

CREDIT CARD FRAUD DETECTION USING MACHINE LEARNING

Parisa Naga Jayasri

Abstract- This Project is focused on credit card fraud detection in real world scenarios. Nowadays credit card frauds are drastically increasing in number as compared to earlier times. Criminals are using fake identity and various technologies to trap the users and get the money out of them. Therefore, it is very essential to find a solution to these types of frauds. In this proposed project we designed a model to detect the fraud activity in credit card transactions. This system can provide most of the important features required to detect illegal and illicit transactions. As technology changes constantly, it is becoming difficult to track the behavior and pattern of criminal transactions. To come up with the solution one can make use of technologies with the increase of machine learning, artificial intelligence and other relevant fields of information technology; it becomes feasible to automate this process and to save some of the intensive amounts of labor that is put into detecting credit card fraud. Initially, we will collect the credit card usage data-set by users and classify it as trained and testing dataset using a random forest algorithm and decision trees. Using this feasible algorithm, we can analyze the larger data-set and user provided current data-set. Then augment the accuracy of the result data. Proceeded with the application of processing of some of the attributes provided which can find affected fraud detection in viewing the graphical model of data visualization. The performance of the techniques is gauged based on accuracy, sensitivity, and specificity, precision. The results is indicated concerning the best accuracy for Random Forest are unit 98.6% respectively.

1. INTRODUCTION

Nowadays Credit card usage has been drastically increased across the world, now people believe in going cashless and are completely dependent on online transactions. The credit card has made the digital transaction easier and more accessible. A huge number of dollars of loss are caused every year by the criminal credit card transactions. Fraud is as old as mankind itself and can take an unlimited variety of different forms. The PwC global economic crime survey of 2017 suggests that approximately 48% of organizations experienced economic crime. Therefore, there's positively a necessity to unravel the matter of credit card fraud detection. Moreover, the growth of new technologies provides supplementary ways in which criminals may commit a scam. The use of credit cards is predominant in modern day society and credit card fraud has been kept on increasing in recent years. Huge Financial losses have been fraudulent effects on not only merchants and banks but also the individual person who are using the credits. Fraud may also affect the reputation and image of a merchant causing non-financial losses that. For example, if a cardholder is a victim of fraud with a certain company, he may no longer trust their business and choose a competitor. Fraud Detection is the process of monitoring the transaction behavior of a cardholder to

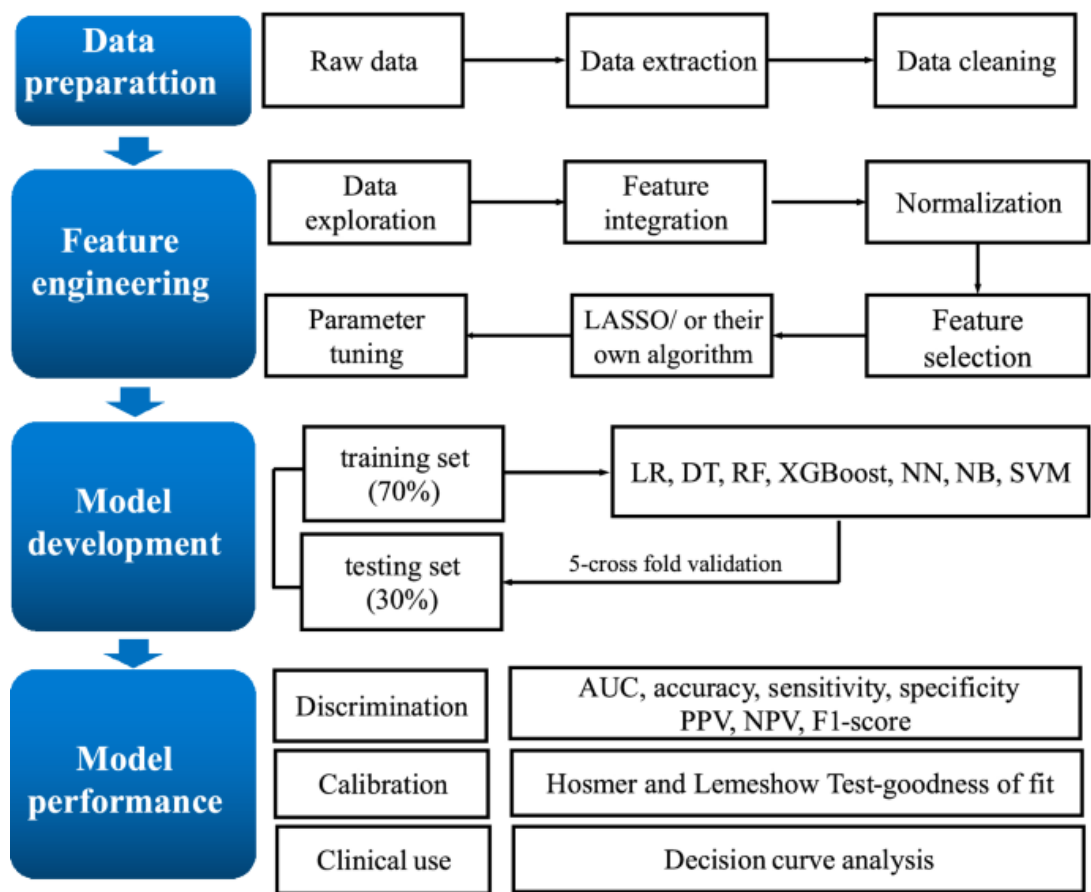
detect whether an incoming transaction is authentic and authorized or not otherwise it will be detected as illicit. In a planned system, we are applying the logistic regression algorithm for classifying the credit card dataset.

