IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588

International Advanced Research Journal in Science, Engineering and Technology

Impact Factor 7.105 ⁜ Vol. 9, Issue 4, April 2022

DOI: 10.17148/IARJSET.2022.9462

Smart Credit Card Fraud Detection Approach Using Supervised Machine Learning Techniques

Sabugar Mohmadfurkan[1] , Asst. Prof Deep Joshi[2]

Student of M.E, Computer Engineering Dept, Grow more Faculty of Engineering, Himmatnagar, Gujarat, India[1]

Grow More Faculty of Engineering, Udaipur – Himmatnagar Highway, Himmatnagar, Gujarat, India[2]

Abstract: Due to the surge of intrigued in online retailing, the utilize of credit cards has been quickly extended in later a long time. Taking the card details to perform online exchanges, which is called extortion, has moreover seen more habitually. Preventive arrangements and moment extortion location methods are broadly considered due to basic monetary misfortunes in numerous industries. In this work, Naïve Bayes, D-TREE,RANDOM FOREST and PBTC(POWER BOOSTING TREE CLASSIFIER) Classifier show for the detection of credit card fakes on the spilling transactions is explored with the utilize of diverse qualities of card transactions. I am applying Naïve Bayes, D-TREE and PBTC(POWER BOOSTING TREE CLASSIFIER) Classifier algorithm to detect the CC fraud then compare the result with all algorithms for getting higher CC fraud accuracy.

keywords- Data mining, Credit card fraud, Fraud detection, Naïve Bayes, D-TREE, PBTC(POWER BOOSTING TREE CLASSIFIER) Classifier

1. INTRODUCTION

The credit card payment system is one of the simplest payment methods and the most common type of financial transaction. However, it is observed that a good number of fraudulent credit card transactions are occurring. Credit card fraud means the unauthorized use of a credit card account. Thus, fraud occurs when the third party starts unauthorized use of the credit card without the consent of the card owner.

These cards can be used for making purchases in both online and offline modes. Online credit purchases need customers to endorse payments by showing at the point of sale their personal identity numbers (PINs), while offline transactions need customers to sign purchase receipts. The method of Credit Card Fraud Detection (CCFD) mainly involves separating fraudulent financial details from genuine data. The models help to classify the pattern of fraud in the databases by applying machine learning algorithms. Various problems are associated with credit card fraud detection and hinder the direction of fraud detection, such as non-availability of the real dataset, size of the dataset, determining the appropriate evaluation parameters, and complex actions of the fraudsters.

2. OBJECTIVES

The main objective of this research is shown below,

- To handle highly imbalanced and skewed data.
- Improve the accuracy of the method for detecting credit card fraud.
- Improve the performance of the credit card fraud detection system.
- The detection of a fraudulent transaction has to be efficient and effective.

3. PROBLEM DEFINITION

- With the growth of e-commerce websites, people and financial companies rely on online services to carry out their transactions that have led to an exponential increase in credit card frauds.
- Fraudulent credit card transactions lead to a loss of a huge amount of money. The design of an effective fraud detection system is necessary to reduce the losses incurred by the customers and financial companies.
- Three categories are there.
  - CATE-1 Traditional card related frauds (application, stolen, account takeover, fake and counterfeit),
  - CATE-2 Merchant related frauds (merchant collusion and triangulation) and
  - CATE-3 Internet frauds (site cloning, credit card generators, and false merchant sites).

IARJSET ISSN (O) 2393-8021, ISSN (P) 2394-1588
International Advanced Research Journal in Science, Engineering and Technology
Impact Factor 7.105 ⯊ Vol. 9, Issue 4, April 2022
DOI: 10.17148/IARJSET.2022.9462

4. MODEL USED

4.1 Basic Overview of Naive Bayes

Naïve Bayes algorithm is a supervised learning algorithm. It is a form of probabilistic classifier model. It is a probabilistic classifier model, which means it can make predictions for several classes at the same time. It is based on the Bayes Theorem. Probabilistic Classifiers are those which make it possible to predict multiple classes. The decision is made based on conditional probability.[3] Bayes' Theorem finds the probability of an event occurring given the probability of another event that has already occurred. Bayes' theorem is stated mathematically as the following equation:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

Here,
P (A) => independent probability of A (prior probability)
P (B) => independent probability of B
P (B|A) => conditional probability of B given A (likelihood)
P (A|B) => conditional probability of A given B (posterior probability)

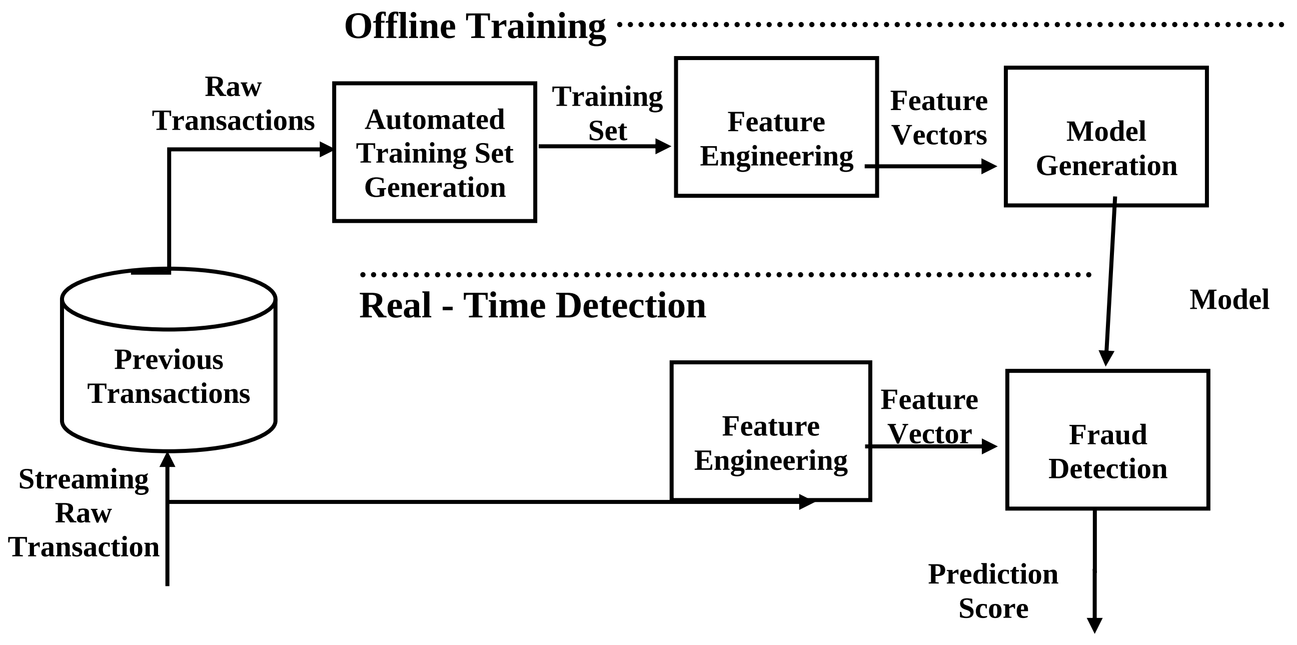
4.2 Basic Overview of Decision Tree

Decision Tree algorithm is a supervised learning algorithm. This is one of the most commonly used approaches to predictive modelling. As the name implies, the model is built in the form of a tree. In the case of a multi-dimensional analysis with multiple groups, this model may be used. The past data (also known as the past vector) is used to construct a model that can predict the output value based on the input. A tree has several nodes, each of which corresponds to one of two vectors. The tree comes to a halt at a leaf node, each of which represents a potential outcome or output. By learning basic decision rules inferred from prior data (training data), a Decision Tree can be used to construct a training model that can be used to predict the target variable's class or value.

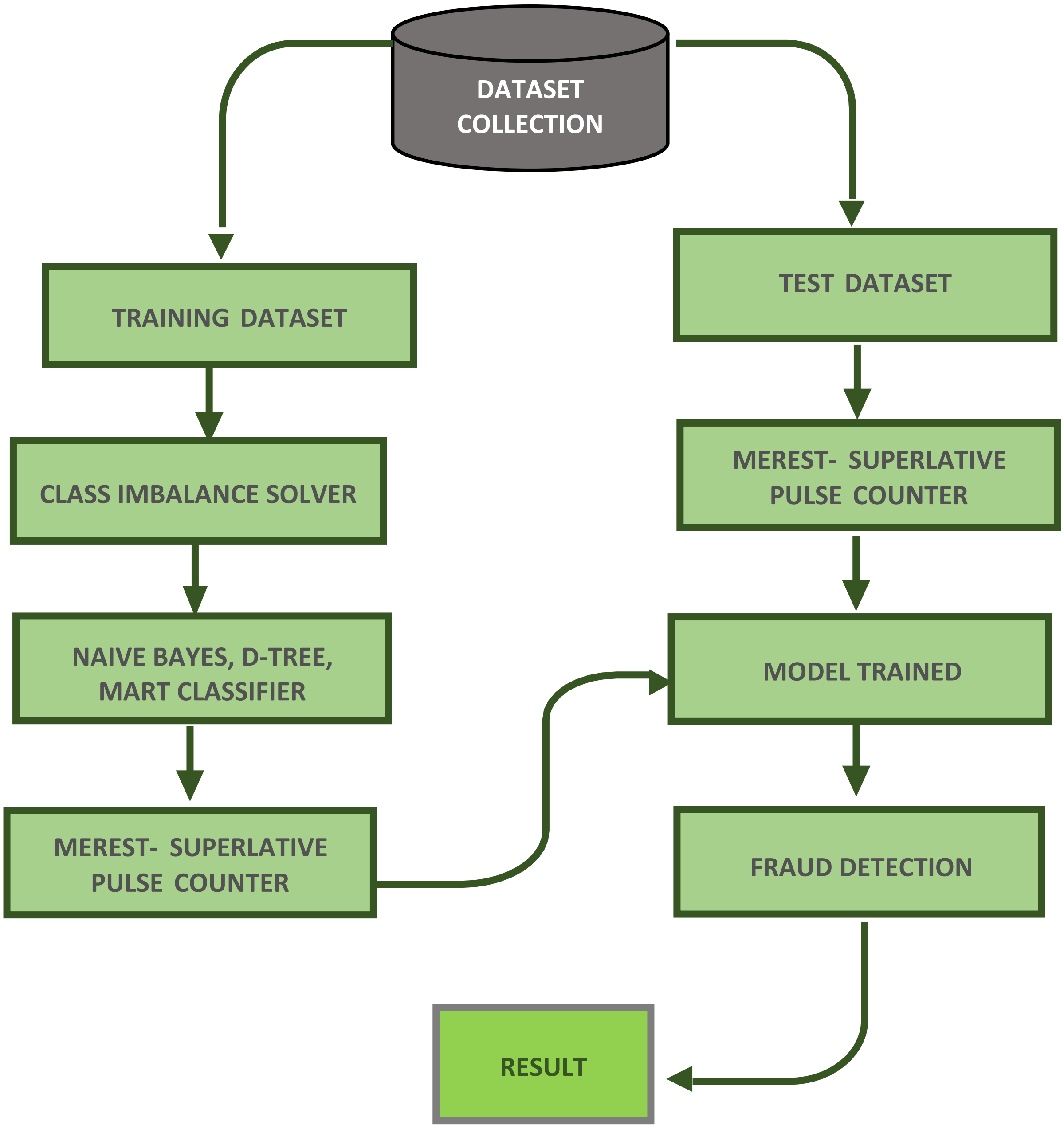
4.3 Basic Overview of PBTC(POWER BOOSTING TREE CLASSIFIER) Classifier

PBTC (PBTC(POWER BOOSTING TREE CLASSIFIER)s) is a predictive data mining implementation of gradient tree boosting methods (regression and classification). PBTC is one of a group of techniques known as boosting. Boosting is a general method for improving the accuracy of any learning algorithm by fitting a collection of low-error models and then combining them to create a high-performing ensemble. PBTC fits a set of very simple classification trees, each requiring just a small amount of computational effort.

5. EXISTING SYSTEM MODEL



5. PROPOSED SYSTEM MODEL


DATASET COLLECTION
TRAINING DATASET
TEST DATASET
CLASS IMBALANCE SOLVER
MEREST- SUPERLATIVE PULSE COUNTER
NAIVE BAYES, D-TREE, MART CLASSIFIER
MODEL TRAINED
MEREST- SUPERLATIVE PULSE COUNTER
FRAUD DETECTION
RESULT

7. DESCRIPTION OF PROPOSED SYSTEM

- It starts with the data collection; here in this step, the collected input data is in the form of CSV files.
- A process to gather context to the input data. Understanding the data for preprocessing and cleaning of datasets. The two columns 'amount' and 'time' were not normalized. The remaining columns were normalized using Principal Component analysis.
- Dataset then divided into a training dataset and test dataset among them 80% of the data will be used for training the model while rest 20% will be used for testing the model, which will be highly skewed or imbalanced.
- Now utilize class imbalance solver technique SMOTE on the dataset, which is used to balance class distribution by randomly increasing minority class examples by replicating them.
- Then apply Merest – Superlative pulse counter on the dataset. It normalizes the data for every feature the minimum value of that feature gets transformed into a 0, the maximum value gets transformed into a 1 and every other value gets transformed into decimal between 0 and 1.

- After the data segregation, the data are fed into machine learning algorithms like Naïve Bayes, D-TREE, and PBTC(POWER BOOSTING TREE CLASSIFIER) (PBTC) Classifier. This step is mainly done using training data to teach the machine to increase its predictive accuracy.
- Once the data have learned enough, our learned model will be ready for testing.
- The learned model is tested using test data to check its predictive accuracy.
- If the predictive accuracy is up to the desired level, then the model is deployed.

8. PROPOSED ALGORITHM

Step 1: Take input from Dataset.
Step 2: Data-preprocessing from Dataset.
Step 3: Divide Training and Testing data from Dataset.
Step 4: Utilize class imbalance solver technique on Dataset
Step 5: Apply Merest – Superlative pulse counter on Dataset
Step 6: Train Model using Naïve Bayes, D-TREE, and PBTC(POWER BOOSTING TREE CLASSIFIER) (PBTC) Classifier algorithm
Step 7: Model Trained
Step 8: Fraud Detection
Step 9: Result

9. DATASET DESCRIPTION

The dataset was obtained from Kaggle, which hosts credit card fraud detection datasets. The dataset includes credit card purchases made by European cardholders in September 2013. In this dataset, we have 492 frauds out of 284,807 transactions that occurred in the last two days. The dataset is heavily skewed, with the optimistic class (frauds) accounting for just 0.172 % of all transactions. It only has numerical input variables that have undergone a PCA transformation. We are unable to include the original features and further background information about the data due to confidentiality concerns. The principal components obtained with PCA are features V1, V2,..., V28; the only features not transformed with PCA are 'Time' and 'Amount.' The seconds elapsed between each transaction and the first transaction in the dataset are stored in the feature 'Time.' The transaction Amount is represented by the function 'Amount.' Feature 'Class' is the response variable and it takes value 1 in case of fraud and 0 otherwise.

