Credit Card Fraud detection

key point

Loading the Dataset:

The dataset is loaded using the pandas library. This dataset contains credit card transactions, with features like transaction amount, time, and anonymized features resulting from a PCA transformation. Data Splitting:

The data is split into training and testing sets using train_test_split. 80% of the data is used for training (X_train, y_train) and 20% for testing (X_test, y_test). Isolation Forest Model:

An Isolation Forest model is initialized with a contamination parameter of 0.02, indicating that it's expected that about 2% of the data are outliers or anomalies. Model Training:

The model is trained on the training data using model.fit(X_train). Prediction:

The model predicts the labels for both the training and testing sets using model.predict(X_train) and model.predict(X_test). Label Conversion:

The predictions are converted from Isolation Forest's format (-1 for outliers, 1 for inliers) to binary labels (0 for normal, 1 for fraud). This is done to align with the original labels. Model Evaluation:

Classification reports and accuracy scores are printed to evaluate the performance of the model on both the training and testing sets.

In [1]:
```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import IsolationForest
from sklearn.metrics import classification_report, accuracy_score
```

In [2]:
```
# Load the dataset

df = pd.read_csv(r"D:\GROWINTERN PROJECTS\create\archive\creditcard.csv")
```

In [3]:
```
# Get features and labels
X = df.drop('Class', axis=1)
y = df['Class']
```

In [4]:
```
# Split 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)

# Initialize the Isolation Forest model
model = IsolationForest(contamination=0.02, random_state=42)
```

In [5]:
```
# Train the model
model.fit(X_train)
```

```
C:\Users\misal\anaconda3\envs\tensoerflows\lib\site-packages\sklearn\base.py:451: UserWarning: X does not have v
alid feature names, but IsolationForest was fitted with feature names
  "X does not have valid feature names, but"
```

Out[5]:
```
IsolationForest(contamination=0.02, random_state=42)
```

In [6]:
```
# Predict the labels
y_pred_train = model.predict(X_train)
y_pred_test = model.predict(X_test)
```

In [7]:
```
# Convert labels (-1 for outliers, 1 for inliers) to binary (0 for normal, 1 for fraud)
y_pred_train[y_pred_train == 1] = 0
y_pred_train[y_pred_train == -1] = 1
y_pred_test[y_pred_test == 1] = 0
y_pred_test[y_pred_test == -1] = 1
```

In [8]:
```
# Evaluate the model
print("Training Set:")
print(classification_report(y_train, y_pred_train))
print(f"Accuracy: {accuracy_score(y_train, y_pred_train)}")

print("\nTesting Set:")
print(classification_report(y_test, y_pred_test))
print(f"Accuracy: {accuracy_score(y_test, y_pred_test)}")
```

```
Training Set:
              precision    recall  f1-score   support

           0       1.00      0.98      0.99    227451
           1       0.06      0.74      0.12       394

    accuracy                           0.98    227845
   macro avg       0.53      0.86      0.55    227845
weighted avg       1.00      0.98      0.99    227845

Accuracy: 0.9808159055498256

Testing Set:
              precision    recall  f1-score   support

           0       1.00      0.98      0.99     56864
           1       0.07      0.78      0.12        98

    accuracy                           0.98     56962
   macro avg       0.53      0.88      0.56     56962
weighted avg       1.00      0.98      0.99     56962

Accuracy: 0.980636213616095
```

Conclusion

The Isolation Forest model is a basic example of an anomaly detection algorithm. It's capable of identifying outliers or anomalies in the dataset.

The dataset used in this code is the Credit Card Fraud Detection dataset, which is a common dataset for fraud detection tasks.

The performance of the model should be further evaluated using more advanced techniques, and hyperparameters may need tuning for better results.

In real-world scenarios, more sophisticated models and feature engineering techniques may be required for accurate fraud detection.

Additionally, a real-world application would involve continuous monitoring and updating of the model to adapt to evolving fraud patterns.

In [ ]: