# CREDIT CARD FRAUD PREDICTION

TANUSHRI

DATE : 29/12/2021

## ABSTRACT

The identification of credit card fraud is currently the most common problem in the modern world. This is because internet transactions and e-commerce platforms are on the increase. Credit card fraud occurs when a credit card is stolen and used for unlawful reasons, or when a fraudster exploits the credit card information for his own interests. In today's environment, we're dealing with a lot of credit card issues. A credit card fraud detection system was created to detect fraudulent actions.

In this report we have devised an optimal method to detect fraudulent credit card transactions using the Random Forest Algorithm which is a supervised Machine learning algorithm.

## PROBLEM STATEMENT

The Credit Card Fraud Detection Problem includes modeling past credit card transactions with the knowledge of the ones that turned out to be fraud. The dataset used for training and testing the model is “Credit Card Fraud Detection” from Kaggle.This model is then used to identify whether a new transaction (from the training set) is fraudulent or not. Our aim here is to detect 100% of the fraudulent transactions while minimizing the incorrect fraud classifications. The proposed model has achieved accuracy of 99.96%.

## MARKET/CUSTOMER NEED ASESSMENT

With increasing amounts of fraudulent credit card transactions happening in today’s world, detection of such fraudulent actions is the need of the hour. Credit card fraud costs consumers and the financial company billions of dollars annually, and fraudsters continuously try to find new rules and tactics to commit illegal actions. Thus, fraud detection systems have become essential for banks and financial institutions, to minimize their losses.

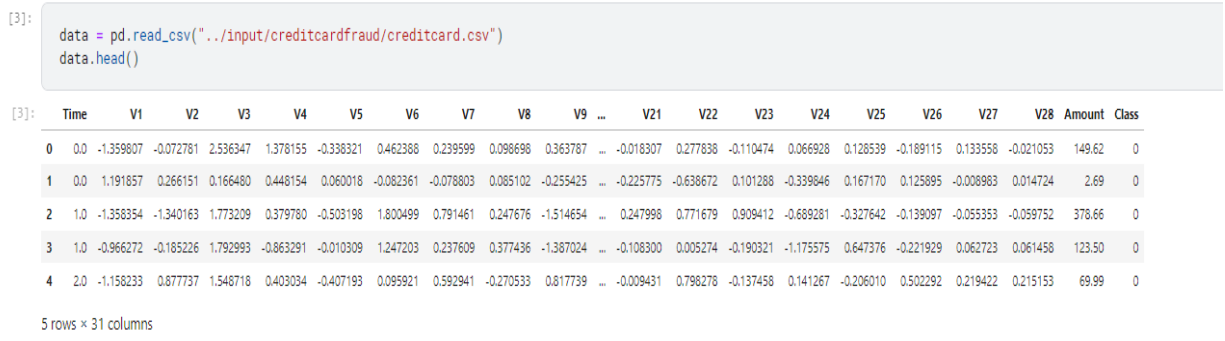
## TARGET SPECIFICATION AND CHARACTERIZATION

Once the Machine Learning-driven Fraud Protection module is integrated into the E-commerce platform, it starts tracking the transactions. Whenever a user requests a transaction, it is processed for some time. Depending on the model's prediction , the institution or merchant can realize if the transaction is fraudulent or not.

## EXTERNAL SEARCH –INFORMATION SOURCES

The dataset can be found on Kaggle. The dataset contains transactions made by credit cards in September 2013 by European cardholders. This dataset presents transactions that occurred in two days, where we have 492 frauds out of 284,807 transactions. The dataset is highly unbalanced, the positive class (frauds) account for 0.172% of all transactions.

It contains only numerical input variables which are the result of a PCA transformation. Unfortunately, due to confidentiality issues, we cannot provide the original features and more background information about the data. Features V1, V2, … V28 are the principal components obtained with PCA, the only features which have not been transformed with PCA are 'Time' and 'Amount'. Feature 'Time' contains the seconds elapsed between each transaction and the first transaction in the dataset. The feature 'Amount' is the transaction Amount, this feature can be used for example-dependent cost-sensitive learning. Feature 'Class' is the response variable and it takes value 1 in case of fraud and 0 otherwise.