Time	**V1**	**V2**	**…**	**V28**	**Amount**	**Class**
0	-1.35981	-0.07278	…	-0.02105	149.62	0
0	1.191857	0.266151	…	0.014724	2.69	0
1	-1.35835	-1.34016	…	-0.05975	378.66	0
1	-0.96627	-0.18523	…	0.061458	123.5	0
2	-1.15823	0.877737	…	0.215153	69.99	0
. . .	. . .	. . .	. . .	. . .	. . .	. . .
172787	-0.73279	-0.05508	…	-0.05353	24.79	0
172788	1.919565	-0.30125	…	-0.02656	67.88	0
172788	-0.24044	0.530483	…	0.104533	10	0
172792	-0.53341	-0.18973	…	0.013649	217	0

[284807 ROWS × 31 COLUMNS]

10. CRITERIA FOR COMPARISON

In order to evaluate the performance of a particular model, we make use of various parameters. The models are used on the trained dataset and the outputs obtained with the use of each model are compared systematically to those produced by the other models. Based on these comparisons, a conclusion is formed as to which is the best suited model for a particular dataset or a particular type of problem. In this paper we make use of the parameters Sensitivity, Precision and Time to compare the various models being used .

(A)Sensitivity

DOI: 10.17148/IARJSET.2022.9462

It is a measure of the proportion of actual positive cases that got predicted as positive or true positive. This actually implies that there are supposed to be some proportion of actual positive cases that would get predicted indirectly as negative. Sensitivity is also sometimes referred to as Recall. Mathematically, sensitivity can be calculated as follows:

Sensitivity = (TP)/(TP+FN)..........(1)

Where, TP = True Positive and FN = False Negative

(B) Precision

It gives a measure of the proportion of positive predictions that are truly positive. It indicates the reliability of a model in predicting a class of interest. Precision is basically a ratio of correctly positively labelled to all positively labelled. Mathematically, precision can be calculated as follows:

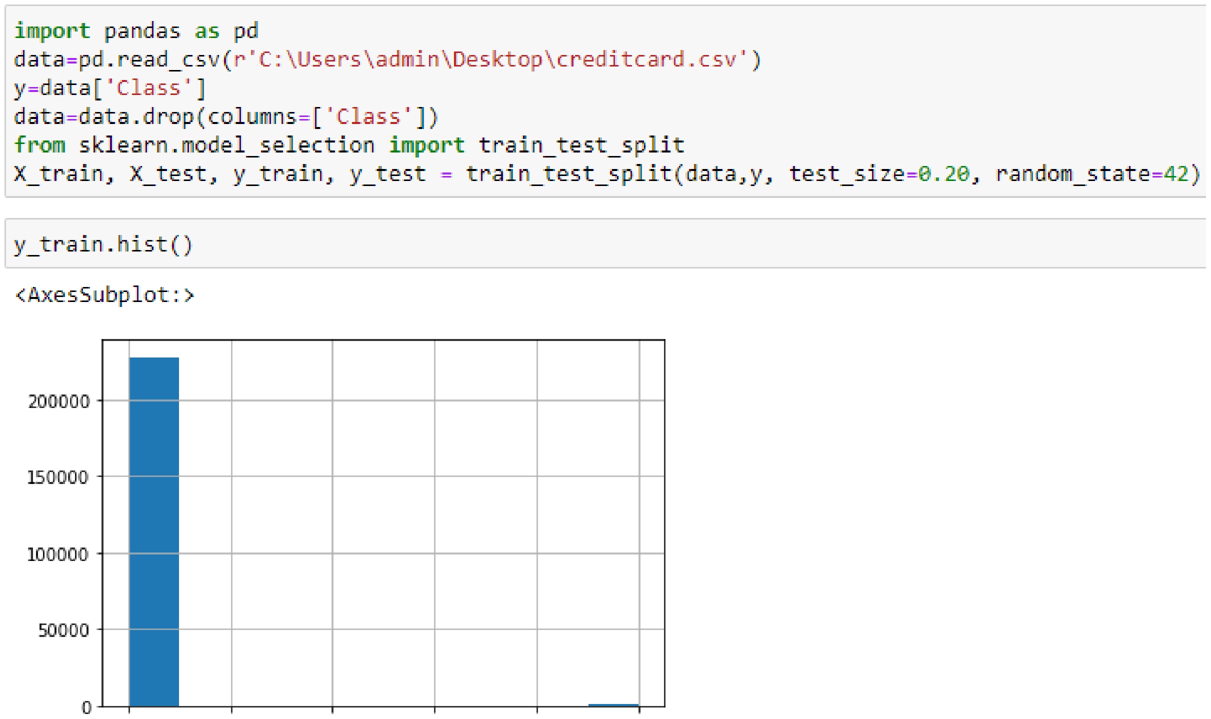
Precision = (TP)/(TP+FP)..........(2)
Where, TP = True Positive and FP = False Positive

(C) Time

Time is used as a parameter for performance evaluation of the various models that are used. We calculate the time for training the model and predicting the test data. The Time calculated is not the actual time, but the approximate time taken by a particular model. This parameter is used to compare the various models used based upon the time taken by them in handling the data.

11. IMPLEMENTATION

Here we had put some snapshots of the implemented algorithms code.



