“Credit Card Fraud Detection”

TABLE OF CONTENTS

Abstract:

The Credit Card Fraud Detection project is used to identify whether a new transaction is fraudulent or not by modelling past credit card transactions with the knowledge of the ones that turned out to be fraud. We will use various predictive models to see how Accurate they are in detecting whether a transaction is a normal payment or a fraud. Classification techniques like Decision Tree , Logistic Regression , SVC and Naïve Bayes are the promising solutions to detect the fraud and non-fraud transactions. Unfortunately, in a certain condition, classification techniques do not perform well when it comes to huge numbers of differences in data distribution.

Credit card fraud events take place frequently and then result in huge financial losses. The number of online transactions has grown in large quantities and online credit card transactions holds a huge share of these transactions. Therefore, banks and financial institutions offer credit card fraud detection applications much value and demand. Fraudulent transactions can occur in various ways and can be put into different categories. This paper focuses on four main fraud occasions in real-world transactions .We are using the datasets provided by Kaggle. This data set includes all transactions recorded over the course of two days.

Introduction:

Credit card is considered as one of the prime cases of fraud , since in a very short time attackers can get a lot of money without much risk and in most of the times the fraud is discovered after a few days. To commit a fraud through credit card, fraudsters are looking for sensitive information such as credit card number, bank account and social security numbers. In case of offline payment (by using the credit card physically), an attacker has to steal the credit card itself, while in case of online payment the fraudsters should steal the customer's identity. Credit card fraud is a significant issue which results in significant loss for banks and issuer companies. Banks take credit card fraud very seriously and have highly sophisticated security systems to monitor transactions and detect frauds. A secured and trusted banking payment system requires high speed verification and authentication mechanisms that let legitimate users easily conduct their business[5].

There are many fraud detection solutions and software which prevent frauds in business such as credit card, retail, e-commerce, insurance and industries. Data Mining Technique is one notable and popular method used in a fraud detection system. It is impossible to be certain about the true intention and rightfulness behind an application or transaction. In reality to seek out possible evidences of fraud from the available data, using mathematical algorithms is the best effective option[6]. Fraud detection in credit card is truly the process of identifying those transactions that are fraudulent into two classes legit class and fraud class. Several techniques are designed and implemented to solve this problem such as genetic algorithm, artificial neural network, frequent item set mining, machine learning algorithms, migrating birds optimization algorithm, comparative analysis of logistic regression, Density-Based SCAN, SVM, decision tree and random forest is carried out.

Credit card fraud detection is a difficult problem to solve. Firstly, due to issue of having only a limited amount of data, it becomes difficult to match a pattern for the given dataset. Secondly, there may be many entries in dataset with truncations of fraudsters which will also fit a pattern of legitimate behaviour. Also the problem has many constraints. Firstly, datasets are not easily accessible to the public and the results of researches are often hidden and censored, making the results inaccessible and due to this it is challenging to benchmarking for the models built. Datasets in previous research with real data in the literature is nowhere mentioned. Secondly, the improvement of methods is more difficult by the fact that security concerns impose a limitation to exchange of ideas and methods in fraud detection and especially in credit card fraud detection. Lastly, the datasets ate continuously evolving and changing making the profiles of normal and fraudulent behaviour always different that is the legit transaction in the past may be fraud in the present and vice versa.

Credit card transactional datasets are rarely available, highly imbalanced and skewed. Optimal feature selection for the models, suitable metric is the most important part of data mining to evaluate the performance of techniques on skewed credit card fraud data.

Proposed Methods:

Logistic Regression

Logistic Regression is a supervised classification method that returns the probability of binary dependent variable that is predicted from the independent variable of dataset that is logistic regression predict the probability of an outcome which has two values either zero or one, yes or no and false or true. Logistic regression has similarities to linear regression but as in linear regression a straight line is obtained, logistic regression shows a curve. The use of one or several predictors or independent variable is on what prediction is based, logistic regression produces logistic curves which plots the values between zero and one[6].

Regression is a regression model where the dependent variable is categorical and analyzes the relationship between multiple independent variables. There are many types of logistic regression model such as binary logistic model, multiple logistic model, binomial logistic models[2]. Binary Logistic Regression model is used to estimate the probability of a binary response based on one or more predictors[6].

$$p = \frac{e^{\alpha+\beta_n X}}{1 + e^{\alpha+\beta_n X}}$$

Above equation represents the logistic regression in mathematical form.

Decision Tree :

Decision tree is an algorithm that uses a tree like graph or model of decisions and their possible outcomes to predict the final decision, this algorithm uses conditional control statement. A Decision tree is an algorithm for approaching discrete-valued target functions, in which decision tree is denoted by a learned function. For inductive learning these types of algorithms are very famous and have been successfully applied to abroad range of tasks. We give label to a new transaction that is whether it is legit or fraud for which class label is unknown and then transaction value is tested against the decision tree, and after that from root node to output/class label for that transaction a path is traced.

Decision rules determines the outcome of the content of leaf node. In general rules have the form of 'If condition 1 and condition 2 but not condition 3 then outcome'[13]. Decision tree helps to determine the worst, best and expected values for different scenarios, simplified to understand and interpret and allows addition of new possible scenarios.

Random Forest

Random Forest is an algorithm for classification and regression[6]. Summarily, it is a collection of decision tree classifiers. Random forest has advantage over decision tree as it corrects the habit of overfitting to their training set. A subset of the training set is sampled randomly so that to train each individual tree and then a decision tree is built, each node then splits on a feature selected from a random subset of the full feature set[14]. Even for large data sets with many features and data instances training is extremely fast in random forest and because each tree is trained independently of the others[15]. The Random Forest algorithm has been found to provides a good estimate of the generalization error and to be resistant to overfitting.

Code:

```
import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

sns.set()

%matplotlib inline


df = pd.read_csv('C:/Users/HP/Documents/creditcard.csv')

print(df.shape)

df.head()

df.info()

class_names = {0:'Not Fraud', 1:'Fraud'}

print(df.Class.value_counts().rename(index = class_names))

fig = plt.figure(figsize = (15, 12))


plt.subplot(5, 6, 1) ; plt.plot(df.V1) ; plt.subplot(5, 6, 15) ; plt.plot(df.V15)

plt.subplot(5, 6, 2) ; plt.plot(df.V2) ; plt.subplot(5, 6, 16) ; plt.plot(df.V16)

plt.subplot(5, 6, 3) ; plt.plot(df.V3) ; plt.subplot(5, 6, 17) ; plt.plot(df.V17)

plt.subplot(5, 6, 4) ; plt.plot(df.V4) ; plt.subplot(5, 6, 18) ; plt.plot(df.V18)

plt.subplot(5, 6, 5) ; plt.plot(df.V5) ; plt.subplot(5, 6, 19) ; plt.plot(df.V19)

plt.subplot(5, 6, 6) ; plt.plot(df.V6) ; plt.subplot(5, 6, 20) ; plt.plot(df.V20)

plt.subplot(5, 6, 7) ; plt.plot(df.V7) ; plt.subplot(5, 6, 21) ; plt.plot(df.V21)

plt.subplot(5, 6, 8) ; plt.plot(df.V8) ; plt.subplot(5, 6, 22) ; plt.plot(df.V22)

plt.subplot(5, 6, 9) ; plt.plot(df.V9) ; plt.subplot(5, 6, 23) ; plt.plot(df.V23)

plt.subplot(5, 6, 10) ; plt.plot(df.V10) ; plt.subplot(5, 6, 24) ; plt.plot(df.V24)
plt.subplot(5, 6, 11) ; plt.plot(df.V11) ; plt.subplot(5, 6, 25) ; plt.plot(df.V25)
plt.subplot(5, 6, 12) ; plt.plot(df.V12) ; plt.subplot(5, 6, 26) ; plt.plot(df.V26)
plt.subplot(5, 6, 13) ; plt.plot(df.V13) ; plt.subplot(5, 6, 27) ; plt.plot(df.V27)
plt.subplot(5, 6, 14) ; plt.plot(df.V14) ; plt.subplot(5, 6, 28) ; plt.plot(df.V28)
plt.subplot(5, 6, 29) ; plt.plot(df.Amount)
plt.show()
from matplotlib import gridspec
features = df.iloc[:,0:28].columns
plt.figure(figsize=(12,28*4))
gs = gridspec.GridSpec(28, 1)
for i, c in enumerate(df[features]):
 ax = plt.subplot(gs[i])
 sns.distplot(df[c][df.Class == 1], bins=50)
 sns.distplot(df[c][df.Class == 0], bins=50)
 ax.set_xlabel('')
 ax.set_title('histogram of feature: ' + str(c))
plt.show()
fraud = df[df['Class'] == 1]
valid = df[df['Class'] == 0]
outlier_fraction = len(fraud)/float(len(valid))
print(outlier_fraction)
print(fraud.shape,valid.shape)
print("Amount details of fraudulent transaction")
fraud.Amount.describe()
print("details of valid transaction")
valid.Amount.describe()
```

