CREDIT CARD FRAUD DETECTION USING R

Prepared by

Leeba Ann Varghese

Table of Contents

1 INTRODUCTION

As the Banking industry is advancing to digitalization, the importance of cybersecurity is becoming crucial. Records show that the risk factors and fraud transactions are increasing even with the strict authentication mechanisms implemented in the banking applications today. As such, there is an urgent need to develop and relook into the current fraud detection mechanisms. In this project, we will use the credit card transactions dataset provided by dataflair (Credit-Card-Dataset.zip - Google Drive) .This data is been uploaded to my GitHub repository .On running the code ,the data will be automatically downloaded ,unzipped and stored into an R dataframe.

2 PROJECT OBJECTIVE

The objective of this project is to analyze the Credit card transactions data and prepare a predictive model to identify and predict fraudulent transactions. I have used 3 models in this project.

1. Logistic Regression Model

2. Decision Tree Model

3. Random Forest Model

3 DATA CLEANUP - PREPARATION OF DATA

3.1 Sources

In this project, we have used the Credit card transactions dataset available in the dataflair website . This data is been uploaded to my GitHub repository.

Below is the link to my GitHub repository.

Project2-CreditcardFraudDetection/creditcard.zip at master · 1eeba/Project2-CreditcardFraudDetection (github.com)

On running the code, the data will be automatically downloaded, unzipped and stored into an R dataframe.

3.2 Preparing the Credit card dataset

We have loaded the Credit card transactions data into R using read.csv function.

```
> creditcard_data
# A tibble: 284,807 × 31
    Time      V1       V2      V3      V4      V5      V6       V7      V8
   <dbl>   <dbl>    <dbl>   <dbl>   <dbl>   <dbl>   <dbl>    <dbl>   <dbl>
 1     0 -1.36   -0.0728   2.54     1.38   -0.338   0.462    0.240    0.0987
 2     0  1.19    0.266    0.166    0.448   0.0600 -0.0824 -0.0788    0.0851
 3     1 -1.36   -1.34     1.77     0.380  -0.503   1.80     0.791    0.248
 4     1 -0.966  -0.185    1.79    -0.863  -0.0103  1.25     0.238    0.377
 5     2 -1.16    0.878    1.55     0.403  -0.407   0.0959   0.593   -0.271
 6     2 -0.426   0.961    1.14    -0.168   0.421  -0.0297   0.476    0.260
 7     4  1.23    0.141    0.0454   1.20    0.192   0.273   -0.00516  0.0812
 8     7 -0.644   1.42     1.07    -0.492   0.949   0.428    1.12    -3.81
 9     7 -0.894   0.286   -0.113   -0.272   2.67    3.72     0.370    0.851
10     9 -0.338   1.12     1.04    -0.222   0.499  -0.247    0.652    0.0695
# … with 284,797 more rows, and 22 more variables: V9 <dbl>, V10 <dbl>,
#   V11 <dbl>, V12 <dbl>, V13 <dbl>, V14 <dbl>, V15 <dbl>, V16 <dbl>,
#   V17 <dbl>, V18 <dbl>, V19 <dbl>, V20 <dbl>, V21 <dbl>, V22 <dbl>,
#   V23 <dbl>, V24 <dbl>, V25 <dbl>, V26 <dbl>, V27 <dbl>, V28 <dbl>,
#   Amount <dbl>, Class <dbl>
```

4 DATA MODELING

4.1 Splitting the data into Test set and Training set

The data is split into test set and training set using the split function with a split ratio of 0.80.

The dimensions of credit card dataset, test set and training set are shown below:

```
> dim(creditcard_data)
[1] 284807     31
> dim(cc_train_set)
[1] 227846     31
> dim(cc_test_set)
[1] 56961    31
```

4.2 Scaling the Amount field in both the sets

Using the scale() function, we will standardize the Amount field in both the test and train sets so that there are no extreme values.

5 DATA EXPLORATION AND DATA VISUALIZATION

We will now visualize the creditcard dataset.

```
· dim(creditcard_data)
1] 284807     31
```

The data set has 28407 observations and 31 columns.