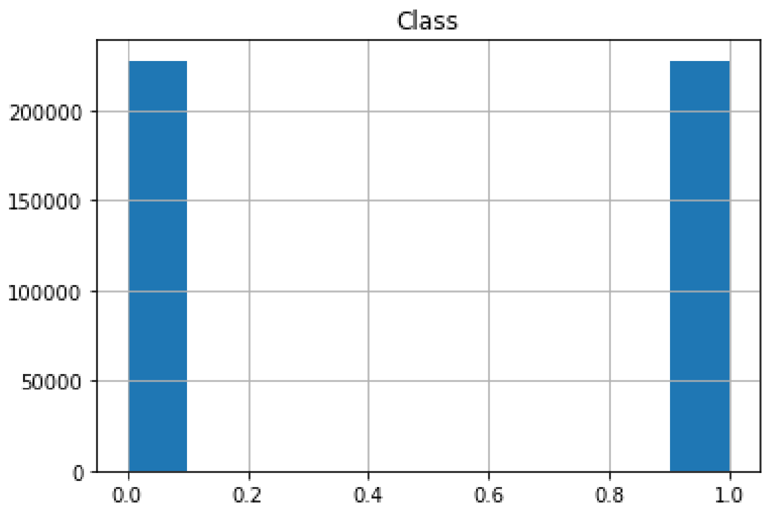
```
import pandas as pd
data=pd.read_csv(r'C:\Users\admin\Desktop\creditcard.csv')
y=data['Class']
data=data.drop(columns=['Class'])
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(data,y, test_size=0.20, random_state=42)
```

```
y_train.hist()
```

```
<AxesSubplot:>
```

Code Snippet 1

```
from imblearn.over_sampling import RandomOverSampler, SMOTE
sm = SMOTE(random_state = 42)
X_res, y_res = sm.fit_resample(X_train,y_train)
X_res = pd.DataFrame(X_res)
Y_res = pd.DataFrame(y_res)
Y_res.hist()
```



Code Snippet 2

```
from sklearn.preprocessing import MinMaxScaler as Merest_Superlative_Pulse_Counter
scaler = Merest_Superlative_Pulse_Counter()
X_res=scaler.fit_transform(X_res)
X_res_test=scaler.transform(X_test)
```

Code Snippet 3

```
from sklearn.naive_bayes import GaussianNB
gnb = GaussianNB()
gnb.fit(X_res, y_res)
gnb.score(X_res_test,y_test)
```

```
0.9769671008742671
```

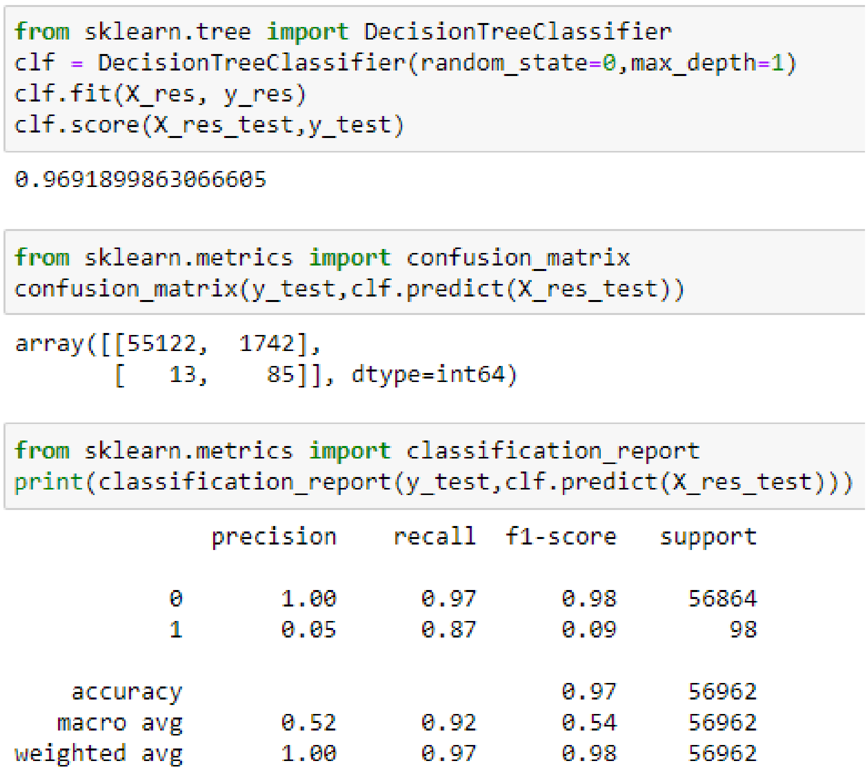
```
from sklearn.metrics import confusion_matrix
confusion_matrix(y_test,gnb.predict(X_res_test))
```

```
array([[55565,  1299],
       [   13,    85]], dtype=int64)
```

```
from sklearn.metrics import classification_report
print(classification_report(y_test,gnb.predict(X_res_test)))
```

```
              precision    recall  f1-score   support

           0       1.00      0.98      0.99     56864
           1       0.06      0.87      0.11        98

    accuracy                           0.98     56962
   macro avg       0.53      0.92      0.55     56962
weighted avg       1.00      0.98      0.99     56962
```

Code Snippet 4



```
from sklearn.tree import DecisionTreeClassifier
clf = DecisionTreeClassifier(random_state=0,max_depth=1)
clf.fit(X_res, y_res)
clf.score(X_res_test,y_test)
```

```
0.9691899863066605
```