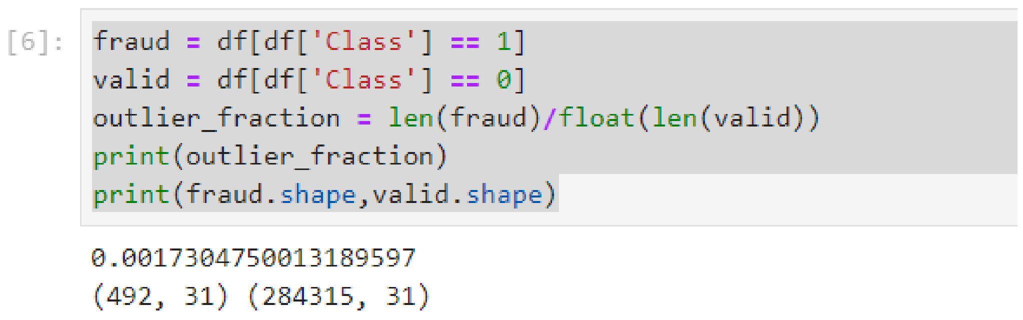
```
from sklearn.model_selection import train_test_split

feature_names = df.iloc[:, 1:30].columns

target = df.iloc[:1, 30: ].columns

print(feature_names)

print(target)

data_features = df[feature_names]

data_target = df[target]

X_train, X_test, y_train, y_test = train_test_split(data_features, data_target,
train_size=0.70, test_size=0.30, random_state=1)

print("Length of X_train is: {X_train}".format(X_train = len(X_train)))

print("Length of X_test is: {X_test}".format(X_test = len(X_test)))

print("Length of y_train is: {y_train}".format(y_train = len(y_train)))

print("Length of y_test is: {y_test}".format(y_test = len(y_test)))

from sklearn.linear_model.logistic import LogisticRegression


clf=LogisticRegression()

clf.fit(X_train, y_train)

yPred=clf.predict(X_test)

acc = accuracy_score(y_test, yPred)

print("The accuracy of Logistic Regression is {}".format(acc))

from sklearn.metrics import classification_report,
accuracy_score,precision_score,recall_score,f1_score

from sklearn.metrics import confusion_matrix

n_outliers = len(fraud)


print("The model used is Logistic Regression classifier")
```

```
acc= accuracy_score(y_test,yPred)

print("The accuracy is {}".format(acc))

prec= precision_score(y_test,yPred)

print("The precision is {}".format(prec))

rec= recall_score(y_test,yPred)

print("The recall is {}".format(rec))

f1= f1_score(y_test,yPred)

print("The F1-Score is {}".format(f1))

LABELS = ['Normal', 'Fraud']

conf_matrix = confusion_matrix(y_test, 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()

#RandomForest

from sklearn.ensemble import RandomForestClassifier


rfc = RandomForestClassifier()

rfc.fit(X_train, y_train)


yPred = rfc.predict(X_test)

from sklearn.metrics import accuracy_score

acc = accuracy_score(y_test, yPred)
```

```
print("The accuracy of Random forest is {}".format(acc))

from sklearn.metrics import classification_report,
accuracy_score,precision_score,recall_score,f1_score

from sklearn.metrics import confusion_matrix


acc= accuracy_score(y_test,yPred)

print("The accuracy is {}".format(acc))

prec= precision_score(y_test,yPred)

print("The precision is {}".format(prec))

rec= recall_score(y_test,yPred)

print("The recall is {}".format(rec))

f1= f1_score(y_test,yPred)

print("The F1-Score is {}".format(f1))

LABELS = ['Normal', 'Fraud']

conf_matrix = confusion_matrix(y_test, 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()
```