Below is the structure of the dataset.

```
 $ Time   : num  0 0 1 2 4 7 9 10 10 11 ...
 $ V1     : num  -1.36 1.192 -1.358 -0.426 1.23 ...
 $ V2     : num  -0.0728 0.2662 -1.3402 0.9605 0.141 ...
 $ V3     : num  2.5363 0.1665 1.7732 1.1411 0.0454 ...
 $ V4     : num  1.378 0.448 0.38 -0.168 1.203 ...
 $ V5     : num  -0.338 0.06 -0.503 0.421 0.192 ...
 $ V6     : num  0.4624 -0.0824 1.8005 -0.0297 0.2727 ...
 $ V7     : num  0.2396 -0.0788 0.79146 0.4762 -0.00516 ...
 $ V8     : num  0.0987 0.0851 0.2477 0.2603 0.0812 ...
 $ V9     : num  0.364 -0.255 -1.515 -0.569 0.465 ...
 $ V10    : num  0.0908 -0.167 0.2076 -0.3714 -0.0993 ...
 $ V11    : num  -0.552 1.613 0.625 1.341 -1.417 ...
 $ V12    : num  -0.6178 1.0652 0.0661 0.3599 -0.1538 ...
 $ V13    : num  -0.991 0.489 0.717 -0.358 -0.751 ...
 $ V14    : num  -0.311 -0.144 -0.166 -0.137 0.167 ...
 $ V15    : num  1.4682 0.6356 2.3459 0.5176 0.0501 ...
 $ V16    : num  -0.47 0.464 -2.89 0.402 -0.444 ...
 $ V17    : num  0.20797 -0.1148 1.10997 -0.05813 0.00282 ...
 $ V18    : num  0.0258 -0.1834 -0.1214 0.0687 -0.612 ...
 $ V19    : num  0.404 -0.1458 -2.2619 -0.0332 -0.0456 ...
 $ V20    : num  0.2514 -0.0691 0.525 0.085 -0.2196 ...
 $ V21    : num  -0.0183 -0.2258 0.248 -0.2083 -0.1677 ...
 $ V22    : num  0.278 -0.639 0.772 -0.56 -0.271 ...
 $ V23    : num  -0.1105 0.1013 0.9094 -0.0264 -0.1541 ...
 $ V24    : num  0.0669 -0.3398 -0.6893 -0.3714 -0.7801 ...
 $ V25    : num  0.129 0.167 -0.328 -0.233 0.75 ...
 $ V26    : num  -0.189 0.126 -0.139 0.106 -0.257 ...
 $ V27    : num  0.13356 -0.00898 -0.05535 0.25384 0.03451 ...
 $ V28    : num  -0.02105 0.01472 -0.05975 0.08108 0.00517 ...
 $ Amount: num  149.62 2.69 378.66 3.67 4.99 ...
 $ Class : int  0 0 0 0 0 0 0 0 0 0 ...
>
```

Time : Indicates the time in seconds elapsed between each transaction and the first transaction in the dataset.

Columns V1-V28 : These are variables that have been transformed into numeric as a part of PCA transformation. Principal Component Analysis (PCA). PCA is a dimensionality-reduction method that is often used to reduce the dimensionality of large data sets, by transforming a large set of variables into a smaller one that still contains most of the information in the large set.

Amount **:** is the transaction amount.

Class **:** is the field that indicates whether a particular transaction is fraudulent or not. It takes two values-0 or 1.Value 1 indicates the transaction is fraud and value 0 indicates a genuine transaction.

5.1 First 6 rows of the dataset

We will have a look at the first 6 rows of the dataset.

```
          V18         V19         V20         V21         V22         V23         V24         V25         V26
1  0.02579058  0.40399296  0.25141210 -0.01830678  0.2778376 -0.11047391  0.06692807  0.1285394 -0.1891148
2 -0.18336127 -0.14578304 -0.06908314 -0.22577525 -0.6386720  0.10128802 -0.33984648  0.1671704  0.1258945
3 -0.12135931 -2.26185710  0.52497973  0.24799815  0.7716794  0.90941226 -0.68928096 -0.3276418 -0.1390966
6  0.06865315 -0.03319379  0.08496767 -0.20825351 -0.5598248 -0.02639767 -0.37142658 -0.2327938  0.1059148
7 -0.61198734 -0.04557504 -0.21963255 -0.16771627 -0.2707097 -0.15410379 -0.78005542  0.7501369 -0.2572368
9  0.11876486  0.57032817  0.05273567 -0.07342510 -0.2680916 -0.20423267  1.01159180  0.3732047 -0.3841573
          V27          V28 Amount Class
1  0.133558377 -0.021053053 149.62     0
2 -0.008983099  0.014724169   2.69     0
3 -0.055352794 -0.059751841 378.66     0
6  0.253844225  0.081080257   3.67     0
7  0.034507430  0.005167769   4.99     0
9  0.011747356  0.142404330  93.20     0
>
```

5.2 Number of fraudulent and non-fraudulent transactions in the dataset

```
> table(creditcard_data$Class)

     0      1
284315    492
```

We see that there are 492 fraudulent transactions among the total set of 284,807 rows.

