A PROJECT REPORT

on

"Credit Card Fraud Detection"

Submitted to

KIIT Deemed to be University

In Partial Fulfillment of the Requirement for the Award of

BACHELOR'S DEGREE IN
COMPUTER SCIENCE AND ENGINEERING

BY

ANUPAM KUMAR ANIKET	1705023
ADITYA	1705008
MANISH KUMAR MATHUR	1705046

UNDER THE GUIDANCE OF
PROF. SIDDHARTH ROUTARAY

SCHOOL OF COMPUTER ENGINEERING
KALINGA INSTITUTE OF INDUSTRIAL TECHNOLOGY
BHUBANESWAR, ODISHA - 751024

May 2020

KIIT Deemed to be University

School of Computer Engineering
Bhubaneswar, ODISHA 751024

CERTIFICATE

This is certify that the project entitled

“Credit Card Fraud Detection“

submitted by

ANUPAM KUMAR ANIKET	1705023
ADITYA	1705008
MANISH KUMAR MATHUR	1705046

is a record of bonafide work carried out by them, in the partial fulfillment of the requirement for the award of Degree of Bachelor of Engineering (Computer Science & Engineering) at KIIT Deemed to be university, Bhubaneswar. This work is done during year 2019-2020, under our guidance.

Date: 25/05/2020

Prof. SIDDHARTH ROUTARAY
Project Guide

Acknowledgments

We are profoundly grateful to Prof. SIDDHARTH ROUTARAY for his expert guidance and continuous encouragement throughout to see that this project rights its target since its commencement to its completion. ....

ANUPAM KUMAR ANIKET
ADITYA
MANISH KUMAR MATHUR

ABSTRACT

As the internet has reached the last corners of the world, there is an increase in credit card transactions which increased the risk of fraud. It is important for credit card companies to identify the fraud transaction so that the customer's interests are protected. problems can be tackled with Data Science and its importance, along with Machine Learning, cannot be overstated. This project intends to illustrate the modeling of a data set using machine learning with Credit Card Fraud Detection. The Credit Card Fraud Detection problem includes modeling past credit card transactions with the data of the ones that turned out to be a fraud. This model is then used to recognize whether a new transaction is fraudulent or not. Our objective here is to detect fraudulent transactions while minimizing incorrect fraud classifications. Credit Card Fraud Detection is a typical sample of classification. We have focused on analyzing and pre-processing the data by using SMOTE for data sampling and we have used multiple machine learning algorithm such as decision tree, random forest and logistic regression on PCA transformed cred card transaction dataset.

Keywords: Machine Learning, SMOTE, Logistic Regression. Fraud Detection, Random Forest, Decision Tree, Classification.

Contents

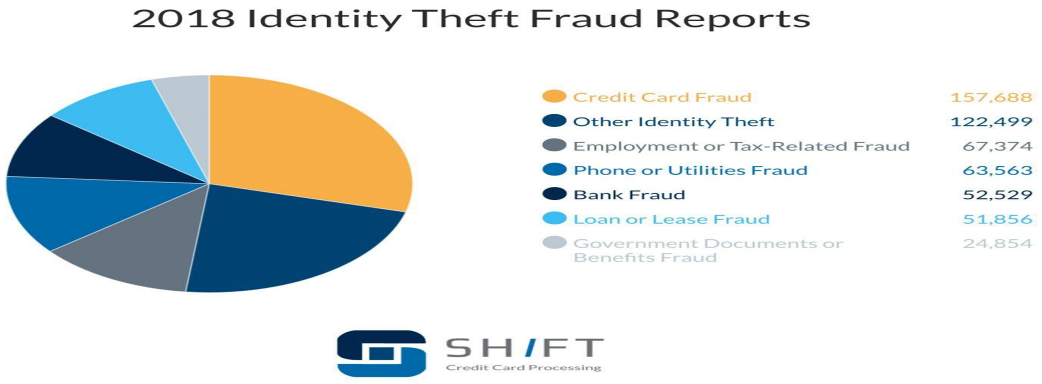
List of Figures

Introduction

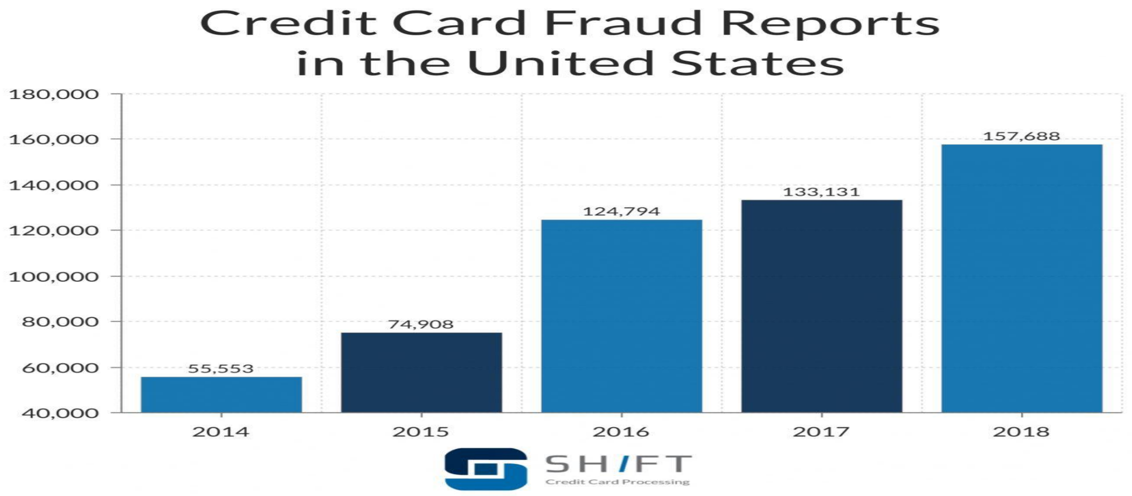
A credit card is a card that allows the users to withdraw the money in advance or purchase the goods and services within the credit limit. It provides extra time for the users to pay the principal amount and some interest on later dates

Credit card fraud can be defined as a case when a person is using someone's else credit card for personal reasons while the owner and the card-issuing authorities are unaware of the fact that the card is being used. Credit card frauds are easy targets. Fraudsters always try to make every fraudulent transaction legitimate, which makes fraud detection a very challenging and difficult task to detect.

Credit card fraud was ranked #1 type of identity thief fraud. Credit card fraud accounted for approximately 36 percent of identity thief fraud.



In 2018, $24.26 Billion was lost due to payment card fraud worldwide and is projected to rise to $35.67 billion in five years and $40.63 billion in 10 years. The USA accounts for 40% of total credit card fraud.



Literature Survey

From the work of view for preventing credit card fraud, several research works were carried out with special emphasis on classification models, data mining, class-imbalance, and neural networks. By reference of a research paper published in the International Conference on Intelligent Data Engineering and Automated Learning [7], it is identified that skewed distribution of data, the mix of Legitimate and fraudulent transactions, and lack of class balancing are the main reasons for the complexity of credit card fraud detection. A report published in First International NAISO Congress, Havana, Cuba[6], suggests a system using Bayesian and neural network techniques to learn models of fraudulent credit card transactions but it lacks to deal with bias created due to class imbalance. Another research, published in the International Journal of Recent Trends in Engineering[8], investigates the usefulness of applying different approaches like the clustering model, the probability density estimation method, and the model based on Bayesian networks for Credit card fraud detection. A technical report published in Helsinki University of Technology[9], proposed an approach that performs statistical modeling of past behavior and produces a novelty measure of current behavior as a negative log-likelihood of the dataset. The aim of this project is to remove the class-imbalance in the dataset and predict the outcome by applying different classification machine learning approaches.

