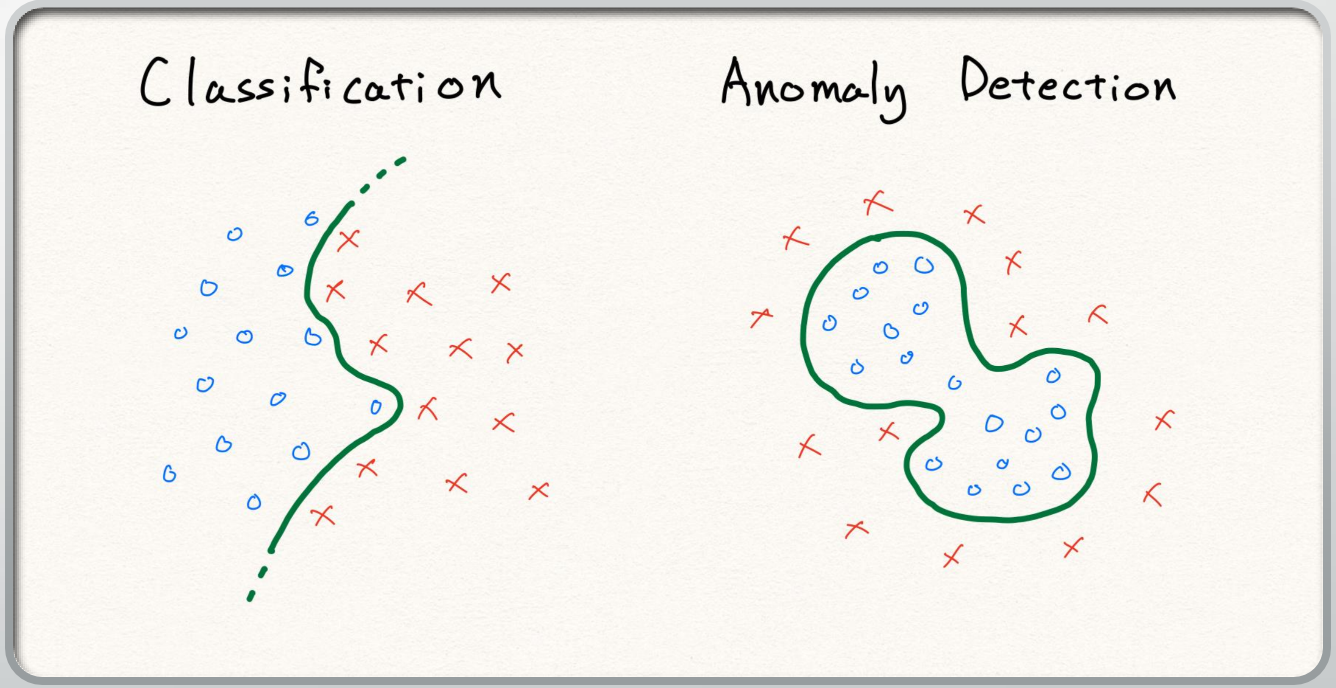
PROGRAM

```
# Data Exploration
 legitimate_transactions.describe()
fraudulent_transactions.describe()
# Data Visualization import matplotlib.pyplot as plt
plt.scatter(legitimate_transactions['amount'],
legitimate_transactions['hour'],
label='Legitimate', color='green')
plt.scatter(fraudulent_transactions['amount'],
fraudulent_transactions['hour'], label='Fraudulent',
color='red') plt.legend() plt.show()
```

Anomaly Detection

Identifying outliers in data using Isolation Forest.

Set a contamination parameter for the expected proportion of outliers.



Program

```
from sklearn.ensemble
import IsolationForest
 clf =
IsolationForest(contamination=0.05)
clf.fit(X)
 # X is your transaction data
predictions = clf.predict(X)
```

Predictive Analytics

Using logistic regression to predict fraud.

Split data into training and test sets for model evaluation.

Program:

```
from sklearn.model_selection  import train_test_split
from sklearn.linear_model  import LogisticRegression
 from sklearn.metrics  import accuracy_score,
confusion_matrix X_train,
 X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)
 model = LogisticRegression() model.fit(X_train, y_train)
 predictions = model.predict(X_test)
 accuracy = accuracy_score(y_test, predictions)
 cm = confusion_matrix(y_test,  predictions)
```

Rule-Based System

Define rules to classify transactions as legitimate or fraudulent.

Rules may be based on transaction amount, location, or other factors.