5.3 Variance of amount

```
> var(creditcard_data$Amount)
[1] 62560.07
```

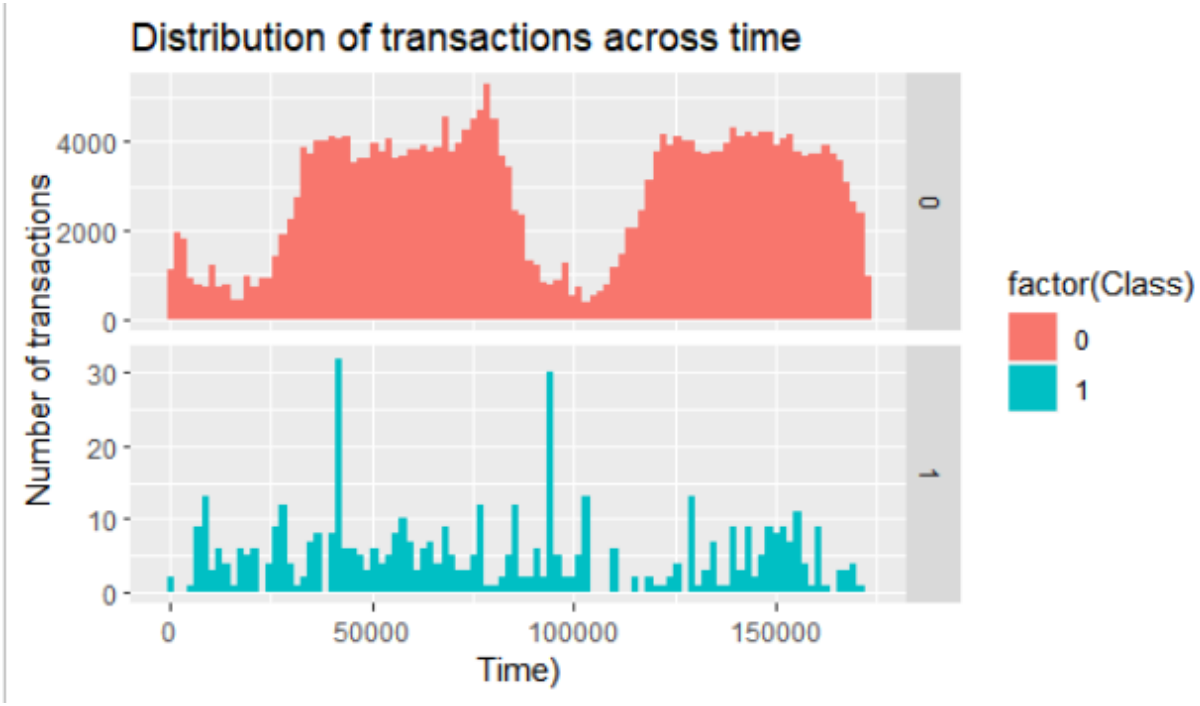
5.4 Standard deviation of amount

```
> sd(creditcard_data$Amount)
[1] 250.1201
```

5.5 View summary of amount

```
summary(creditcard_data$Amount)
   Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
   0.00      5.60     22.00     88.35     77.17 25691.16
```