Software Requirements Specification

3.1 PURPOSE

As the use of plastic money is increasing day by day there is also increasing in case of fraud in credit, Debit card, and online transactions day by day. According to RBI in 2017-2018 a total of 911 credit card fraud was registered to amount to RS65.26 crores. The purpose of this project is to use a machine-learning algorithm to detect the fraud and protect the customers and businesses from this. It will try to protect the user's money and encourage more users towards a cashless economy.

3.2 PRODUCT SCOPE

The scope of this project is very diverse, it can be used by different public sector banks, corporate banks, e-commerce companies, and any organization which has any type of payment involved by the customers.

3.3 PRODUCT PERSPECTIVE

This is a new self-contained product. However, our product is in Beta stage and further, it may be improved but for that more research work is needed.

3.4 PRODUCT PERSPECTIVE

This application will use the dataset which is consists of various information in the mixture. This dataset is highly unbalanced. Since providing transaction details of a customer is considered to issue related to confidentiality therefore most of the features in the dataset are transformed using principal component analysis (PCA). V1, V2, V3,..., V28 are PCA applied features and rest i.e., time, amount and class are non-PCA applied features

3.5 OPERATING ENVIRONMENT

3.5.1 Hardware Requirements

Devices	Description
RAM	4GB or more
Processor	Intel Core Duo 2.0 GHz or more
Hard Disk	100 GB or more

3.5.2 Software Requirements

Purpose	Software
Operating System	Windows 8/10,Linux, Mac OS x
Front End	Jupyter notebook
Back End	Numpy,Panda,matplotlib,seaborn,sklearn
Scripting Language	Python

3.6 DESIGN AND IMPLEMENTATION CONSTRAINTS

3.6.1 Hardware Limitations

The major hardware limitations faced by the system are as follows:
If the appropriate hardware is not there like processor, RAM, hard disks
-the problem in processing as it is timetaking
-if appropriate storage is not there our whole database will crash due to less storage because our main requirement is large storage.

3.6.2 Reliability Constraints

The major reliability constraints are as follows:

- The software should be efficiently designed so as to give reliable recognition of fraud transaction and so that it can be used for more pragmatic purpose.
- The design should be versatile and user friendly.
- The application should be fast and reliable.
- The system be compatible with future upgradation.

3.7 FUNCTIONAL REQUIREMENTS

1. In this system there is a lot of internal functioning.
2. User can check whether a transaction is legitimate or fraud
3. User can flag a fraud transaction and cancel the transaction
4.User can check the output as many times as per his requirement

3.8 NON-FUNCTIONAL REQUIREMENTS

3.8.1 Performance Requirements

- The development of the software will be based on the object oriented model.
- The timeline of this software must be in our mind.
- The performance of the functions must me good.
- The risk factor must be taken at initial step for better performance of the software.
- For individual function the performance will be well.
- There will be various ways of retrieving data and it takes less time..
- The overall performance of the model will reliable and enable the users to work Efficiently

3.8.2 Security Requirements

The whole software is secure from the outside accessing.

3.10 SOFTWARE QUALITY ATTRIBUTES

Availability- The availability of the software is easy and for everyone.

Correctness- The results of the function are approximately accurate.

Flexibility- The operation may be flexible and reports can be presented in many ways.

Maintainability- After the deployment of the project if any error occurs then it can be easily maintain by the software developer.

Reliability-The performance of the software is better which will increase the reliability of the software.

Reusability-The data and record that are predicted in the application can be reused if needed.

Usability-To performs any operations and to understand the functioning of software is very easy.

Productivity-This software will produce every desired result with accurately.

Timelines-The time limit is very important. It will provide fast accessing.

Cost effective-This software is less in cost and bearable by any organization.

Requirement Analysis

The entire project depends on various libraries of python.
The libraries are as follows:

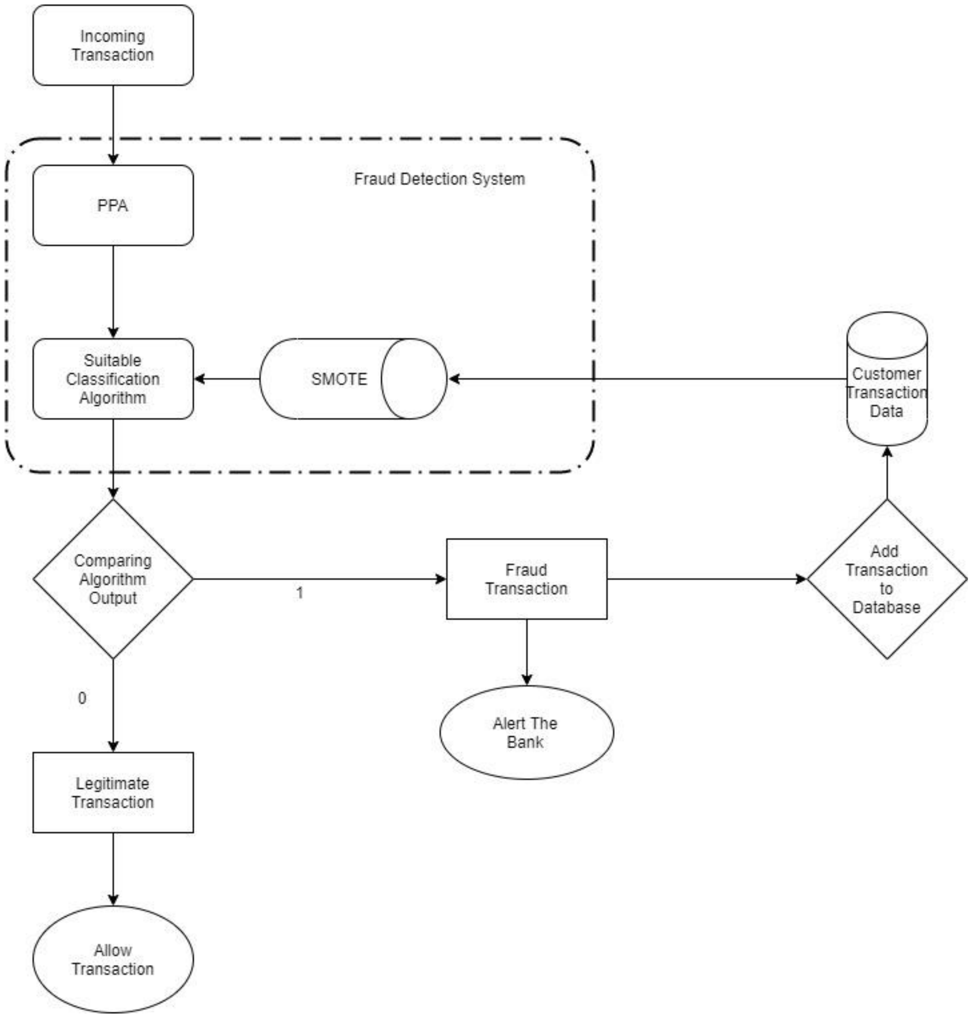
NumPy: NumPy is the fundamental package for scientific computing with Python. It contains among other things:
- a powerful N-dimensional array object
- sophisticated (broadcasting) functions
- tools for integrating C/C++ and Fortran code
- useful linear algebra, Fourier transform, and random number capabilities.

Pandas: pandas is an open source, BSD-licensed library providing high-performance, easy-touse data structures and data analysis tools for the python programming language. pandas is a NumFOCUS sponsored project. This will help ensure the success of development of pandas as a world-class open-source project, and makes it possible to donate to the project.

Python: This module implements a number of iterator building blocks inspired by constructs from APL, Haskell and SML. Each has been recast in a form suitable for Python.