[3]:
```
data = pd.read_csv("../input/creditcardfraud/creditcard.csv")
data.head()
```

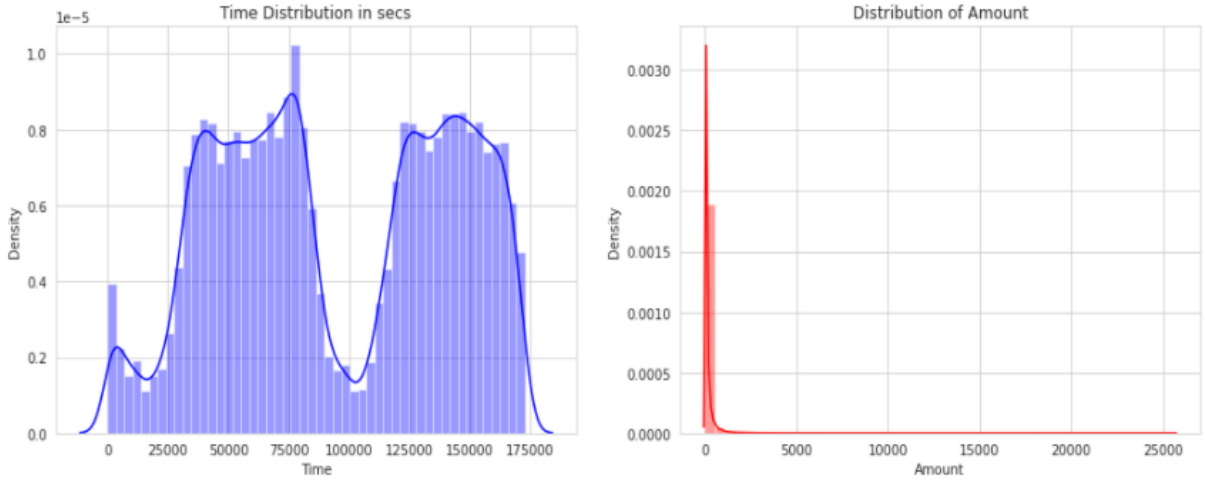
[3]:

| | Time | V1 | V2 | V3 | V4 | V5 | V6 | V7 | V8 | V9 | ... | V21 | V22 | V23 | V24 | V25 | V26 | V27 | V28 | Amount | Class |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 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 | 0.066928 | 0.128539 | -0.189115 | 0.133558 | -0.021053 | 149.62 | 0 |
| 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 | -0.339846 | 0.167170 | 0.125895 | -0.008983 | 0.014724 | 2.69 | 0 |
| 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 | -0.689281 | -0.327642 | -0.139097 | -0.055353 | -0.059752 | 378.66 | 0 |
| 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 | -1.175575 | 0.647376 | -0.221929 | 0.062723 | 0.061458 | 123.50 | 0 |
| 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 | 0.141267 | -0.206010 | 0.502292 | 0.219422 | 0.215153 | 69.99 | 0 |

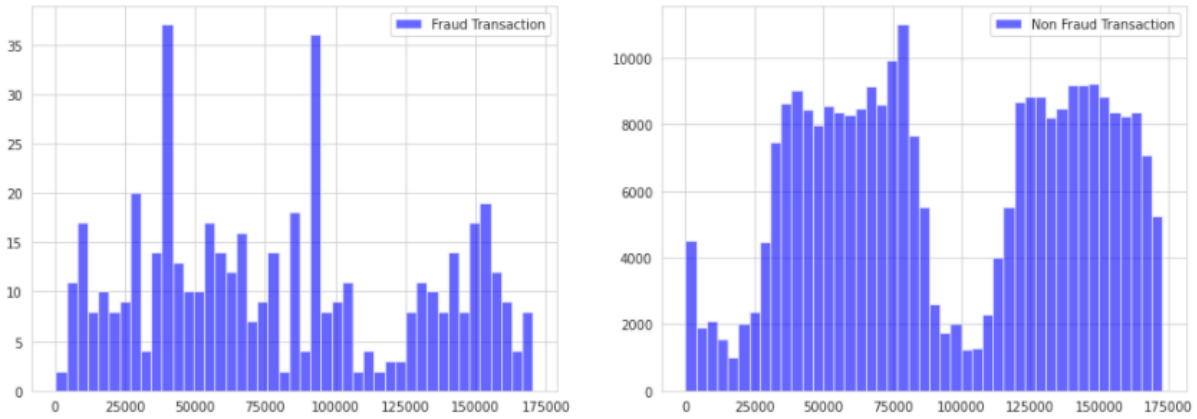
5 rows × 31 columns

# BENCHMARKING

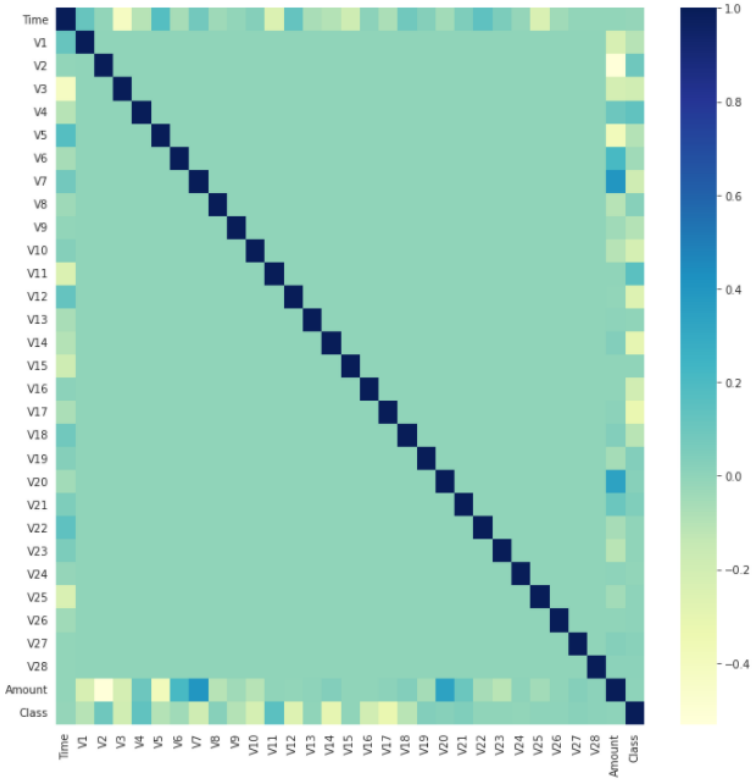

(a)Transaction Class Distribution




(b)




(c) Plot of Fraud and Non Fraud Transactions during Different time frames




(d) The correlation matrix shows that none of the components V1 to V28 have any correlation to each other.

## APPLICABLE REGULATIONS

- Availability of real time credit card transaction data.
- A survey is needed to collect some on ground information like number of visitors during different time frames and transactions made by them in different kind of institutions and shops.

## APPLICABLE CONSTRAINTS

There is lack of real time data of credit card fraud transactions due to inability to reveal a persons credit card details and personal information like transactions made etc.

## CONCEPT GENERATION

The dataset is imported and a DataFrame is created. With the help of exploratory data analysis, we try to come up with insights regarding if the type of transaction is associated with any timeframe. There seems to be no such correlation. The columns of Time and Amount are scaled. The dataset is split into 2 categories: Training and Test set.

## MODEL DEVELOPMENT AND METHODOLOGY

The Random Forest algorithm is used to train and test the model. Various metrics are observed by testing the model on the Testing data.

### RANDOM FOREST ALGORITHM

Random forest is a Supervised Machine Learning Algorithm that is used widely in Classification and Regression problems. It builds decision trees on different samples and takes their majority vote for classification and average in case of regression.