5.6 Distribution of transactions across time -Transaction versus time

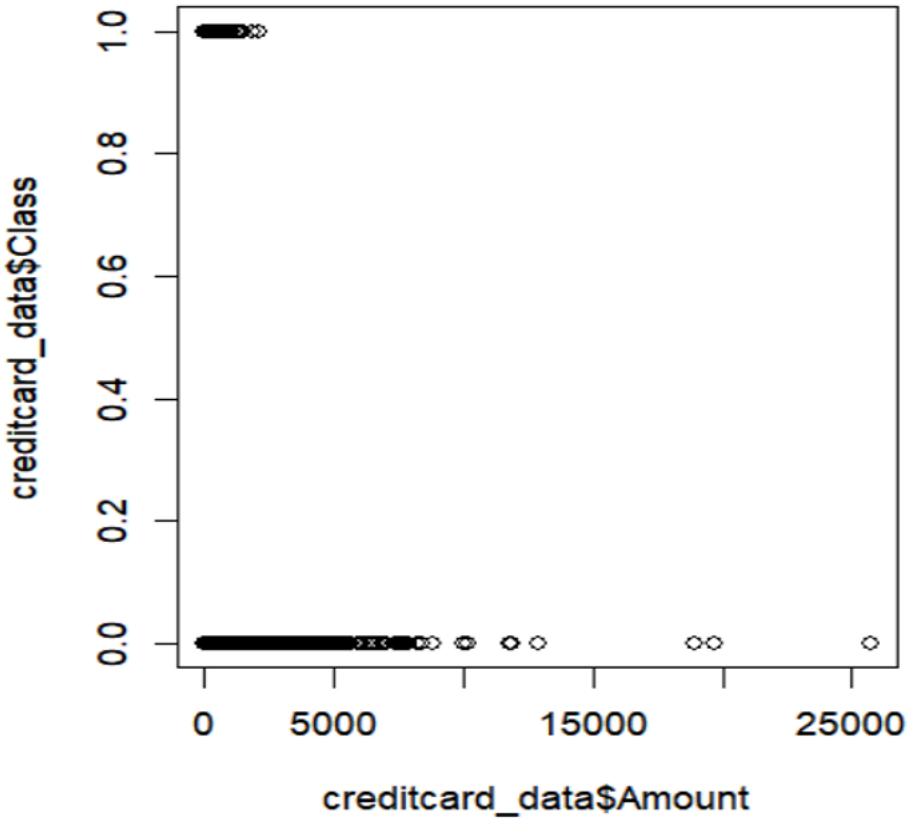


We see that time doesn't influence much in detecting fraudulent transactions.

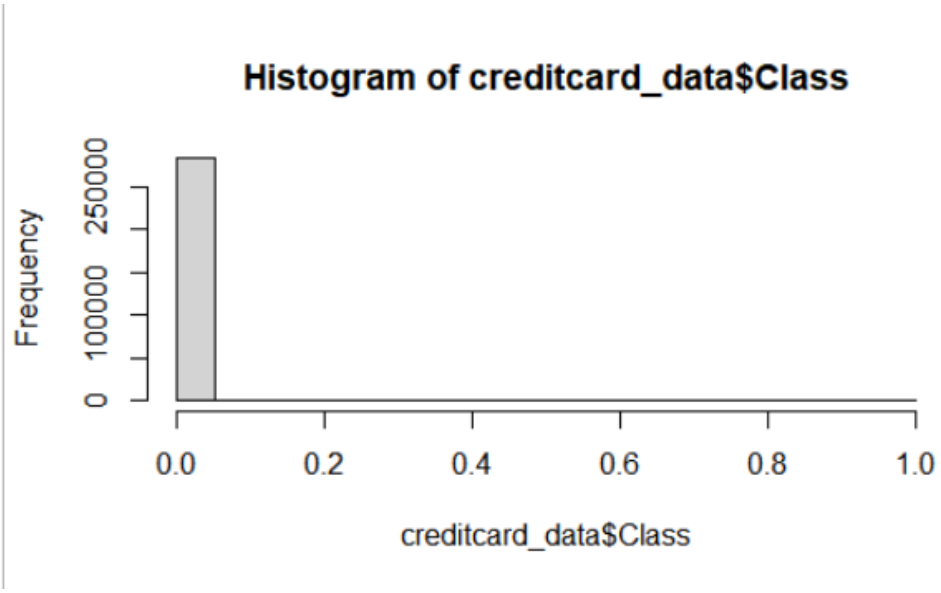
5.7 Plot of Amount and Class

In order of see the range of amount for fraudulent transaction, we have plotted the amount and Class.

We see that the fraudulent transactions fall under shorter amounts.