Scikit: Simple and efficient tools for data mining and data analysis. Accessible to everybody, and reusable in various contexts. Built on NumPy, SciPy, and matplotlib. Open source, commercially usable-BSD license

System Design



The above flowchart shows our system design. The customer will do transactions by giving their card details these details will be converted by principal component analysis to protect the users derails. Now as we know that most of the transaction is legitimate so there are very fewer fraud transactions due to this we first apply synthetic minority oversampling in our data set which will remove biasing in our prediction. After applying SMOTE we applied the machine-learning algorithm to detect the validity of the transaction. If our model finds that the transaction is legitimate it will allow the transaction. Id founds that the transaction is fraud it will alert the bank so that transaction can be stopped and customer is protected from any type of fraud. After this, that transaction details will be saved to the data set so that our model can learn from more data which will make our model more effective and useful.

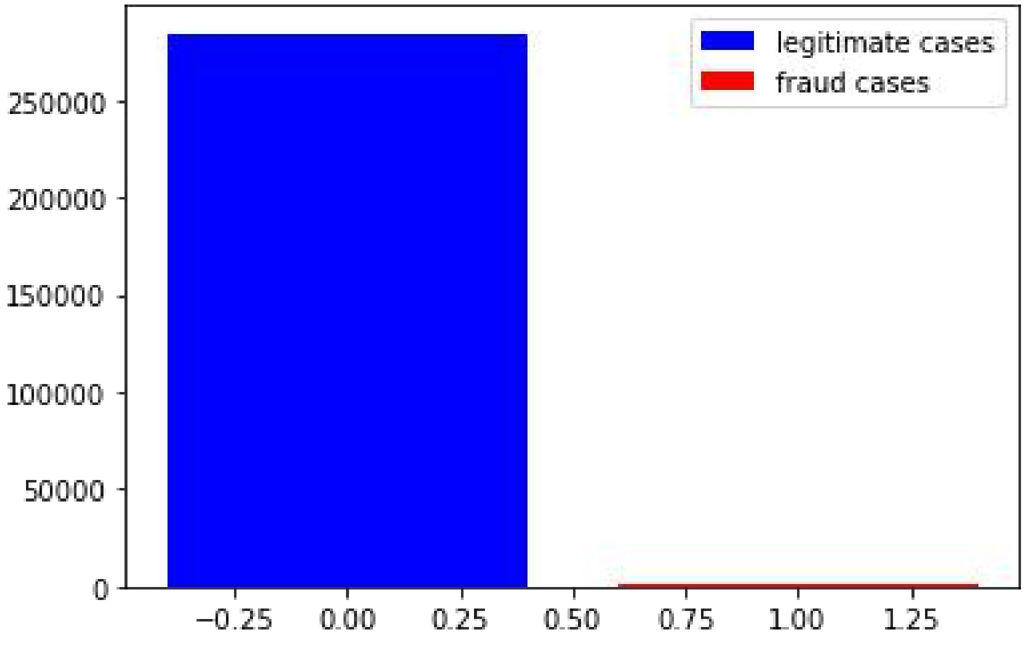
Project Planning

The data set which we are using for this project consist of 284,807 rows and 31 columns. There are total 284,807 transactions

S No	Feature	Description
1	Time	Time of Transaction
2	Amount	Transaction Amount
3	Class	0 - Not Fraud 1 -Fraud
4	PCA applied columns	Encrypted data to protect identity

Most of the features in data set is transformed using Principal Component Analysis(PCA) V1,V2,V3……….V28. This is done to keep the transaction details Of the customers confidential.

Below is the graph which shows the amount of valid transaction and fraud transaction present in our data set



From the bar graph, we can see that fraud transactions are in the minority as compare to valid transactions. There are around 0.172% of fraud transaction that leads to 492 cases.

As fraud transactions are very less so if we apply our machine learning algorithm directly it can give biased results.So to protect our model from biasing we did oversample of minority data in our data set. This can be achieved by simply duplicating examples from the minority class in the training dataset prior to fitting a model. This can balance the class distribution but does not provide any additional information to the model. We used Synthetic Minority Oversampling Technique (SMOTE) for this.

SMOTE first selects a minority class instance at random and finds its k nearest minority class neighbors. The synthetic instance is then created by choosing one of the k nearest neighbors b at random and connecting a and b to form a line segment in the feature space. The synthetic instances are generated as a convex combination of the two chosen instances a and b.

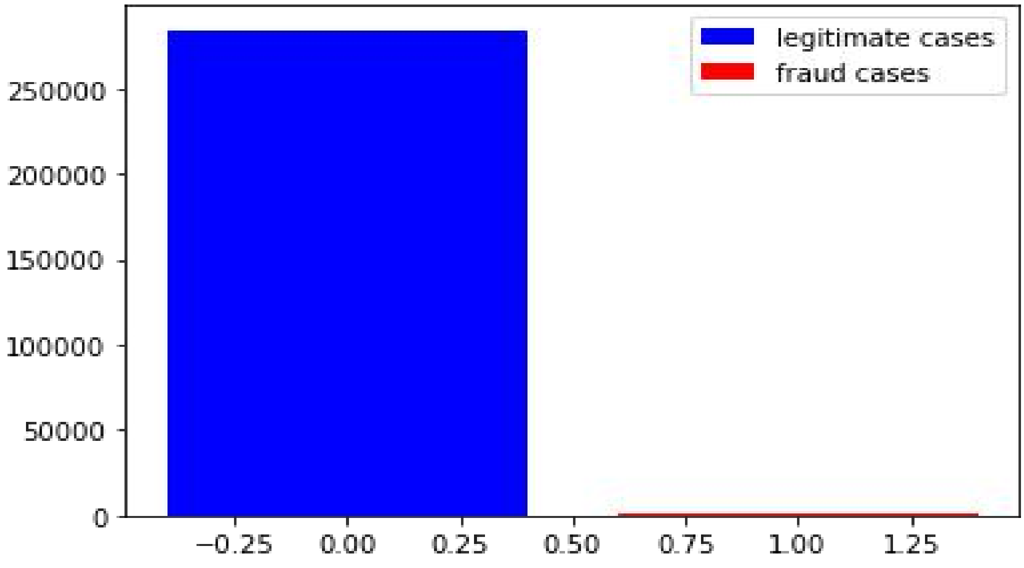
After this, we then applied different machine learning algorithm in our model such logistic regression, decision tree, and random forest to detect the fraud transaction. If the said transaction is valid it is accepted and if it is a fraud the transaction is stopped. We also save the said data and use it for machine training so that our model is in continuous process of learning which makes it more robust, useful, and as it learns from previous data its capability to detect fraud transaction in future increases.

Implementation

We have written our code in python language and used multiple library functions. Firstly we did oversample in our data set so that bias can be removed and then We applied multiple machine learning algorithms to get the desired outcome.