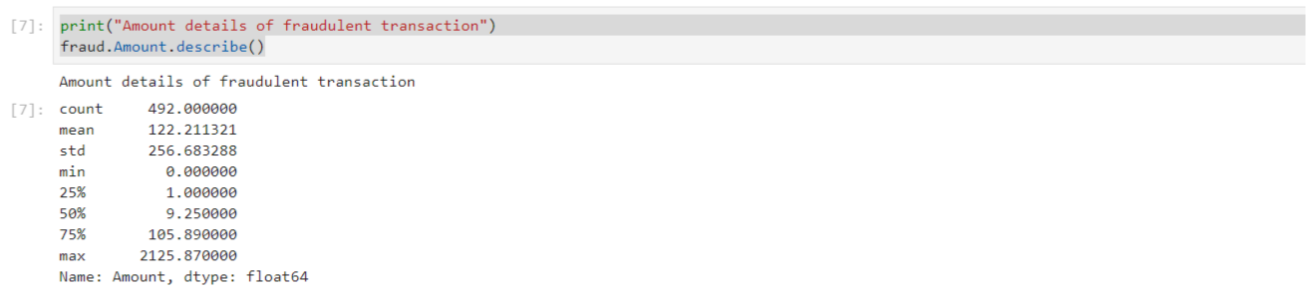
Result:

Fraud Vs Valid Transaction:



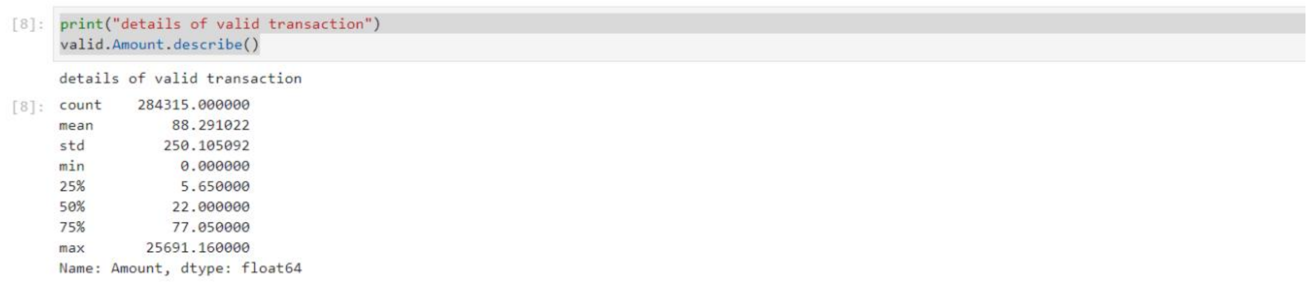
```
[6]: fraud = df[df['Class'] == 1]
     valid = df[df['Class'] == 0]
     outlier_fraction = len(fraud)/float(len(valid))
     print(outlier_fraction)
     print(fraud.shape,valid.shape)

     0.0017304750013189597
     (492, 31) (284315, 31)
```

Details of Fraud Transactions:



```
[7]: print("Amount details of fraudulent transaction")
     fraud.Amount.describe()

     Amount details of fraudulent transaction

[7]: 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
```