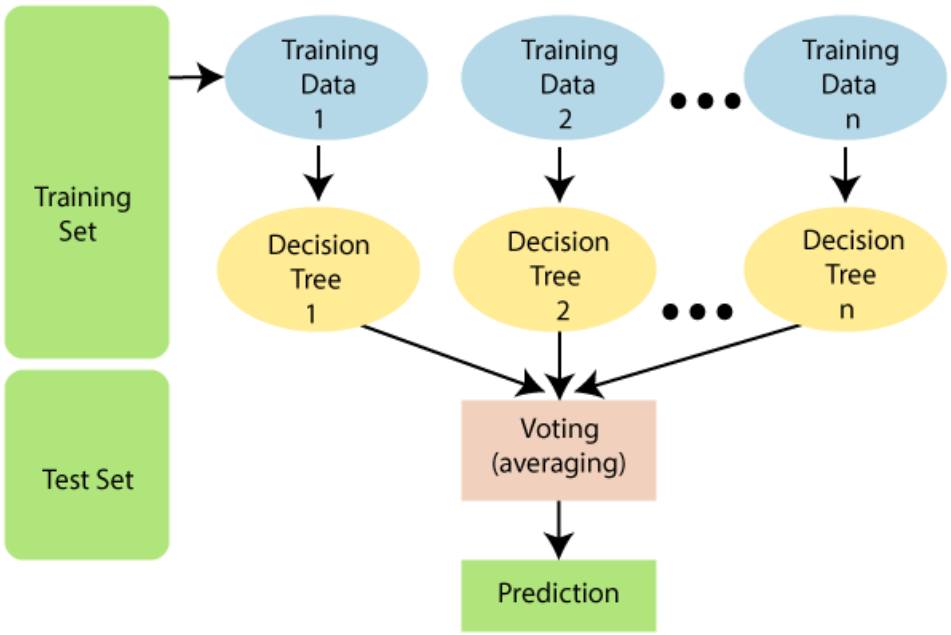
One of the most important features of the Random Forest Algorithm is that it can handle the data set containing continuous variables as in the case of regression

and categorical variables as in the case of classification. It performs better results for classification problems.

It is based on the concept of ensemble learning, which is a process of combining multiple classifiers to solve a complex problem and to improve the performance of the model.

Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset." Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, and it predicts the final output.

The greater number of trees in the forest leads to higher accuracy and prevents the problem of overfitting.



```python
from sklearn.ensemble import RandomForestClassifier

rf_clf = RandomForestClassifier(n_estimators=100, oob_score=False)
rf_clf.fit(X_train, y_train)

y_train_pred = rf_clf.predict(X_train)
y_test_pred = rf_clf.predict(X_test)

print_score(y_train, y_train_pred, train=True)
print_score(y_test, y_test_pred, train=False)
```

The obtained accuracy results are

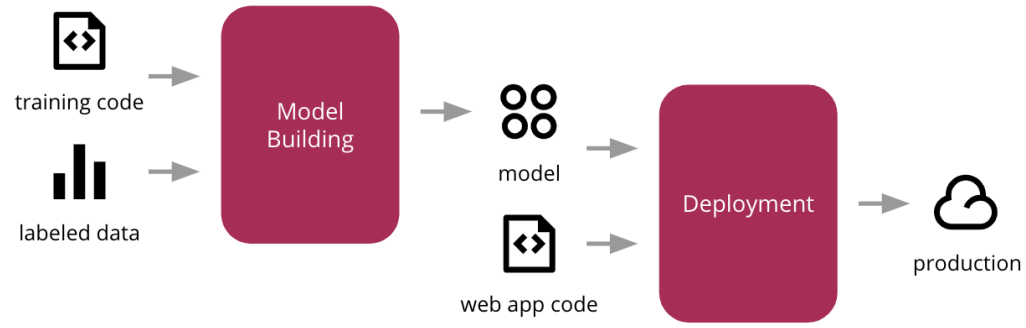
```
Train Result:

Accuracy Score: 100.00%
Classification Report:
                    0       1  accuracy  macro avg  weighted avg
precision       1.000   1.000     1.000      1.000         1.000
recall          1.000   1.000     1.000      1.000         1.000
f1-score        1.000   1.000     1.000      1.000         1.000
support    159204.000 287.000     1.000 159491.000    159491.000
Confusion Matrix:
 [[159204      0]
 [     0    287]]

Test Result:

Accuracy Score: 99.96%
Classification Report:
                   0       1  accuracy  macro avg  weighted avg
precision      1.000   0.956     1.000      0.978         1.000
recall         1.000   0.794     1.000      0.897         1.000
f1-score       1.000   0.867     1.000      0.934         1.000
support    85307.000 136.000     1.000  85443.000     85443.000
Confusion Matrix:
 [[85302     5]
 [   28   108]]
```

**Why do we use Random Forest?**

- It takes less training time as compared to other algorithms.
- It predicts output with high accuracy, even for the large dataset it runs efficiently.
- It can also maintain accuracy when a large proportion of data is missing.