Our Code

```
import pandas as pd
#importing data set
dataset = pd.read_csv('D:\creditcard\card-credit.csv')
import matplotlib.pyplot as plt
legitimate_cases = []
fraud_cases = []
for i in dataset['Class']:
if(i == 0):
      legitimate_cases.append(i)
else:
      fraud_cases.append(i)


#comparision between number of legitimate cases and number of fraud cases
plt.bar([0],[len(legitimate_cases)],color = 'blue',label = 'legitimate cases')
plt.bar([1],[len(fraud_cases)],color = 'red' ,label = 'fraud cases')
plt.legend()
```

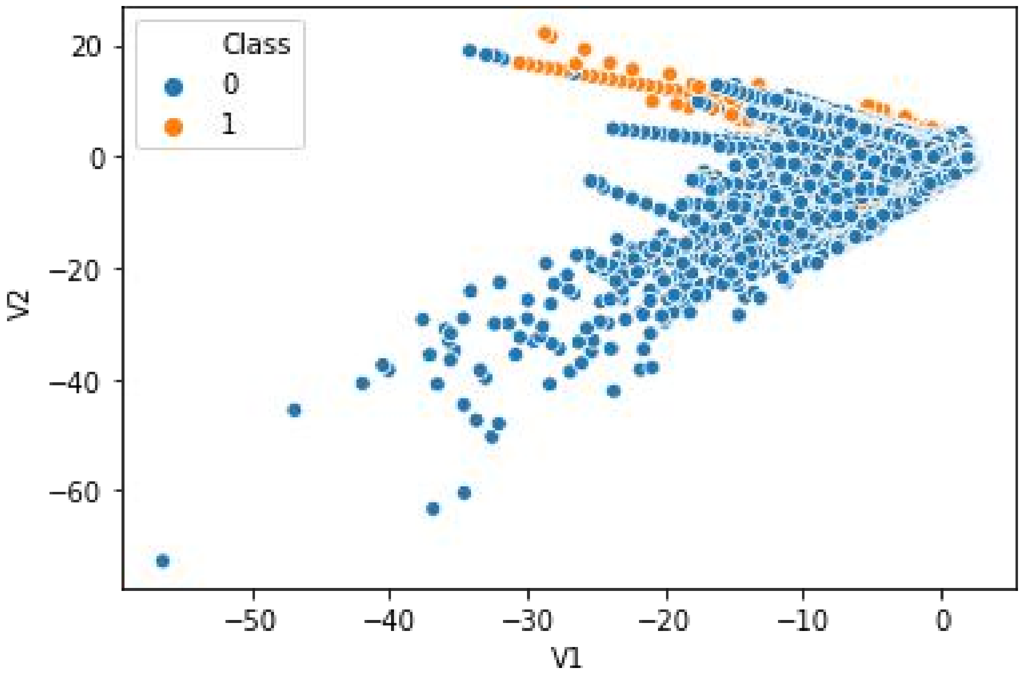


```
round(len(fraud_cases)/(len(fraud_cases) + len(legitimate_cases)) * 100,4)
0. 1727
#only 0.1727 data is fraud
#to tackle this problem we use SMOTE method
```

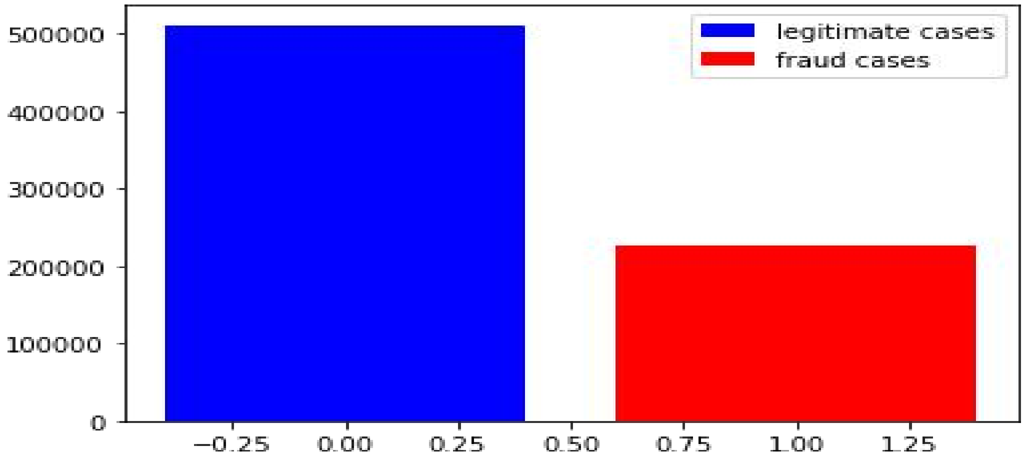
```
dataset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 284807 entries, 0 to 284806
Data columns (total 31 columns):
 #   Column  Non-Null Count   Dtype  
---  ------  --------------   -----  
 0   Time    284807 non-null  float64
 1   V1      284807 non-null  float64
 2   V2      284807 non-null  float64
 3   V3      284807 non-null  float64
 4   V4      284807 non-null  float64
 5   V5      284807 non-null  float64
 6   V6      284807 non-null  float64
 7   V7      284807 non-null  float64
 8   V8      284807 non-null  float64
 9   V9      284807 non-null  float64
 10  V10     284807 non-null  float64
 11  V11     284807 non-null  float64
 12  V12     284807 non-null  float64
 13  V13     284807 non-null  float64
 14  V14     284807 non-null  float64
 15  V15     284807 non-null  float64
 16  V16     284807 non-null  float64
 17  V17     284807 non-null  float64
 18  V18     284807 non-null  float64
 19  V19     284807 non-null  float64
 20  V20     284807 non-null  float64
 21  V21     284807 non-null  float64
 22  V22     284807 non-null  float64
 23  V23     284807 non-null  float64
 24  V24     284807 non-null  float64
 25  V25     284807 non-null  float64
 26  V26     284807 non-null  float64
 27  V27     284807 non-null  float64
 28  V28     284807 non-null  float64
 29  Amount  284807 non-null  float64
 30  Class   284807 non-null  int64  
dtypes: float64(30), int64(1)
memory usage: 67.4 MB
```

```
# hence there is no missing value in the dataset

import seaborn as sns
x = dataset['V1']
y = dataset['V2']
sns.scatterplot(x="V1", y="V2", hue="Class",data=dataset)
```