1.1 Logistic Regression

Logistic regression is used for binary classification where we use sigmoid function, that takes input as independent variables and produces a probability value between 0 and 1.

1.2 SCOPE OF THE PROPOSED WORK

In this proposed project we designed a protocol or a model to detect the fraud activity in credit card transactions. This system is capable of providing most of the essential features required to detect fraudulent and legitimate transactions. As technology changes, it becomes difficult to track the Modeling and pattern of fraudulent transactions. With the rise of machine learning, artificial intelligence and other relevant fields of information technology, it becomes feasible to automate this process and to save some of the intensive amount of labour that is put into detecting credit card fraud Detection.



2. SOFTWARE AND HARDWARE REQUIREMENT

2.1 Hardware

- OS – Windows 7, 8 and 10 (32 and 64 bit)
- RAM – 4GB

2.2 Software

- Python
- Anaconda

3. SYSTEM ARCHITECTURE

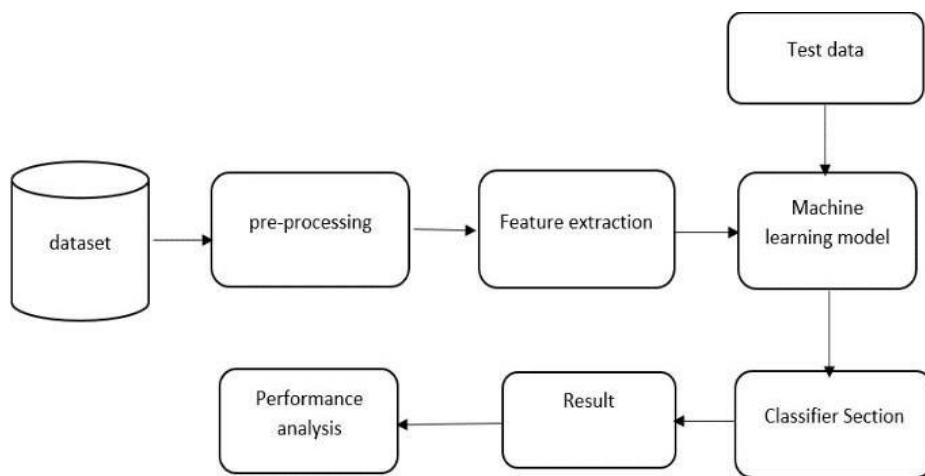


Fig 1 System Architecture

3.1 PACKAGES

Which are being used for data exploration, pro processing and for using random forest algorithm are:

- **NumPy**: For simple arrays.
- **Pandas**: For reading the file.
- **SciKit**: Learn- for pre-processing.
- **LogisticRegression**
- **Accuracy_score**

```
: import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
```

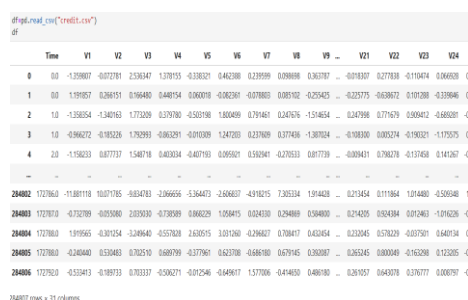
Importing libraries

4. MODULES

- Data collection
- Data pre-processing
- Feature extraction
- Evaluation model

4.1 Data Collection:

Data used in this paper is a set of product reviews collected from credit card transactions records. This step is concerned with selecting the subset of all available data that you will be working with. ML problems start with data preferably, lots of data (examples or observations) for which you already know the target answer. Data for which you already know the target answer is called labelled data.



The screenshot shows a Jupyter Notebook cell with the code `df=pd.read_csv("credit.csv")` and the resulting DataFrame. The DataFrame has columns: Time, V1, V2, V3, V4, V5, V6, V7, V8, V9, ..., V21, V22, V23, V24. The first five rows are displayed, showing numerical values for each feature. The last row is truncated with an ellipsis.

| | Time | V1 | V2 | V3 | V4 | V5 | V6 | V7 | V8 | V9 | ... | V21 | V22 | V23 | V24 |
|--------|----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----|-----------|-----------|-----------|-----------|
| 0 | 0.0 | -1.259807 | -0.072781 | 2.536347 | 1.278153 | -0.338321 | 0.462388 | 0.239599 | 0.098888 | 0.363787 | ... | -0.018307 | 0.277838 | -0.110474 | 0.066828 |
| 1 | 0.0 | 1.191957 | 0.268751 | 0.166480 | 0.448154 | 0.060018 | -0.062361 | -0.070803 | 0.085192 | -0.255425 | ... | -0.225775 | -0.638672 | 0.101208 | -0.339646 |
| 2 | 1.0 | -1.302554 | -1.340163 | 1.773209 | 0.379780 | -0.530788 | 1.800409 | 0.791461 | 0.247676 | -1.514654 | ... | 0.247998 | 0.771679 | 0.000412 | -0.085091 |
| 3 | 1.0 | -0.966272 | -0.193226 | 1.762993 | -0.063291 | -0.010385 | 1.247203 | 0.237609 | 0.377436 | -1.387024 | ... | -0.108300 | 0.005234 | -0.190321 | -1.175075 |
| 4 | 2.0 | -1.150233 | 0.077337 | 1.548718 | 0.403034 | -0.407193 | 0.095921 | 0.502941 | -0.270533 | 0.017739 | ... | -0.009431 | 0.706278 | -0.157458 | 0.141267 |
| ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| 294002 | 172706.0 | -0.180119 | 10.070165 | -0.034783 | -2.066656 | -5.364473 | -3.606037 | -0.918175 | 7.302334 | 1.914428 | ... | 0.213454 | 0.111864 | 1.014480 | -0.500540 |
| 294003 | 172707.0 | -0.172789 | -0.055080 | 2.039330 | -0.738099 | 0.066229 | 1.028415 | 0.024330 | 0.294889 | 0.594000 | ... | 0.214205 | 0.554384 | 0.052463 | -1.056226 |
| 294004 | 172708.0 | 1.919565 | -0.301254 | -0.249640 | -0.557820 | 2.630515 | 3.017040 | -0.268627 | 0.708417 | 0.430164 | ... | 0.233045 | 0.578228 | -0.037501 | 0.640134 |
| 294005 | 172709.0 | -0.240440 | 0.530483 | 0.702510 | 0.680799 | -0.377961 | 0.623708 | -0.686180 | 0.679145 | 0.300387 | ... | 0.265245 | 0.000040 | -0.163208 | 0.123205 |
| 294006 | 172710.0 | -0.550413 | -0.180733 | 0.703337 | -0.506271 | -0.012540 | -0.649617 | 1.577006 | -0.474830 | 0.480180 | ... | 0.261957 | 0.043078 | 0.376777 | 0.000797 |

294007 rows x 27 columns

Fig. 2: Importing python packages for data exploration, preprocessing

4.2 Data Pre-processing

Pre-processing is the process of three important and common steps as follows:

- **Formatting:** It is the process of putting the data in a legitimate way that it would be suitable to work with. Format of the data files should be formatted according to the need. Most recommended format is .csv files.
- **Cleaning:** Data cleaning is a very important procedure in the path of data science as it constitutes the major part of the work. It includes removing missing data and complexity with naming category and so on. For most of the data scientists, Data Cleaning continues of 80% of work.
- **Sampling:** This is the technique of analyzing the subsets from whole large datasets, which could provide a better result and help in understanding the behavior and pattern of data in an integrated way