## CONCEPT DEVELOPMENT

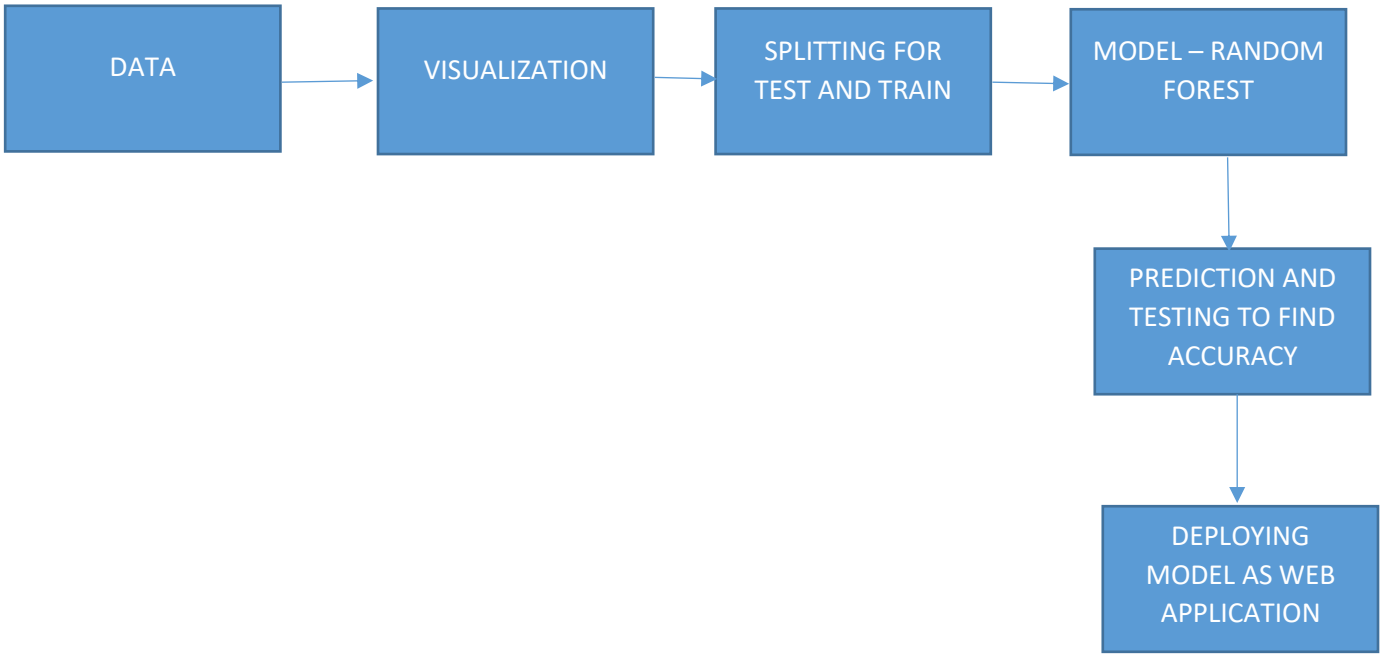
We can use appropriate APIs such as Flask in order to deploy the model. In this way it would be made more user friendly to be used in business small scale or large scale.




## FINAL REPORT PROTOTYPE

Back end: This involves analyzing the dataset and finding out about different parameters such as Time, Amount and the transaction category. Exploratory data analysis is performed and different parameters are plotted with respect to transactions. The data is split into training and test set and model is trained.

Front end: With the help of APIs like Flask we create a web application for deployment of the model. This web application can be used in different small scale and large scale firms for credit card fraud detection.




## CODE IMPLEMENTATION

https://www.kaggle.com/tanushrikumar/credit-card-fraud-detection/notebook

## CONCLUSION

With the help of this prototype using Machine learning models and programming language Python, we can try to solve the prevalent problem of credit card fraud detection. Credit card fraud detection systems need to be implemented by merchants, banks and institutions as it would lead to financial losses otherwise.

REFERENCES

https://www.sciencedirect.com/science/article/pii/S187705092030065X

https://ieeexplore.ieee.org/document/9121114

https://www.javatpoint.com/machine-learning-random-forest-algorithm