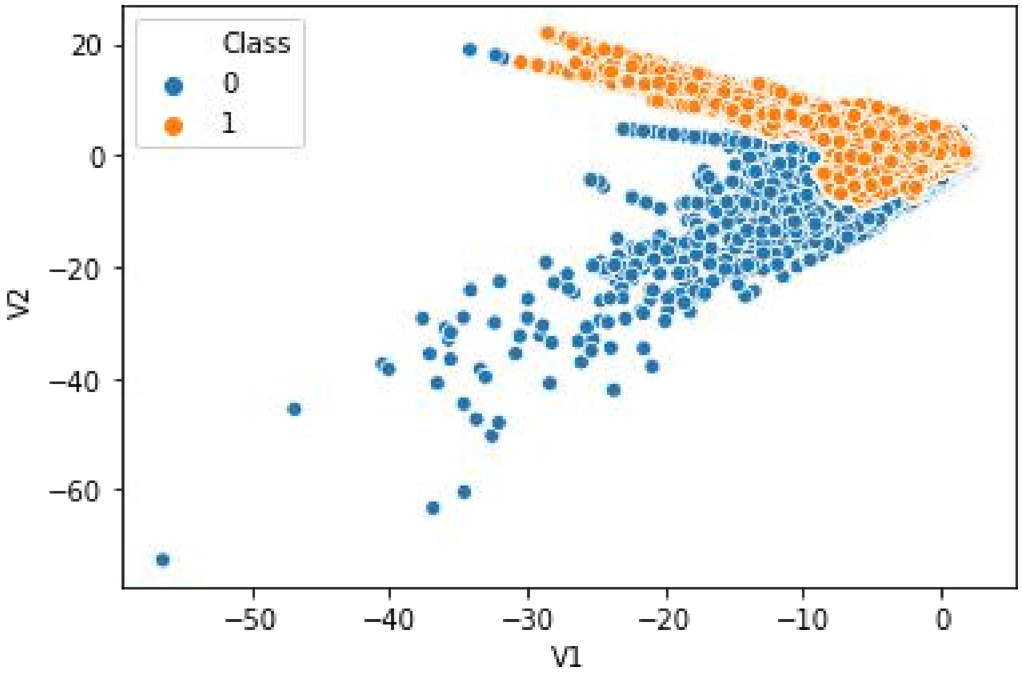
```
from sklearn.model_selection import train_test_split
X = dataset.iloc[ : ,  :-1]
y = dataset.iloc[ : , -1]
X_train,X_test,y_train,y_test = train_test_split(X,y,test_size = 0.2)
from imblearn.over_sampling import SMOTE
sm = SMOTE(random_state = 2)
X_train_res, y_train_res = sm.fit_sample(X_train, y_train.ravel())
for i in y_train_res:
if(i == 0):
        legitimate_cases.append(i)
else:
        fraud_cases.append(i)
plt.bar([0],[len(legitimate_cases)],color = 'blue',label = 'legitimate cases')
plt.bar([1],[len(fraud_cases)],color = 'red' ,label = 'fraud cases')
plt.legend()
```



```
y_train_res
array([0, 0, 0, ..., 1, 1, 1])
(X_train_res)
```

```
array([[ 7.96930000e+04, -2.24136492e+00,  2.32437654e+00, ...,
        -5.17454863e-01, -1.65388613e+00,  9.81000000e+00],
       [ 1.28564000e+05, -2.19311737e+00,  1.97071302e+00, ...,
         7.48508919e-01,  4.08926894e-01,  9.77000000e+00],
       [ 1.50133000e+05,  1.83704802e+00, -6.89038123e-01, ...,
        -1.86541540e-01, -7.30530124e-02,  1.88200000e+02],
       ...,
       [ 9.39330928e+04, -1.08830325e+01,  7.16294446e+00, ...,
        -1.13404481e+00, -2.55145772e-01,  6.52007810e+00],
       [ 1.20075806e+05, -4.06020575e+00,  3.58031147e+00, ...,
         1.63274894e-01,  6.67149573e-01,  1.27985932e+02],
       [ 5.04245130e+04, -5.17972680e+00,  4.03093717e+00, ...,
         6.53958015e-01,  9.69032963e-02,  2.62326698e+02]])
```

```
#plotting X_train_res and y_train res to see increase in fraud cases using SMOTE
xt = X_train_res.tolist()
yt = y_train_res.tolist()
for i in range(0, len(xt)) :
  xt[i].append(yt[i])

#creating dataframe from 2d numpy array
df = pd.DataFrame(xt)
df.columns=["Time","V1","V2","V3","V4","V5","V6","V7","V8","V9","V10","V11
","V12","V13","V14","V15","V16","V17","V18","V19","V20","V21","V22","V23",
"V24","V25","V26","V27","V28","Amount","Class"]
sns.scatterplot(x="V1", y="V2", hue="Class",data=df)
```



```
p = len(fraud_cases)/(len(legitimate_cases) + len(fraud_cases)) * 100
P
30.816055889831713
#we have around 30% of fraud cases
#implementing Logistic Regression
```

```
import statsmodels.api as stats
x=stats.add_constant(X_train_res)
reg_log = stats.Logit(y_train_res,x)
results_temp_log = reg_log.fit()
#iterations is less than 35
Optimization terminated successfully.
         Current function value: 0.055786
         Iterations 15
results_temp_log.summary()
# the LLR value in summary table is 0.000 hence the model is significa
```

```
                      Logistic Regression Results
  Dep. Variable:   Class              No. Observations: 454888
      Model:       Logit                Df Residuals:   454857
     Method:       MLE                    Df Model:     30
      Date:        Mon, 25 May 2020   Pseudo R-squ.:    0.9195
      Time:        20:02:51            Log-Likelihood:  -25376.
   converged:      True                   LL-Null:      -3.1530e+05
Covariance Type:   nonrobust           LLR p-value:     0.00
```

```
from sklearn.linear_model import LogisticRegression
LR=LogisticRegression(max_iter=400)
LR.fit(X_train_res,y_train_res.values.ravel())
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
                   intercept_scaling=1, l1_ratio=None, max_iter=400,
                   multi_class='auto', n_jobs=None, penalty='l2',
                   random_state=None, solver='lbfgs', tol=0.0001, verbose=0,
                   warm_start=False)

y_pred_LR = LR.predict(X_test)
```

```
from sklearn.metrics import confusion_matrix
results_LR = confusion_matrix(y_test,y_pred_LR)
print(results_LR)
```

```
[[55865  1006]
 [   13    78]]
```

```
import sklearn.metrics as metrics
score_LR = metrics.accuracy_score(y_test, y_pred_LR)
score_LR
```

```
0.9821108809381693
```

```
non_common = results_LR[0][1]+results_LR[1][0]
overall = y_test.shape[0]

print('Missclassification rate:'+str(non_common/overall))
```

```
Missclassification rate:0.017889119061830695
```

```
#Implementing Decision Tree
from sklearn import tree
clf = tree.DecisionTreeClassifier()
clf = clf.fit(X_train_res,y_train_res)
fig, ax = plt.subplots(figsize=(10, 10))
tree.plot_tree(clf.fit(X_train_res, y_train_res), ax=ax)
y_pred = clf.predict(X_test)
from sklearn.metrics import confusion_matrix
results = confusion_matrix(y_test,y_pred)
print(results)
```

```
[[56785    86]
 [   23    68]]
```

```
y_test.shape
```

```
(56962,)
```

```
import sklearn.metrics as metrics
score = metrics.accuracy_score(y_test, y_pred)
Score
```

```
0.9980864435939749
```

```
non_common_DT = results[0][1]+results[1][0]
overall_DT = y_test.shape[0]
```

```
print('Missclassification rate:'+str(non_common_DT/overall_DT))
```

```
Missclassification rate:0.0019135564060250693
```

```
#Applying Random Forest Classification

from sklearn.ensemble import RandomForestClassifier
Random_Class=RandomForestClassifier(n_estimators=150)
Random_Class.fit(X_train_res,y_train_res.values.ravel())
```

```
RandomForestClassifier(bootstrap=True, ccp_alpha=0.0, class_weight=None,
                       criterion=’gini’, max_depth=None, max_features=’auto’,
                       max_leaf_nodes=None, max_samples=None,
                       min_impurity_decrease=0.0, min_impurity_split=None,
                       min_samples_leaf=1, min_samples_split=2,
                       min_weight_fraction_leaf=0.0, n_estimators=150,
                       n_jobs=None, oob_score=False, random_state=None,
                       verbose=0, warm_start=False)
```

```
y_pred_Random = Random_Class.predict(X_test)
results_Random = confusion_matrix(y_test,y_pred_Random)
print(results_Random)
```

```
[[56863     8]
 [   25    66]]
```

```
score_Random = metrics.accuracy_score(y_test, y_pred_Random)
score_Random
```

```
0.9994206664091185
```

```
non_common_Random = results_Random[0][1]+results_Random[1][0]
overall_Random = y_test.shape[0]

print('Missclassification rate:'+str(non_common_Random/overall_Random))
```

```
Missclassification rate:0.0005793335908149292
```