4.3 Data Exploration

```
df.head()
```

| | Time | V1 | V2 | V3 | V4 | V5 | V6 | V7 | V8 | V9 | ... | V21 | V22 | V23 | V24 | V25 |
|---|------|-----------|-----------|----------|-----------|-----------|-----------|-----------|-----------|-----------|-----|-----------|-----------|-----------|-----------|-----------|
| 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 |
| 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 |
| 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 |
| 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 |
| 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 |

5 rows × 31 columns

Fig. 3: Data exploration

6.3.1 Pre-processing with python commandsSTEP 1:

```
df['Class'].value_counts()
```

```
Class
0    284315
1      492
Name: count, dtype: int64
```

Fig. 4: Pre-processing

STEP2:

```
legal=df[df.Class==0]  
fraud=df[df.Class==1]
```

Fig. 5: Preprocessing Step 2

STEP 3: Acquired trained and testing dataset from the large dataset

```
print(legal.shape)
```

```
(284315, 31)
```

```
print(fraud.shape)
```

```
(492, 31)
```

Fig. 6: Describing shape

```
legal.Amount.describe()
```

```
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
```

```
fraud.Amount.describe()
```

```
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
```

Fig. 7: Process of data extraction

4.4 Data visualization

Data Visualisation is the method of representing the data in a graphical and pictorial way, data scientists depict a story by the results they derive from analysing and visualising the data. The best tool used is Tableau which has many features to play around with data and fetch wonderful results.

```
legal_sample=legal.sample(n=492)
```

```
new_dataset=pd.concat([legal_sample,fraud],axis=0)
```

Fig. 8

```
new_dataset.head()
```

| | Time | V1 | V2 | V3 | V4 | V5 | V6 | V7 | V8 | V9 | ... | V21 |
|--------|----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----|-----------|
| 49003 | 43871.0 | 1.173123 | 0.283653 | 0.243150 | 0.562312 | -0.222112 | -0.974534 | 0.362160 | -0.211114 | -0.548981 | ... | -0.329416 |
| 102061 | 68038.0 | 1.019025 | -0.142003 | 1.032570 | 1.193445 | -0.438068 | 0.928477 | -0.697154 | 0.455687 | 0.390736 | ... | 0.269868 |
| 181366 | 124947.0 | 2.126662 | -1.128220 | -0.580418 | -0.930368 | -1.139242 | -0.388630 | -1.158064 | 0.069708 | 0.073824 | ... | 0.344865 |
| 149266 | 90913.0 | -1.786994 | 1.431856 | 1.604550 | 5.076226 | -1.175423 | 0.109534 | -0.493523 | 0.954915 | -0.422498 | ... | -0.183135 |
| 154839 | 103239.0 | -1.805682 | -0.670727 | 0.125133 | -0.286330 | 3.321070 | 0.581253 | 0.516257 | 0.276393 | 0.009190 | ... | 0.240974 |

5 rows × 31 columns

Fig.9

```
new_dataset["Class"].value_counts()
```

Class

0 492

1 492

Name: count, dtype: int64

Fig.10

```
X=new_dataset.drop(columns="Class",axis=1)
Y=new_dataset["Class"]
```

Fig.11

```
df.groupby("Class").mean()
```

| | Time | V1 | V2 | V3 | V4 | V5 | V6 | V7 | V8 | V9 | ... | V20 | V21 | |
|-------|--------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|-----|-----------|-----------|-------|
| Class | | | | | | | | | | | | | | |
| 0 | 94838.202258 | 0.008258 | -0.006271 | 0.012171 | -0.007860 | 0.005453 | 0.002419 | 0.009637 | -0.000987 | 0.004467 | ... | -0.000644 | -0.001235 | -0.00 |
| 1 | 80746.806911 | -4.771948 | 3.623778 | -7.033281 | 4.542029 | -3.151225 | -1.397737 | -5.568731 | 0.570636 | -2.581123 | ... | 0.372319 | 0.713588 | 0.01 |

2 rows × 30 columns

```
#splitting the data into training and testing data
```

```
X_train,X_test,Y_train,Y_test=train_test_split(X,Y,test_size=0.2,stratify=Y,random_state=2)
```

```
print(X.shape,X_train.shape,X_test.shape)
```

(984, 30) (787, 30) (197, 30)