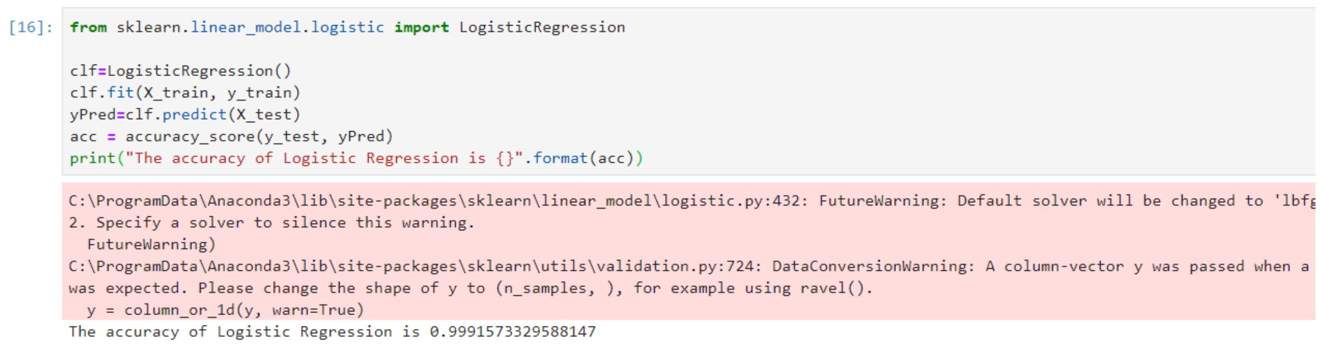
Details Valid Transactions:



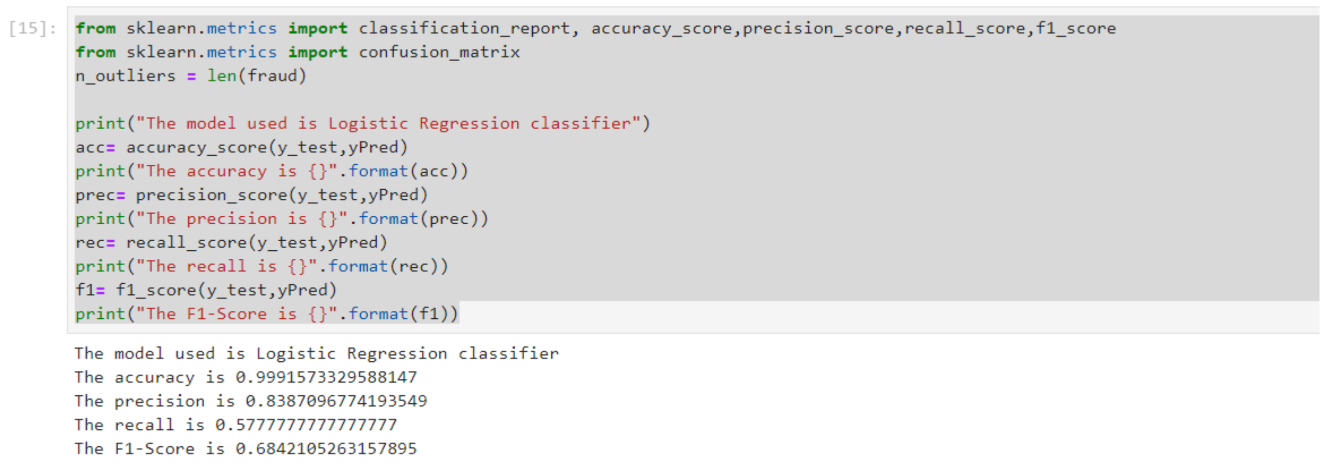
```
[8]: print("details of valid transaction")
     valid.Amount.describe()

     details of valid transaction

[8]: 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
```

Logestic Regression Classifier

Accuracy of Logestic Regression Classifier:



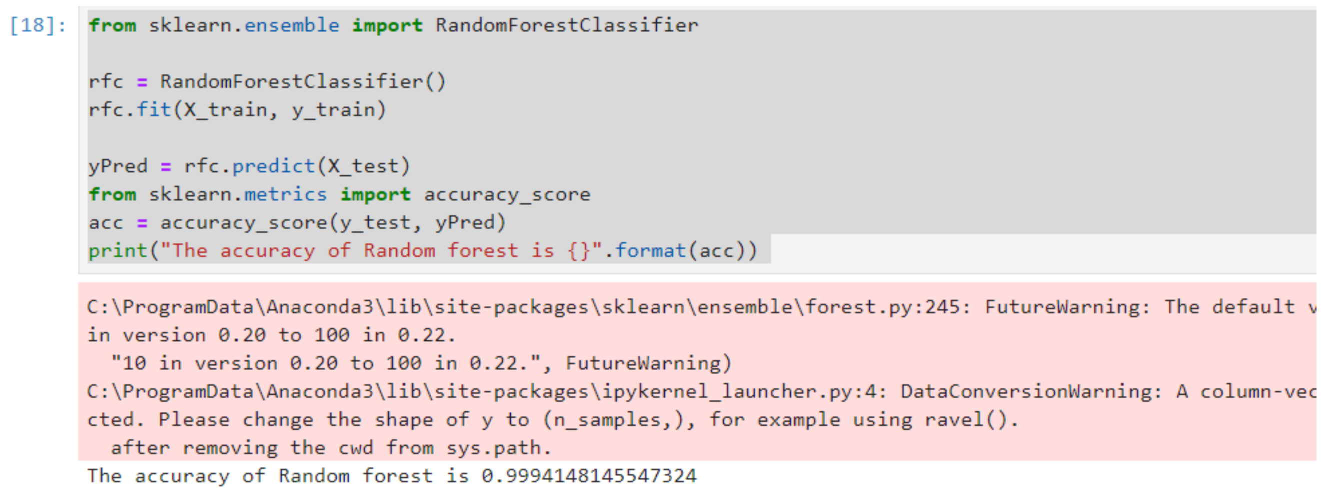
```
[16]: from sklearn.linear_model.logistic import LogisticRegression

      clf=LogisticRegression()
      clf.fit(X_train, y_train)
      yPred=clf.predict(X_test)
      acc = accuracy_score(y_test, yPred)
      print("The accuracy of Logistic Regression is {}".format(acc))

      C:\ProgramData\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:432: FutureWarning: Default solver will be changed to 'lbfg
      2. Specify a solver to silence this warning.
        FutureWarning)
      C:\ProgramData\Anaconda3\lib\site-packages\sklearn\utils\validation.py:724: DataConversionWarning: A column-vector y was passed when a
      was expected. Please change the shape of y to (n_samples, ), for example using ravel().
        y = column_or_1d(y, warn=True)
      The accuracy of Logistic Regression is 0.9991573329588147
```



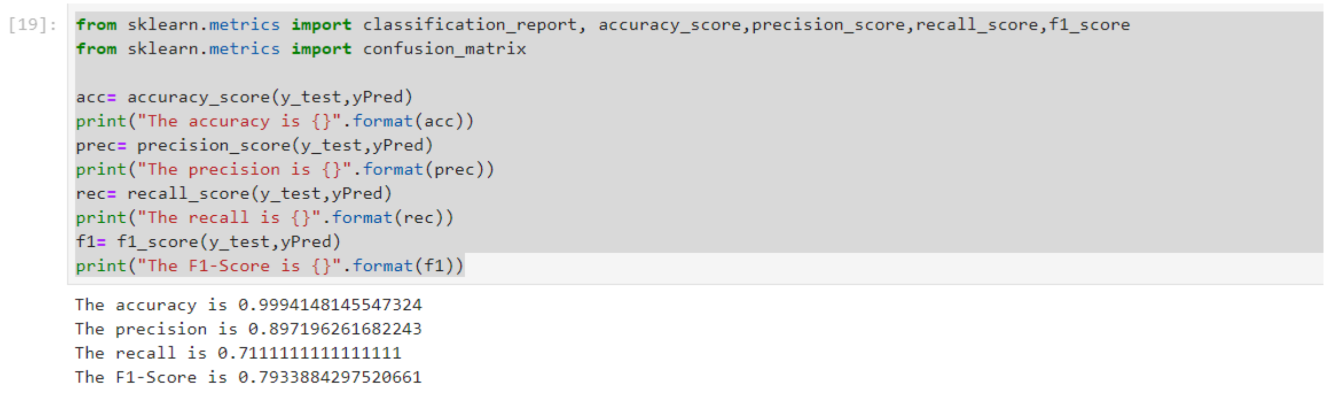
```
[15]: from sklearn.metrics import classification_report, accuracy_score,precision_score,recall_score,f1_score
      from sklearn.metrics import confusion_matrix
      n_outliers = len(fraud)

      print("The model used is Logistic Regression classifier")
      acc= accuracy_score(y_test,yPred)
      print("The accuracy is {}".format(acc))
      prec= precision_score(y_test,yPred)
      print("The precision is {}".format(prec))
      rec= recall_score(y_test,yPred)
      print("The recall is {}".format(rec))
      f1= f1_score(y_test,yPred)
      print("The F1-Score is {}".format(f1))

      The model used is Logistic Regression classifier
      The accuracy is 0.9991573329588147
      The precision is 0.8387096774193549
      The recall is 0.5777777777777777
      The F1-Score is 0.6842105263157895
```