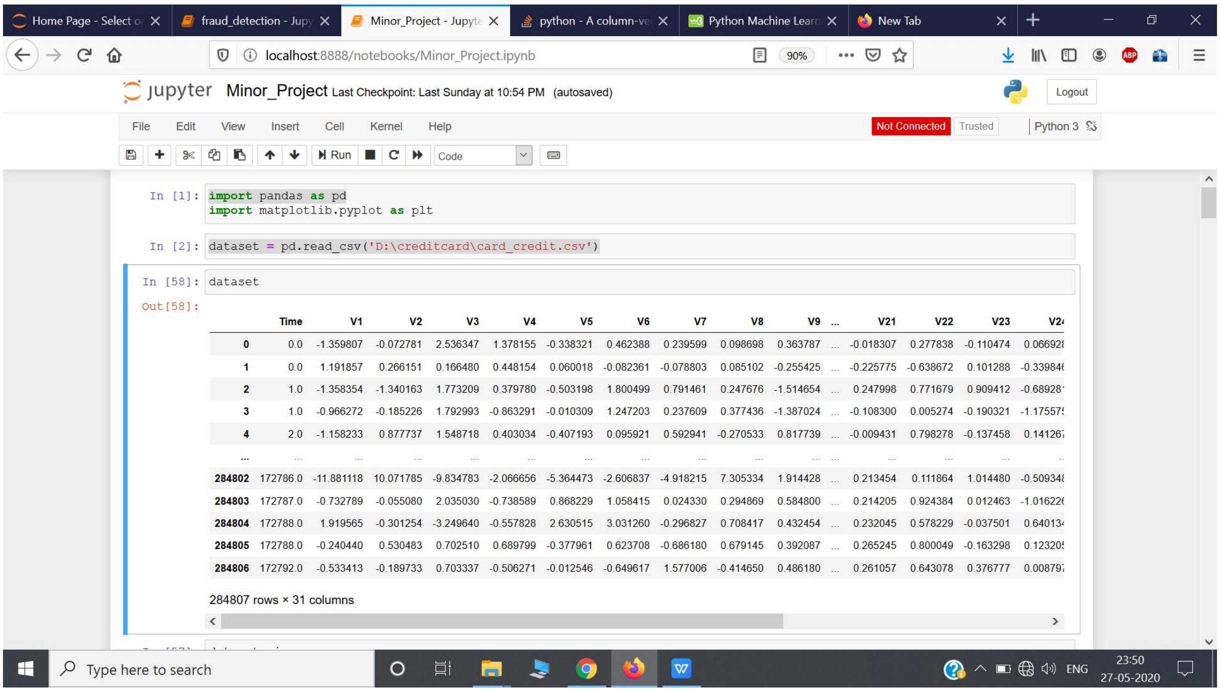
System Testing

6.1 Test Cases and Test Results

We have applied three machine learning algorithms mainly Logistic Regression,Decision Tree and Random Forest Classifier.
From the table we can clearly see that the random forest classifier detect the fraud credit card transaction more easily and accurately as compared to logistic regression and Decision tree

Test No	Algorithm	Accuracy score	Missclassification Rate	Confusion Matrix Result
T01	Logistic Regression	0. 9668550963800429	0. 017889119061830695	[[55865 1006] [13 78]
T02	Decision Tree	0. 9980864435939749	0. 0019135564060250693	[[56785 86] [23 68]]
T03	Random Forest	0. 999420666409185	0. 0005793335908149292	[[56863 8] [25 66]]

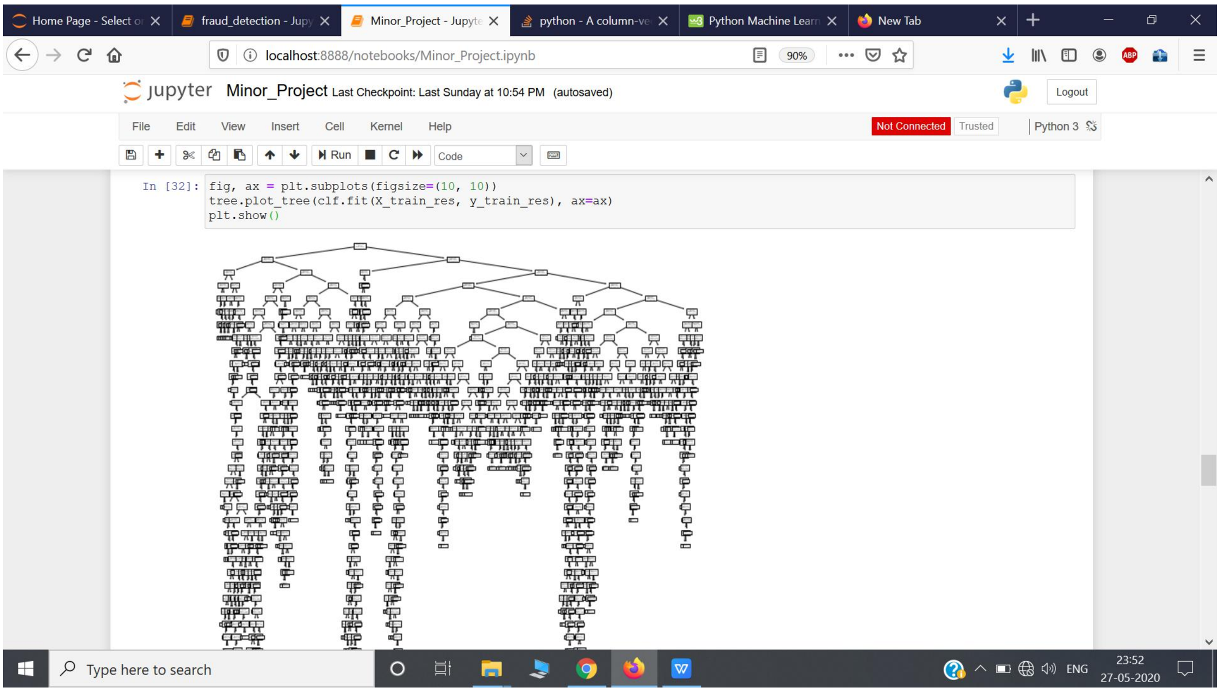
Screen shots of Project



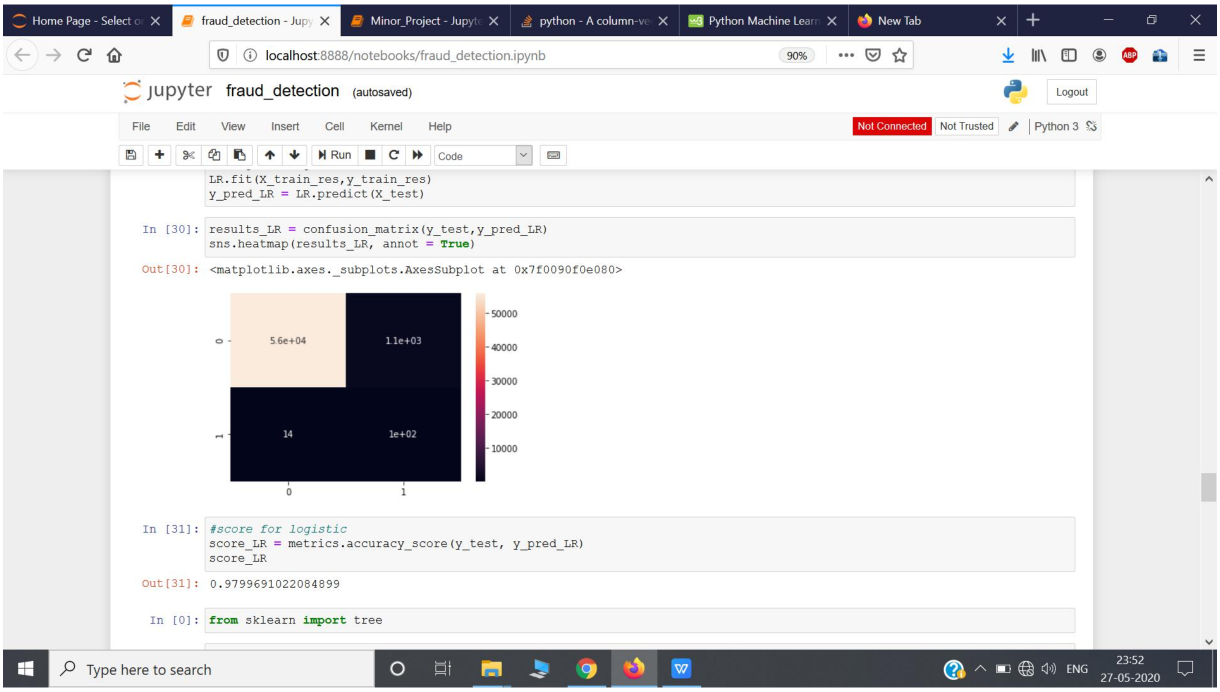
```
In [1]: import pandas as pd
        import matplotlib.pyplot as plt

In [2]: dataset = pd.read_csv('D:\creditcard\card_credit.csv')

In [58]: dataset
Out[58]:
```