Fig.12 Training and fitting of dataset

4.5 Feature extraction

Feature extraction is the process of studying the behavior and pattern of the analyzed data and draw the features for further testing and training. Finally, our models are trained using the Classifier algorithm. We use classify module on Natural Language Toolkit library on Python. We use the labelled dataset gathered. The rest of our labelled data will be used to evaluate the models. Some machine learning algorithms were used to classify pre-processed data. The chosen classifiers were Logistic Regression. These algorithms are very popular in text classification tasks.

```
#Logistic Regression  
model=LogisticRegression()
```

```
#training the logisticregression model with training data  
model.fit(X_train,Y_train)
```

▼ LogisticRegression

```
LogisticRegression()
```

```
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,  
                    intercept_scaling=1, l1_ratio=None, max_iter=100,  
                    multi_class='auto', n_jobs=None, penalty='l2',  
                    random_state=None, solver='lbfgs', tol=0.0001, verbose=0,  
                    warm_start=False)
```

▼ LogisticRegression

```
LogisticRegression()
```

Fig 12 Logistic Regression

4.6 Evaluation model

Model Evaluation is an essential part of the model development process. It helps to find the best model that represents our data and how well the selected model will work in the future. Evaluating model performance with the data used for training is not acceptable in data science because it can effortlessly generate overoptimistically and over fitted models. To avoid overfitting, evaluation methods such as hold out and cross-validations are used to test to evaluate model performance. The result will be in the visualized form. Representation of classified data in the form of graphs. Accuracy is well-defined as the proportion of precise predictions for the test data. It can be calculated easily by mathematical calculation i.e. dividing the number of correct predictions by the number of total predictions.

5. ALGORITHM

5.1 Logistic Regression

Logistic regression is a **supervised machine learning algorithm** used for **classification tasks** where the goal is to predict the probability that an instance belongs to a given class or not. Logistic regression is a statistical algorithm which analyze the relationship between two data factors.

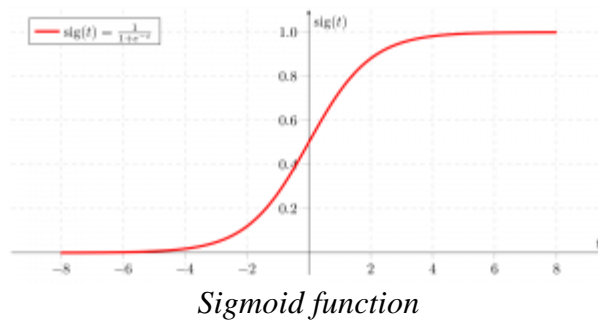
5.2 Types of Logistic Regression

On the basis of the categories, Logistic Regression can be classified into three types:

1. **Binomial:** In binomial Logistic regression, there can be only two possible types of the dependent variables, such as 0 or 1, Pass or Fail, etc.
2. **Multinomial:** In multinomial Logistic regression, there can be 3 or more possible unordered types of the dependent variable, such as “cat”, “dogs”, or “sheep”
3. **Ordinal:** In ordinal Logistic regression, there can be 3 or more possible ordered types of dependent variables, such as “low”, “Medium”, or “High”.

5.3. Sigmoid Function

we use the sigmoid function where the input will be z and we find the probability between 0 and 1. i.e. predicted y .



As shown above, the figure sigmoid function converts the continuous variable data into the probability i.e. between 0 and 1.

- $\sigma(z)$ tends towards 1 as
- $\sigma(z)$ tends towards 0 as
- $\sigma(z)$ is always bounded between 0 and 1

```
# accuracy on training data
X_train_prediction = model.predict(X_train)
training_data_accuracy = accuracy_score(X_train_prediction, Y_train)

print('Accuracy on Training data : ', training_data_accuracy)

Accuracy on Training data :  0.9364675984752223

# accuracy on test data
X_test_prediction = model.predict(X_test)
test_data_accuracy = accuracy_score(X_test_prediction, Y_test)

print('Accuracy score on Test Data : ', test_data_accuracy)

Accuracy score on Test Data :  0.9390862944162437
```

Fig. 13: Accuracy for the dataset

6. CONCLUSION

Hence, we have acquired the result of an accurate value of credit card fraud detection i.e. 0.9390862944162437(93.90%) using a Logistic Regression algorithm with new enhancements. In comparison to existing modules, this proposed module is applicable for the larger dataset and provides more accurate results. The Logistic Regression algorithm will provide better performance with many training data, but speed during testing and application will still suffer. Usage of more pre-processing techniques would also assist.