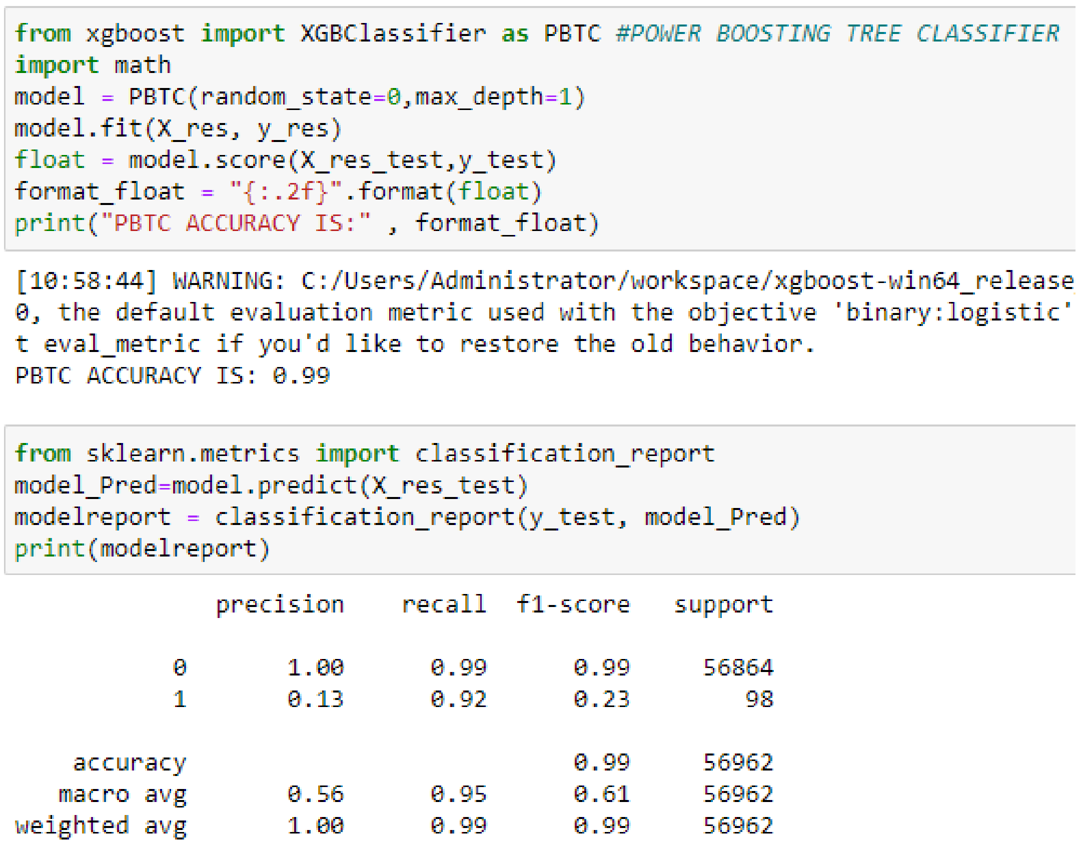
```
from sklearn.metrics import confusion_matrix
confusion_matrix(y_test,clf.predict(X_res_test))
```

```
array([[55122,  1742],
       [   13,    85]], dtype=int64)
```

```
from sklearn.metrics import classification_report
print(classification_report(y_test,clf.predict(X_res_test)))
```

```
              precision    recall  f1-score   support

           0       1.00      0.97      0.98     56864
           1       0.05      0.87      0.09        98

    accuracy                           0.97     56962
   macro avg       0.52      0.92      0.54     56962
weighted avg       1.00      0.97      0.98     56962
```

Code Snippet 5



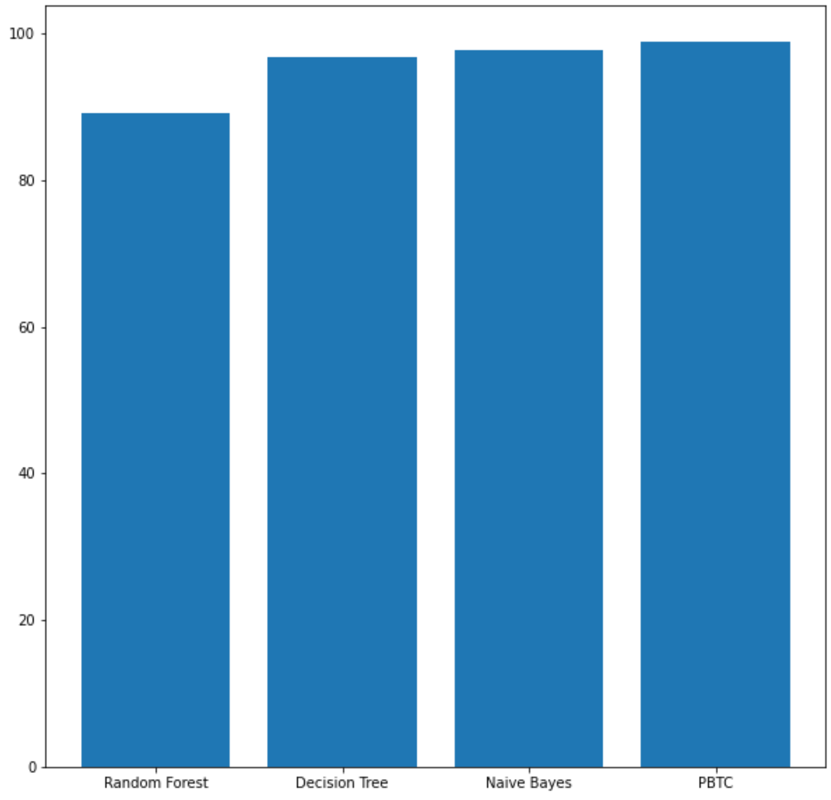
```
from xgboost import XGBClassifier as PBTC #POWER BOOSTING TREE CLASSIFIER
import math
model = PBTC(random_state=0,max_depth=1)
model.fit(X_res, y_res)
float = model.score(X_res_test,y_test)
format_float = "{:.2f}".format(float)
print("PBTC ACCURACY IS:" , format_float)
```

```
[10:58:44] WARNING: C:/Users/Administrator/workspace/xgboost-win64_release
0, the default evaluation metric used with the objective 'binary:logistic'
t eval_metric if you'd like to restore the old behavior.
PBTC ACCURACY IS: 0.99
```

```
from sklearn.metrics import classification_report
model_Pred=model.predict(X_res_test)
modelreport = classification_report(y_test, model_Pred)
print(modelreport)
```