	Time	V1	V2	V3	V4	V5	V6	V7	V8	V9	...	V21	V22	V23	V2
0	0.0	-1.359807	-0.072781	2.536347	1.378155	-0.338321	0.462388	0.239599	0.098698	0.363787	...	-0.018307	0.277838	-0.110474	0.06692
1	0.0	1.191857	0.266151	0.166480	0.448154	0.060018	-0.082361	-0.078803	0.085102	-0.255425	...	-0.225775	-0.638672	0.101288	-0.33984
2	1.0	-1.358354	-1.340163	1.773209	0.379780	-0.503198	1.800499	0.791461	0.247676	-1.514654	...	0.247998	0.771679	0.909412	-0.68928
3	1.0	-0.966272	-0.185226	1.792993	-0.863291	-0.010309	1.247203	0.237609	0.377436	-1.387024	...	-0.108300	0.005274	-0.190321	-1.17557
4	2.0	-1.158233	0.877737	1.548718	0.403034	-0.407193	0.095921	0.592941	-0.270533	0.817739	...	-0.009431	0.798278	-0.137458	0.14126
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	..
284802	172786.0	-11.881118	10.071785	-9.834783	-2.066656	-5.364473	-2.606837	-4.918215	7.305334	1.914428	...	0.213454	0.111864	1.014480	-0.50934
284803	172787.0	-0.732789	-0.055080	2.035030	-0.738589	0.868229	1.058415	0.024330	0.294869	0.584800	...	0.214205	0.924384	0.012463	-1.01622
284804	172788.0	1.919565	-0.301254	-3.249640	-0.557828	2.630515	3.031260	-0.296827	0.708417	0.432454	...	0.232045	0.578229	-0.037501	0.64013
284805	172788.0	-0.240440	0.530483	0.702510	0.689799	-0.377961	0.623708	-0.686180	0.679145	0.392087	...	0.265245	0.800049	-0.163298	0.12320
284806	172792.0	-0.533413	-0.189733	0.703337	-0.506271	-0.012546	-0.649617	1.577006	-0.414650	0.486180	...	0.261057	0.643078	0.376777	0.00879

284807 rows × 31 columns

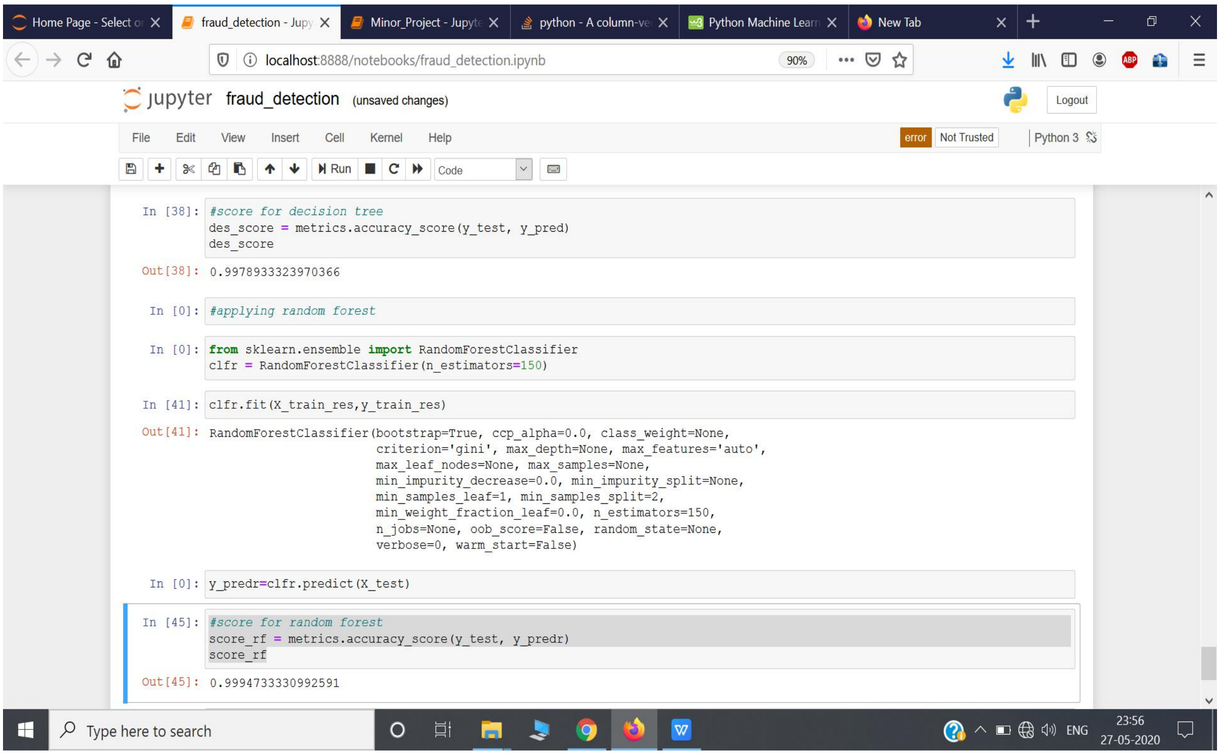


```
In [32]: fig, ax = plt.subplots(figsize=(10, 10))
         tree.plot_tree(clf.fit(X_train_res, y_train_res), ax=ax)
         plt.show()
```



```
        LR.fit(X_train_res,y_train_res)
        y_pred_LR = LR.predict(X_test)

In [30]: results_LR = confusion_matrix(y_test,y_pred_LR)
         sns.heatmap(results_LR, annot = True)

Out[30]: <matplotlib.axes._subplots.AxesSubplot at 0x7f0090f0e080>
```

```
In [31]: #score for logistic
         score_LR = metrics.accuracy_score(y_test, y_pred_LR)
         score_LR

Out[31]: 0.9799691022084899

In [0]: from sklearn import tree
```



```
In [38]: #score for decision tree
         des_score = metrics.accuracy_score(y_test, y_pred)
         des_score

Out[38]: 0.9978933323970366

In [0]: #applying random forest

In [0]: from sklearn.ensemble import RandomForestClassifier
        clfr = RandomForestClassifier(n_estimators=150)

In [41]: clfr.fit(X_train_res,y_train_res)

Out[41]: RandomForestClassifier(bootstrap=True, ccp_alpha=0.0, class_weight=None,
                                criterion='gini', max_depth=None, max_features='auto',
                                max_leaf_nodes=None, max_samples=None,
                                min_impurity_decrease=0.0, min_impurity_split=None,
                                min_samples_leaf=1, min_samples_split=2,
                                min_weight_fraction_leaf=0.0, n_estimators=150,
                                n_jobs=None, oob_score=False, random_state=None,
                                verbose=0, warm_start=False)

In [0]: y_predr=clfr.predict(X_test)

In [45]: #score for random forest
         score_rf = metrics.accuracy_score(y_test, y_predr)
         score_rf

Out[45]: 0.9994733330992591
```

Conclusion and Future Scope

10.1 Conclusion

Credit card fraud is without a doubt an act of criminal dishonesty. Through this project, we have tried to protect the customer from this kind of fraud. In this project, we have reached to the accuracy of 99.94% and achieved a misclassification rate of 0.000579

This project has also explained in detail, how machine learning can be applied to get better results in fraud detection along with the algorithm, code, explanation of its implementation, and experimentation results.

