import numpy as np

import pandas as pd

import matlotlib.pyplot as plt

import seaborn as sns

from matplotlib import gridspec

path ="cerdit.csv"

data =pd.read\_csv(path)

data.head()

print(data.shape)

print(data.describe())

fraud =data[data['class'] == 1]

valid =data[data['class'] == 0]

print(outlierFraction)

print('Fraud cases: {}'.format(len(data[data['Class'] == 1])))

print('Valid Transactions: {}'.format(len(data[data['Class'] == 0])))

print("Amount details of the fraudulent transaction")

fraud.Amount.describe()

print("details of valid transaction")

valid.Amount.describe()

corrmat = data.corr()

fig = plt.figure(figsize = (12, 9))

sns.heatmap(corrmat, vmax = .8, square = True)

plt.show()

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

Y = data["Class"]

print(X.shape)

print(Y.shape)

xData = X.values

yData = Y.values

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

xTrain, xTest, yTrain, yTest = train\_test\_split(

xData, yData, test\_size = 0.2, random\_state = 42)

from sklearn.ensemble import RandomForestClassifier

rfc = RandomForestClassifier()

rfc.fit(xTrain, yTrain)

yPred = rfc.predict(xTest)

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

from sklearn.metrics import precision\_score, recall\_score

from sklearn.metrics import f1\_score, matthews\_corrcoef

from sklearn.metrics import confusion\_matrix

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))

LABELS = ['Normal', 'Fraud']

conf\_matrix = confusion\_matrix(yTest, yPred)

plt.figure(figsize =(12, 12))

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()