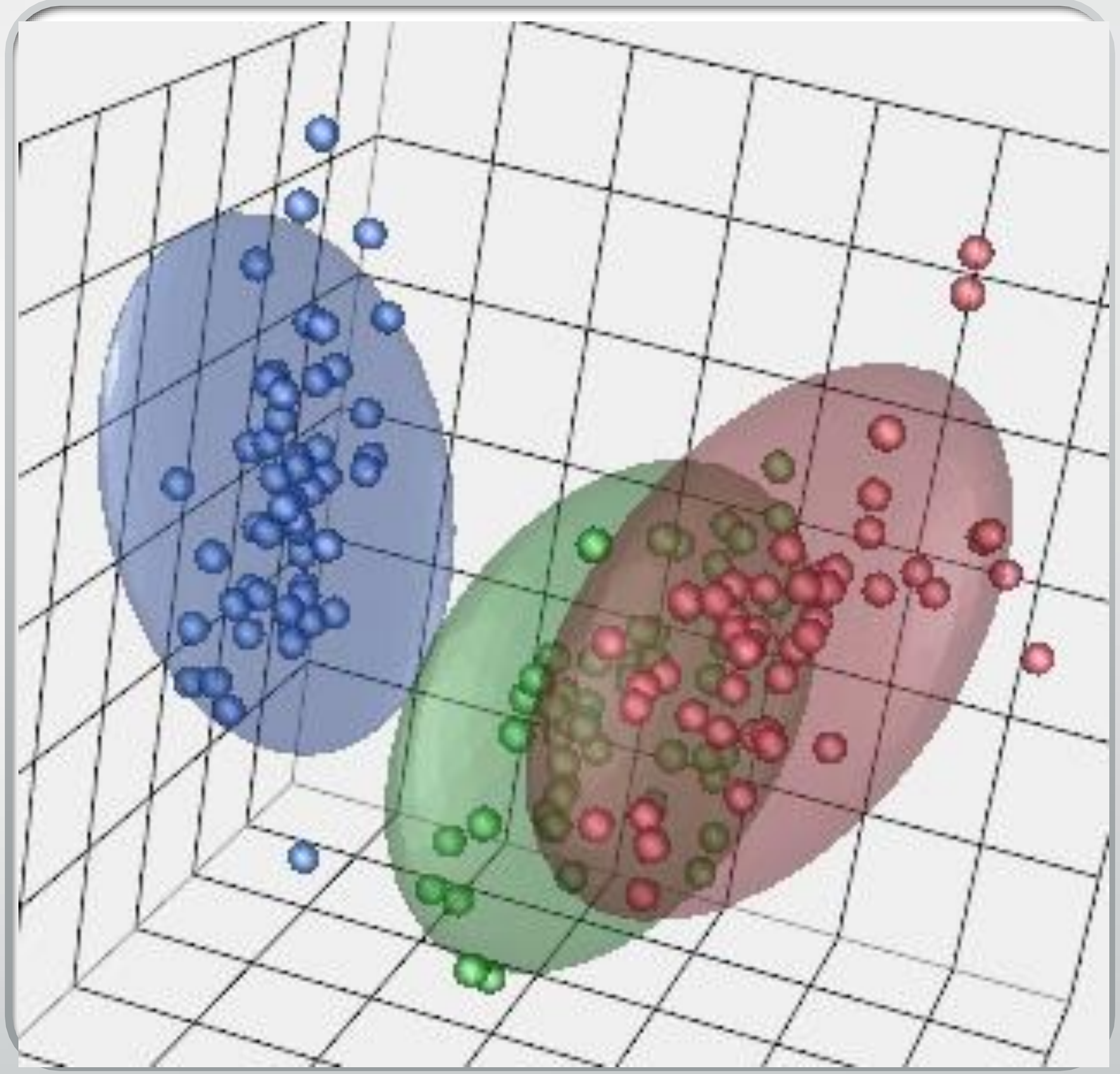
Program:

```
def rule_based_fraud_detection(transaction):
if transaction['amount'] > 1000 and
transaction['location'] not in trusted_locations:
 return "Fraudulent" else:
 return "Legitimate" result =
rule_based_fraud_detection(transaction_data)
```

Unsupervised Learning (K-Means Clustering)

Grouping transactions into clusters.

Requires specifying the number of clusters.



PROGRAM

from sklearn.cluster

import KMeans kmeans = KMeans(n_clusters=2)

kmeans.fit(X) labels = kmeans.labels_

Deep Learning (Using TensorFlow and Keras)

Building a neural network model for fraud prediction.

Training the model using labeled data and an appropriate loss function.

Network Analysis:

Examining transaction networks and relationships to identify

patterns of fraud, especially in cases of organized fraud rings.

Time Series Analysis

Studying transaction time series data to detect anomalies and seasonality in fraud patterns.

THANK YOU