10.2 Future Scope

In our future work, we attempt other Machine Learning and Deep Learning algorithms to get better predictions.

As we know the size of data will keep increasing so we will implement this model on technologies like apache-spark so we can handle both an imbalanced dataset and the realtime problem (to have a response during the financial transaction runtime) with improved accuracy.

We will also attempt to make good working UI for better use of the system.

References

[1] https://www.researchgate.net/publication/336800562_Credit_Card_Fraud_Detection_using_Machine_Learning_and_Data_Science

[2] Credit Card Fraud Detection using Machine Learning Algorithms Vaishnavi Nath Dornadulaa , Geetha Sa

[3] https://www.kaggle.com/mlg-ulb/creditcardfraud

[4] https://shiftprocessing.com/credit-card-fraud-statistics/

[5] https://machinelearningmastery.com/smote-oversampling-for-imbalanced-classification/

[6] Sam Maes, Karl Tuyls, Bram Vanschoenwinkel, Bernard Manderick. Credit Card Fraud Detection Using Bayesian and Neural Networks First International NAISO Congress on Neuro Fuzzy Technologies, Havana, Cuba. 2002.

[7] M.J. Kim and T.S. Kim, “A Neural Classifier with Fraud Density Map for Effective Credit Card Fraud Detection,” Proc. International Conference on Intelligent Data Engineering and Automated Learning, Lecture Notes in Computer Science, Springer Verlag, no. 2412, pp. 378-383, 2002.

[8] Dr.R.Dhanapal, “An intelligent information retrieval agent”, Elsevier International Journal on Knowledge Based Systems 2008

[9] Jaakko Hollmén, “Novelty filter for fraud detection in mobile communications networks”. Technical Report A48, Helsinki University of Technology, Laboratory of Computer and Information Science, October 1997

CREDIT CARD FRAUD DETECTION

ANUPAM KUMAR ANIKET
1705023

Abstract: This project intends to illustrate the modelling of a data set using machine learning with Credit Card Fraud Detection. The Credit Card Fraud Detection Problem includes modelling past credit card transactions with the data of the ones that turned out to be fraud. This model is then used to recognize whether a new transaction is fraudulent or not. Our objective here is to detect the fraudulent transactions while minimizing the incorrect fraud classifications.

Individual contribution and findings: Individual contribution and findings: My job in the completion of this project was to make the dataset fit for applying classification algorithms on it. First I checked for any missing values in the dataset and as there were none of it the dataset was good to go. Next, I checked for outliers and found that data were highly biased towards legitimate transactions than fraud ones as there were very few fraud transactions as compared to legitimate transactions. Numerically only 0.172% of data which were of fraud transaction. Machine Learning algorithms tend to produce unsatisfactory classifiers when faced with imbalanced datasets. For any imbalanced data set, if the event to be predicted belongs to the minority class and the event rate is less than 5%, it is usually referred to as a rare event. So I decided to go for the SMOTE(Synthetic Minority Over-sampling Technique) for an imbalanced dataset. This technique is followed to avoid overfitting which occurs when exact replicas of minority instances are added to the main dataset. A subset of data is taken from the minority class as an example and then new synthetic similar instances are created. These synthetic instances are then added to the original dataset. The new dataset is used as a sample to train the classification models. On applying SMOTE the number of fraud transactions was around 30% of the whole dataset. Now dataset set was good to predict results on it.

Individual contribution to project report preparation: I contributed in the preparation of the project report by making suitable system design and diagram for the fraud detection model. I have also written the future scope for the project and finally, in the implementation part, I wrote about my works in the project that is data cleaning and oversampling.

Full Signature of Supervisor: …………………………… Full signature of the student: ……………………………..

CREDIT CARD FRAUD DETECTION

ADITYA
1705008

Abstract: This project intends to illustrate the modelling of a data set using machine learning with Credit Card Fraud Detection. The Credit Card Fraud Detection Problem includes modelling past credit card transactions with the data of the ones that turned out to be fraud. This model is then used to recognize whether a new transaction is fraudulent or not. Our objective here is to detect the fraudulent transactions while minimizing the incorrect fraud classifications.

Individual contribution and findings: For the completion this project my contribution was to implement a model which suits well for the prediction. Since the objective of the problem statement was to predict the outcome of a dependent variable class which boils down the possible outcomes to a Yes(1) and No(0) situation. As Logistic Regression deals well with yes and no outcome I implemented Logistic Regression on the processed data set on the basis of logistic equation:-

$$\ell = \log_b \frac{p}{1-p} = \beta_0 + \beta_1 x_1 + \beta_2 x_2$$

Regression summary table gave more insights on the significance of the model. Maximum likelihood estimation technique is used which shows how likely is the model describes the underlying relationship of the independent variables. The LogLikelihoodRatio(LLR p-value) has value ~0.00 which shows the model is significant for the dataset. The p-value for every independent variable is close to 0.00 which shows significance of each independent variable and also implies No Endogeneity . The accuracy score for the test dataset came out to be near 0.986 but can change a little as random state for the train test split is set to 2. From the confusion metrics, the Missclassification rate came out to be 0.013 .

Individual contribution to project report preparation: In this project report I gave my contribution in writing the literature survey which was about the previous researches done in the field of fraud detection which gave us insights on the challenges of the topic. I also wrote the code of implementation of logistic regression and its findings in project report and finally I wrote the conclusion of the project.

Full Signature of Supervisor: ……………………………

Full signature of the student: …………………………..

CREDIT CARD FRAUD DETECTION

MANISH KUMAR MATHUR
1705046

Abstract: This project intends to illustrate the modelling of a data set using machine learning with Credit Card Fraud Detection. The Credit Card Fraud Detection Problem includes modelling past credit card transactions with the data of the ones that turned out to be fraud. This model is then used to recognize whether a new transaction is fraudulent or not. Our objective here is to detect the fraudulent transactions while minimizing the incorrect fraud classifications.

Individual contribution and findings: In this project I have firstly applied two machine learning algorithm mainly Decision Tree algorithm and Random forest Algorithm to predict whether a transaction is legitimate or fraud.The first algorithm which I applied is decision tree.I have applied this because we are working with classification problem and decision tree works well with it.Working of algorithm is first it select the target attribute and then we calculate the information gain of the target attribute after that we calculate the entropy of each feature in our data set after getting these two we the finally calculate gain for all attributes by subtracting information gain by entropy,the highest gain will be selected as root node ad split is done.By applying decision tree we got a accuracy of 0.9980864435939749 and a misclassification rate of 0.0019135564060250693 . The only problem with decision tree is overfitting.So to remove over fitting I have applied another machine learning algorithm known as Random Forest Classifier.It is a type of ensemble learning method in which we combine multiple decision tress to predict the result.For this project I have used a total of 150 trees by keeping the value of n_estimator equal to 150. After applying Random Forest Classifier we achieved accuracy of 0.999420666409185 And a missclassification rate of 0.0005793335908149292.

Individual contribution to project report preparation:
In this project report I gave my contribution in writing the introduction by collecting information from different sources which are related to our project.I attached different graphs for better visualization. I also wrote the project planning by mentioning different stages which are present in our project by giving sone brief explanation.I also contributed in preparation of software requirement specification.and finally in implementation part I wrote my part of code which includes random forest and decision tree.

Full Signature of Supervisor: ……………………………

Full signature of the student: …………………………..