الجامعة السعودية الإلكترونية
SAUDI ELECTRONIC UNIVERSITY
2011-1432
كلية الحوسبة والمعلوماتية |
College of Computing and Informatics

Second Semester – 2021/2022

Course Code	DS650
Course Name	Predictive Analytics
Assignment type	Critical Thinking
Module	12
Total Points	110 Points

Student ID	G200007615
Student Name	Abdulaziz Alqumayzi
CRN	21601

Solutions:

Critical Thinking Assignment 3

Develop a Program to Detect Credit Card Frauds

Introduction

In this exercise, we will create a program to detect credit card frauds, as well as present a python programming code and program results. The dataset from Kaggle was used: https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud?resource=download

Full Python Programming Code

```
# import the necessary packages
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
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
from sklearn.metrics import f1_score, matthews_corrcoef, confusion_matrix


df = pd.read_csv("creditcard.csv")


# Determine number of fraud cases in dataset
fraud = df[df['Class'] == 1]
valid = df[df['Class'] == 0]
outlier_Fraction = len(fraud)/float(len(valid))
print('Outlier Fraction: ',outlier_Fraction)
print('Fraud Transactions: {}'.format(len(df[df['Class'] == 1])))
print('Valid Transactions: {}'.format(len(df[df['Class'] == 0])))


print('Amount details of the fraudulent
transaction',fraud.Amount.describe())
print('details of valid transaction',valid.Amount.describe())


# dividing the X and the Y from the dataset
X = df.drop(['Class'], axis = 1)
Y = df["Class"]
print(X.shape)
print(Y.shape)
# getting just the values for the sake of processing
# (its a numpy array with no columns)
xData = X.values
yData = Y.values
```

```
# Split the data into training and testing sets
xTrain, xTest, yTrain, yTest = train_test_split(xData, yData, test_size =
0.2, random_state = 30)

# Building a Random Forest Model using skicit learn
rfc = RandomForestClassifier()
rfc.fit(xTrain, yTrain)
yPred = rfc.predict(xTest)


# Building all kinds of evaluating parameters
n_outliers = len(fraud)
n_errors = (yPred != yTest).sum()
print("The model used is Random Forest classifier")

acc = accuracy_score(yTest, yPred)
print("The accuracy is {}".format(acc))

prec = precision_score(yTest, yPred)
print("The precision is {}".format(prec))

rec = recall_score(yTest, yPred)
print("The recall is {}".format(rec))

f1 = f1_score(yTest, yPred)
print("The F1-Score is {}".format(f1))

MCC = matthews_corrcoef(yTest, yPred)
print("The Matthews correlation coefficient is{}".format(MCC))


# Visualizing the Confusion Matrix
LABELS = ['Normal', 'Fraud']
conf_matrix = confusion_matrix(yTest, yPred)
plt.figure(figsize =(10, 8))
sns.heatmap(conf_matrix, xticklabels = LABELS, yticklabels = LABELS, annot
= True, fmt ="d");
plt.title("Confusion matrix")
plt.ylabel('True class')
plt.xlabel('Predicted class')
plt.show();
```

Transactions results:

```
Outlier Fraction:  0.0017304750013189597
Fraud Transactions: 492
Valid Transactions: 284315
```

Amount details of the fraudulent transaction results:

```
Amount details of the fraudulent transaction
count     492.000000
mean      122.211321
std       256.683288
min         0.000000
```

```
25%          1.000000
50%          9.250000
75%        105.890000
max       2125.870000
Name: Amount, dtype: float64
```

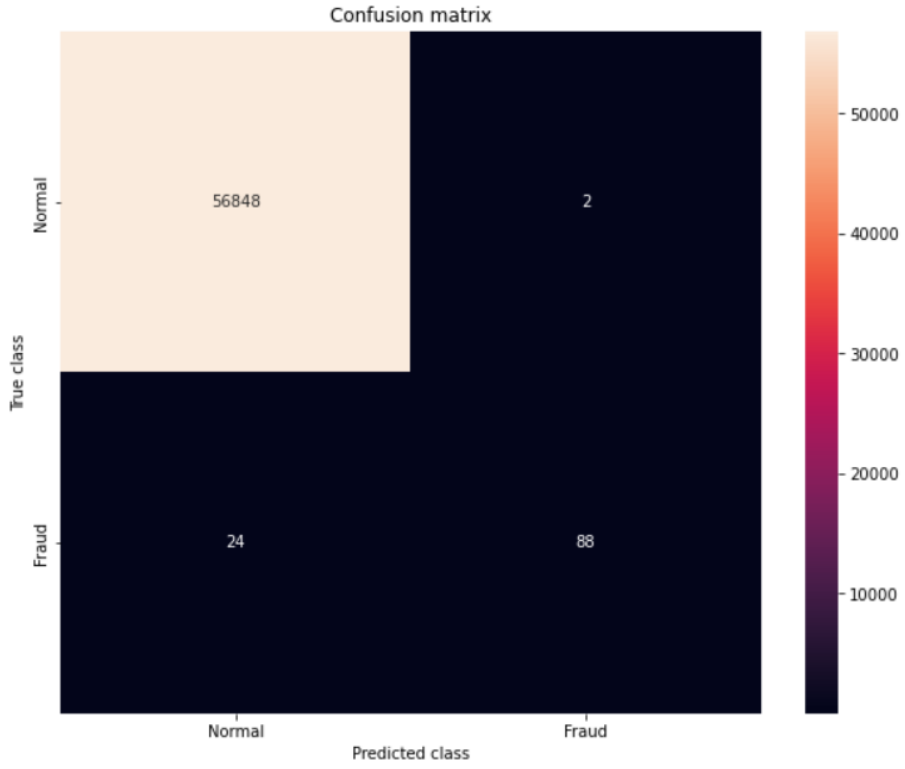
Details of valid transaction results:

```
details of valid transaction
count    284315.000000
mean         88.291022
std         250.105092
min           0.000000
25%           5.650000
50%          22.000000
75%          77.050000
max       25691.160000
Name: Amount, dtype: float64
```

Model evaluation results:

```
The model used is Random Forest classifier
The accuracy is 0.9994908886626171
The precision is 0.946236559139785
The recall is 0.7857142857142857
The F1-Score is 0.8585365853658538
The Matthews correlation coefficient is0.862007429170268
```

Confusion Matrix results:



References

Machine Learning Group . (2018, March 23). *Credit Card Fraud Detection*. Kaggle. Retrieved April 19, 2022, from https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud?resource=download

ML: Credit Card Fraud Detection. GeeksforGeeks. (2022, January 20). Retrieved April 19, 2022, from https://www.geeksforgeeks.org/ml-credit-card-fraud-detection/

Liu, Y. (2020). *Python machine learning by example Build Intelligent Systems using python, tensorflow 2, pytorch, and scikit-learn*. Packt.