RandomForest Classifier

Accuracy:



```
[18]: from sklearn.ensemble import RandomForestClassifier

      rfc = RandomForestClassifier()
      rfc.fit(X_train, y_train)

      yPred = rfc.predict(X_test)
      from sklearn.metrics import accuracy_score
      acc = accuracy_score(y_test, yPred)
      print("The accuracy of Random forest is {}".format(acc))

      C:\ProgramData\Anaconda3\lib\site-packages\sklearn\ensemble\forest.py:245: FutureWarning: The default v
      in version 0.20 to 100 in 0.22.
        "10 in version 0.20 to 100 in 0.22.", FutureWarning)
      C:\ProgramData\Anaconda3\lib\site-packages\ipykernel_launcher.py:4: DataConversionWarning: A column-vec
      cted. Please change the shape of y to (n_samples,), for example using ravel().
        after removing the cwd from sys.path.
      The accuracy of Random forest is 0.9994148145547324
```



```
[19]: from sklearn.metrics import classification_report, accuracy_score,precision_score,recall_score,f1_score
      from sklearn.metrics import confusion_matrix

      acc= accuracy_score(y_test,yPred)
      print("The accuracy is {}".format(acc))
      prec= precision_score(y_test,yPred)
      print("The precision is {}".format(prec))
      rec= recall_score(y_test,yPred)
      print("The recall is {}".format(rec))
      f1= f1_score(y_test,yPred)
      print("The F1-Score is {}".format(f1))

      The accuracy is 0.9994148145547324
      The precision is 0.897196261682243
      The recall is 0.7111111111111111
      The F1-Score is 0.7933884297520661
```

Conclusion:

Credit card fraud has become more and more rampant in recent years. To improve merchants' risk management level in an automatic, scientific and effective way, building an accurate, efficient and easy-handling credit card risk monitoring system is one of the key tasks for the merchant banks. In this study, three classification methods were used to a deep analysis of the credit cards history business information and have built the fraud detecting models. We present our work and demonstrate the advantages of the data mining techniques including neural networks, logistic regression and decision tree to the credit card fraud detection, for the purpose of reducing the bank's risk. The results show that the proposed classifier of neural networks and logistic regression approaches outperform decision tree in solving the problem under investigation.

References:

[1] https://aihubprojects.com/credit-card-fraud-detection/

[2] https://ieeexplore.ieee.org/abstract/document/8776942/S. Xuan, G. Liu, Z. Li, L. Zheng, S. Wang, and G. N. Surname, "Random Forest for Credit Card Fraud,""15th Int. Conf . Networking, Sens Control, 2018

[3] https://www.researchgate.net/publication/334761474_Real-time_Credit_Card_Fraud_Detection_Using_Machine_Learning/Anuruddha Thennakoon.