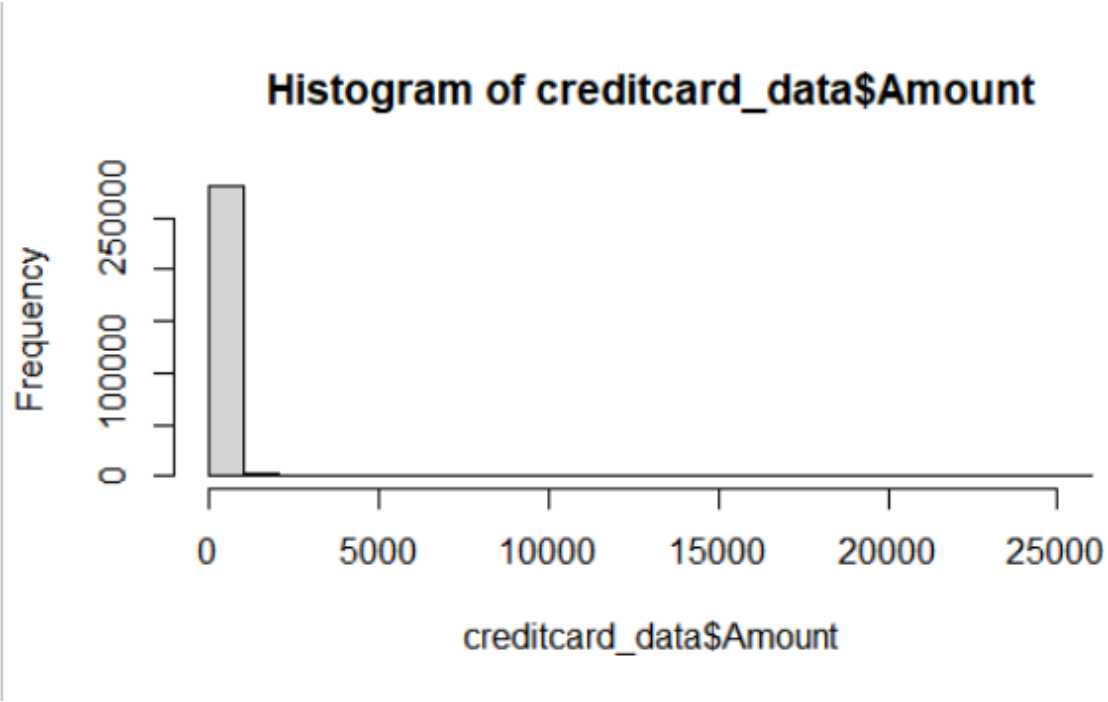


5.8 Histogram of Class

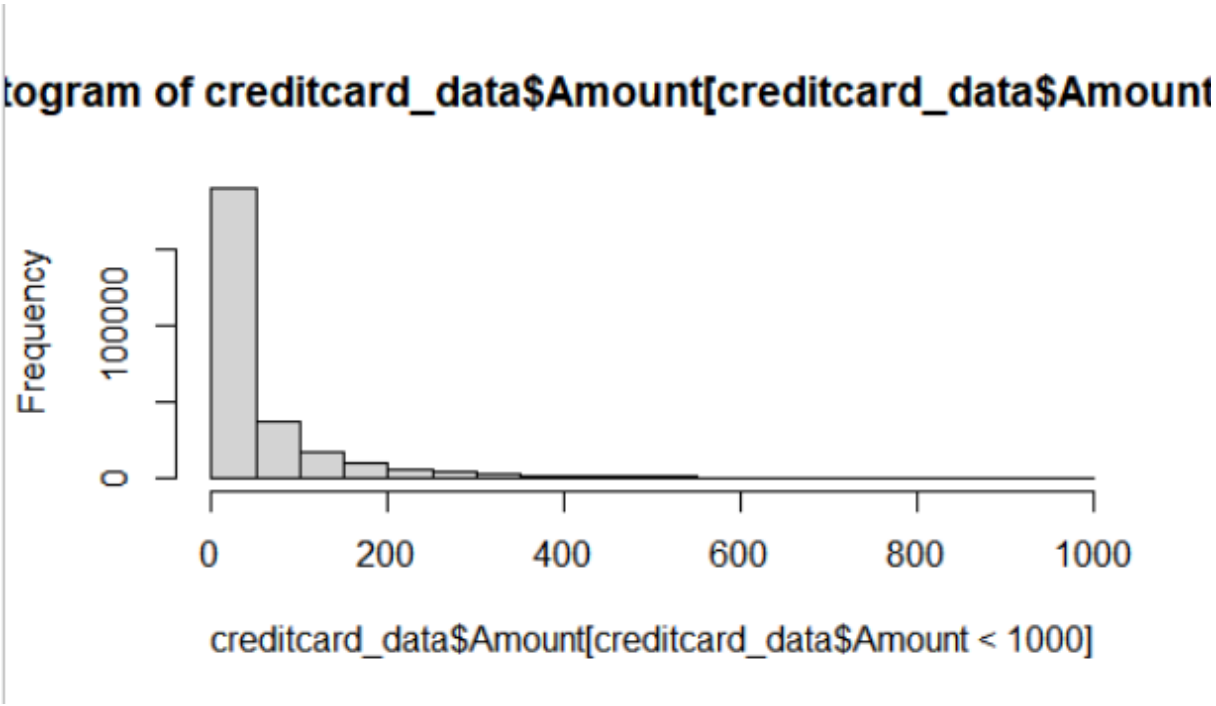


We see that the frequency of fraud transactions are less.

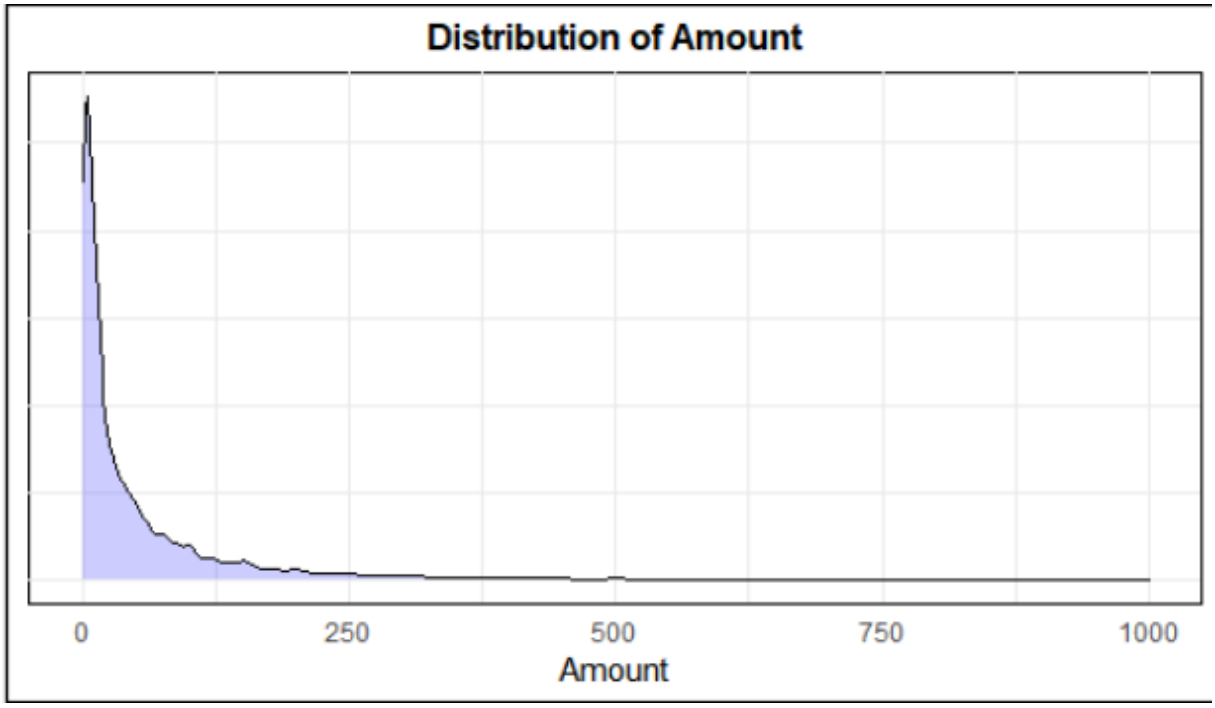
5.9 Histogram of Amount



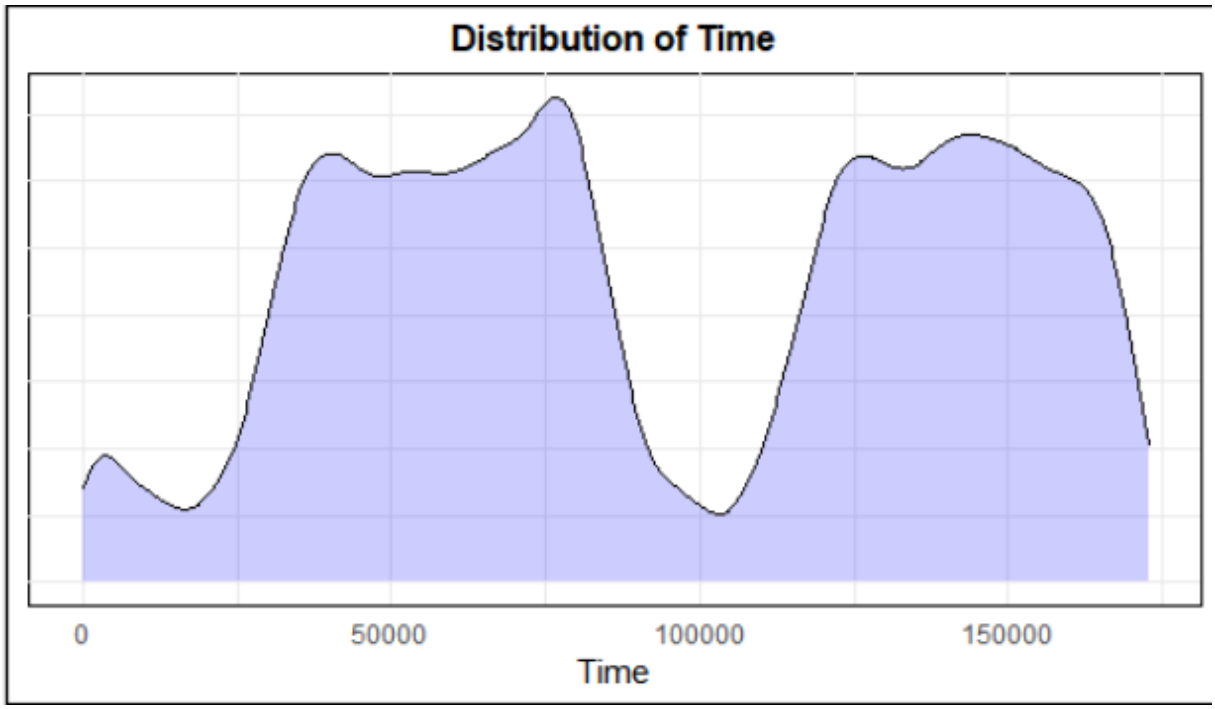
For more clarity ,we will take a lesser range of amount, say Amount <1000.