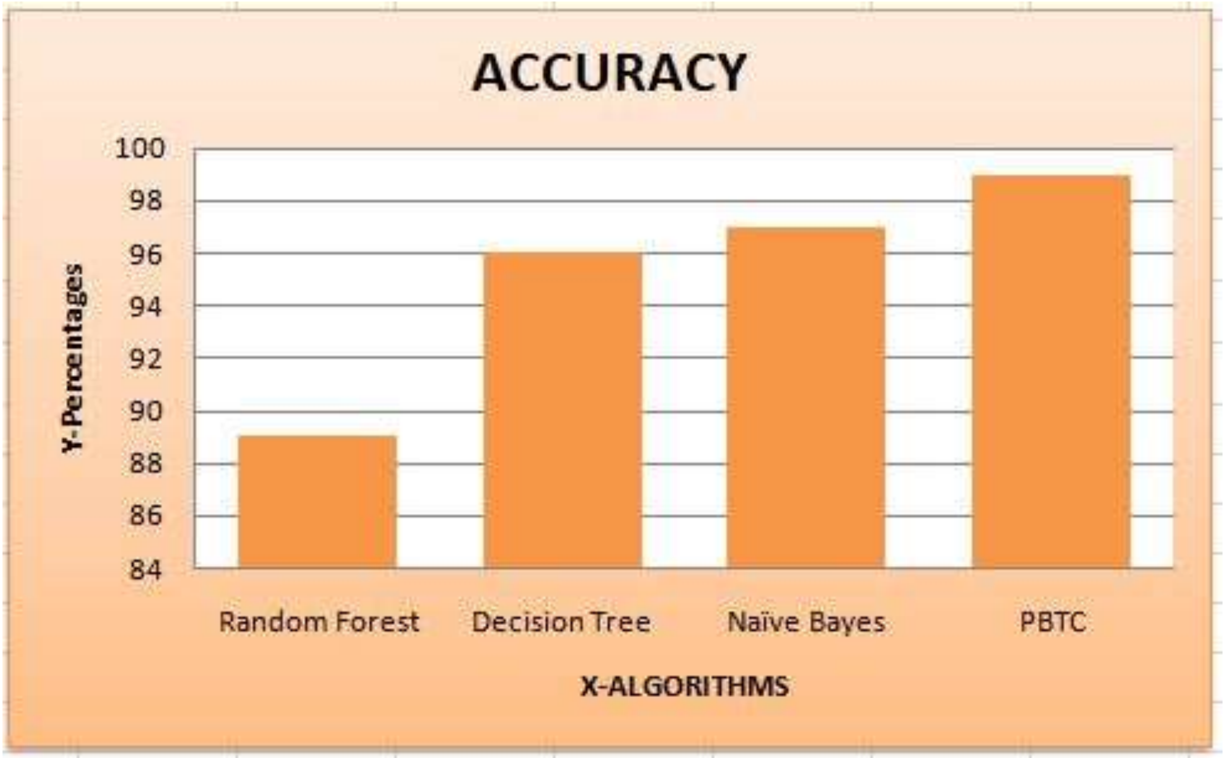
```
              precision    recall  f1-score   support

           0       1.00      0.99      0.99     56864
           1       0.13      0.92      0.23        98

    accuracy                           0.99     56962
   macro avg       0.56      0.95      0.61     56962
weighted avg       1.00      0.99      0.99     56962
```