The below plot gives us a much better clarity on the distribution of amount.

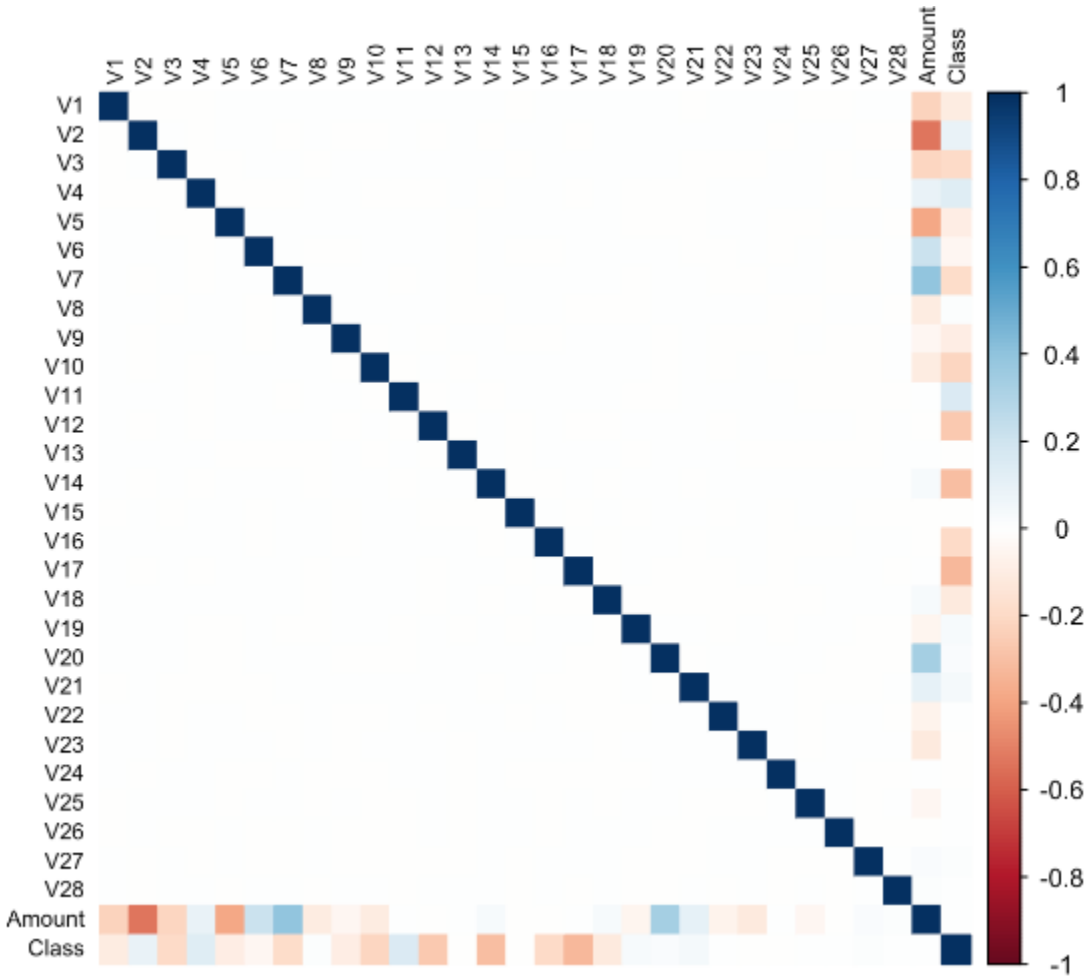


5.10 Plot showing distribution of Transaction time and Amount



5.11 Plot showing correlation of variables

Below plot shows us the correlation between all the variables and amount and class and see if there are any variables that corelate with each other. We use the cor() function for this.



From the above graph, we can see that most of the features are not correlated. In fact, all the anonymous variables are independent to each other.

6 EVALUATION METRIC

In this project we have used 2 metrics.

1. ROC curve

2. Confusion matrix.

ROC curves are used to characterize the sensitivity/specificity. We use the area under the ROC to measure the accuracy of our model.

Confusion Matrix gives us the predictions against actual values. We use two dimensional matrices here We get a picture of the specificity and sensitivity of our model using Confusing Matrix.

Confusion Matrix can be explained as below:

		Actual Values	
		1	0
Predicted Values	1	True Positive	False Positive
	0	False Positive	True Negative

1 – Positive

0 -Negative

For each of the 3 models, I have included screenshots of the Confusion Matrix and this gives us an idea of how the model performs.

7 DATA MODELING APPROACHES- BUILDING THE MODELS

Now that we have analyzed and visualized the data, we will try 3 different models.