Code Snippet 6

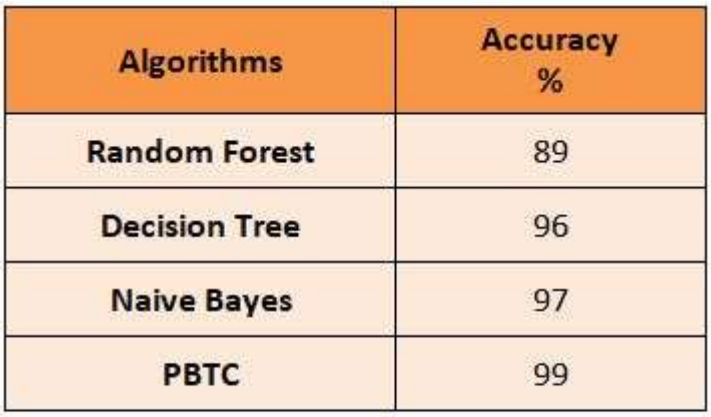


Code Snippet 7

11.1 Comparison & Result



Comparison of Algorithms Base on Accuracy



Algorithms	Accuracy %
Random Forest	89
Decision Tree	96
Naive Bayes	97
PBTC	99

Performance Table Results

Performance of these four algorithms such as PBTC, Naïve Bayes, RANFODM FOEST , and Decision Tree. PBTC has achieved highest accuracy of 99% as compare to Naïve Bayes ,RF and Decision Tree algorithms.

12. CONCLUSION & FUTURE WORK

12.1 Conclusion

The research work was carried out to compare the ability of machine learning algorithms as to how accurately they differentiate and classify the fraud and non-fraud transactions of the credit card dataset with SMOTE technique and to check out if the performance is improved or not. PBTC(POWER BOOSTING TREE CLASSIFIER)s (PBTC) showed the optimal performance for all the data proportions as compared to Naı¨ve Bayes (NB) ,Random Forest(RF)and Decision Tree (D Tree). PBTC was successful in getting higher accuracy as compared to Naı¨ve Bayes and Decision Tree. The PBTC showed the maximum accuracy of 99%, NB showed 97.70% , D-Tree showed 96.92% and RANFODM FOEST 89%. Also, PBTC shows the better Precision, Recall, and F-Measure as compare to NB and D-Tree technique.

12.2 Future Work

In the future, there can be other resampling methods, which could be put to application for the skewed dataset for credit card fraud detection. These methods could be improved to achieve better results. Also using our statistics could be compared with the other techniques like Logistic Regression, K-Nearest Neighbour, Random-Forest, Support Vector Machine, and Neural Network.

REFERENCES

PAPERS

1) Ali Ye¸ silkanat(B), Barı¸ s Bayram, Bilge K¨oro˘glu, and Se¸ cil Arslan, “An Adaptive Approach on Credit Card Fraud Detection Using Transaction Aggregation and Word Embeddings” © IFIP International Federation for Information Processing 2020, Published by Springer Nature Switzerland AG 2020, I. Maglogiannis et al. (Eds.): AIAI 2020, IFIP AICT 583, pp. 3–14, 2020.
2) Fayaz Itoo, Meenakshi, Satwinder Singh, “Comparison and analysis of logistic regression, Naı¨ve Bayes and KNN machine learning algorithms for credit card fraud detection” © Bharati Vidyapeeth’s Institute of Computer Applications and Management 2020.
3) Samidha Khatri, Aishwarya Arora, Arun Prakash Agrawal, “Supervised Machine Learning Algorithms for Credit Card Fraud Detection: A Comparison” © 2020 IEEE.
4) Aye Aye Khine, Hint Wint Khin, “Credit Card Fraud Detection Using Online Boosting with Extremely Fast Decision Tree” © 2020 IEEE.
5) Dingling Ge, Shunyu Chang, “Credit Card Fraud Detection Using Lightgbm Model” © 2020 IEEE.
6) Altyeb Altaher Taha, Sharaf Jameel Malebary, “An Intelligent Approach to Credit Card Fraud Detection Using an Optimized Light Gradient Boosting Machine ” © 2020 IEEE, VOLUME 8, 2020.

7) J. V. V. Sriram Sasank, G. Ram sahith, K.Abhinav, Meena Belwal, ”Credit Card Fraud Detection Using Various Classification and Sampling Techniques: A Comparative Study” Proceedings of the Fourth International Conference on Communication and Electronics Systems (ICCES 2019).
8) Debachudamani Prusti, Santanu Kumar Rath, “Fraudulent Transaction Detection in Credit Card by Applying Ensemble Machine Learning techniques”, 10th International Conference on Computing, Communication and Networking Technologies (ICCCNT 2019).
9) Kaithekuzhical Leena Kurien, Dr. Ajeet Chikkamannur, “Detection And Prediction Of Credit Card Fraud Transactions” © International Journal of Engineering Sciences & Research Technology (IJESRT 2019).
10) Rishi Banerjee, Gabriela Bourla, Steven Chen, Mehal Kashyap, Sonia Purohit, Jacob Battipaglia, “Comparative Analysis of Machine Learning Algorithms through Credit Card Fraud Detection” © 2018 IEEE MIT Undergraduate Research Technology Conference (URTC).

DATASET:

11) https://www.kaggle.com/janicechen1280/creditcard

WEBSITES:

12) https://www.hackerearth.com/practice/machine-learning/machine-learning-algorithms/beginners-tutorial-on-xgboost-parameter-tuning-r/tutorial/
13) https://www.hackerearth.com/practice/machine-learning/machine-learning-algorithms/ml-decision-tree/tutorial/
14) https://towardsdatascience.com/5-smote-techniques-for-oversampling-your-imbalance-data-b8155bdbe2b5
15) https://www.javatpoint.com/python-features