7.1 Model 1: Logical Regression Model

Logistic regression model is a commonly used model to calculate or predict the probability of a binary (yes/no) event occurring

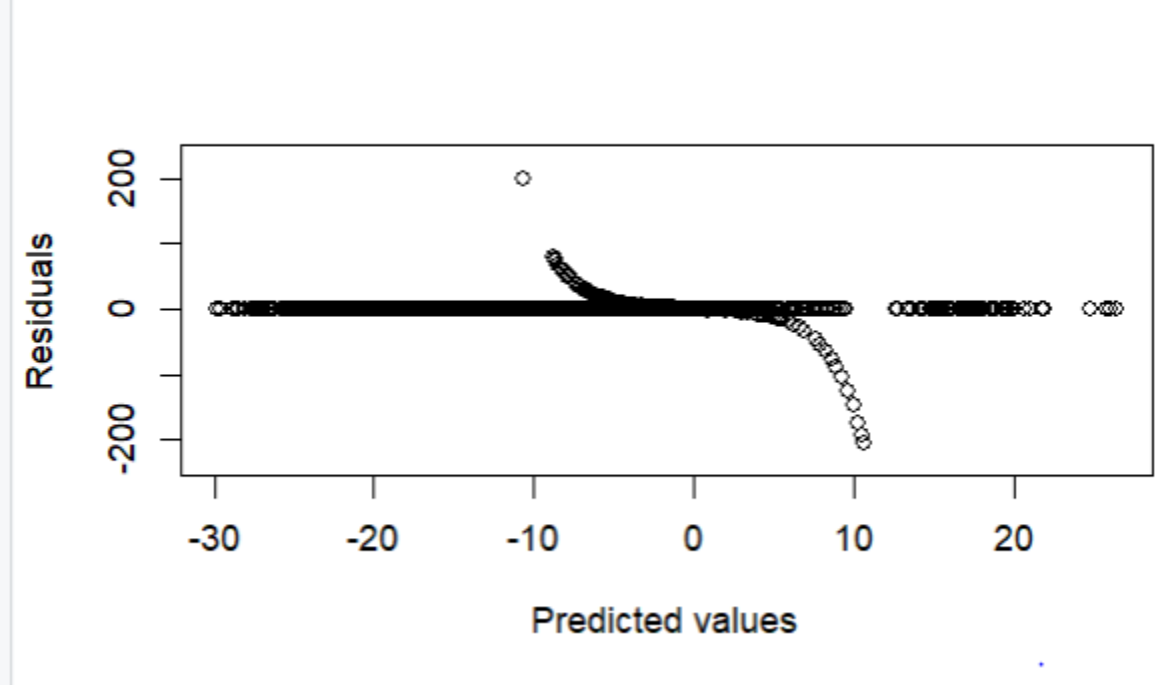
We first build our model on our training set - cc_train_set.

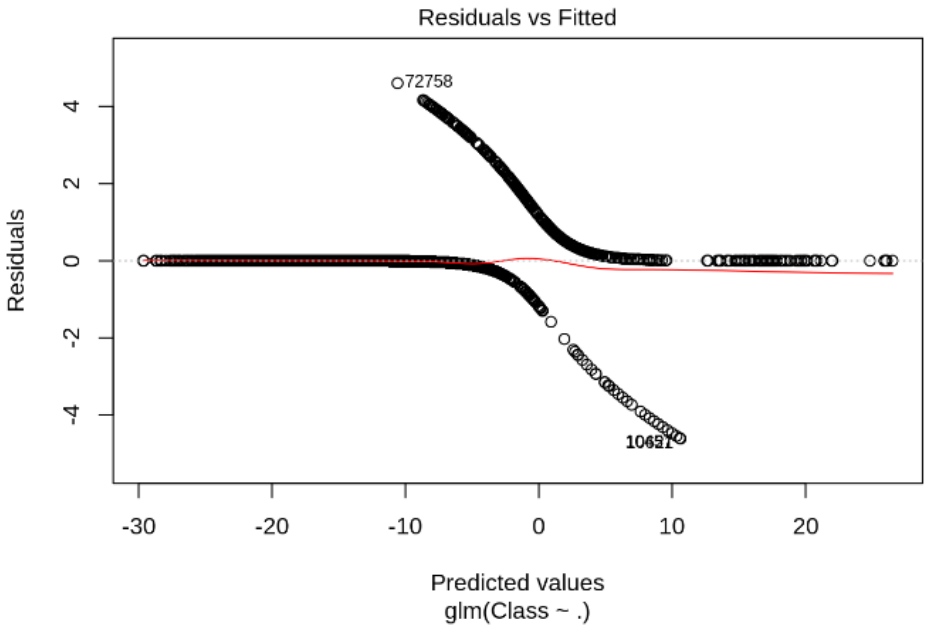
Below is the summary:

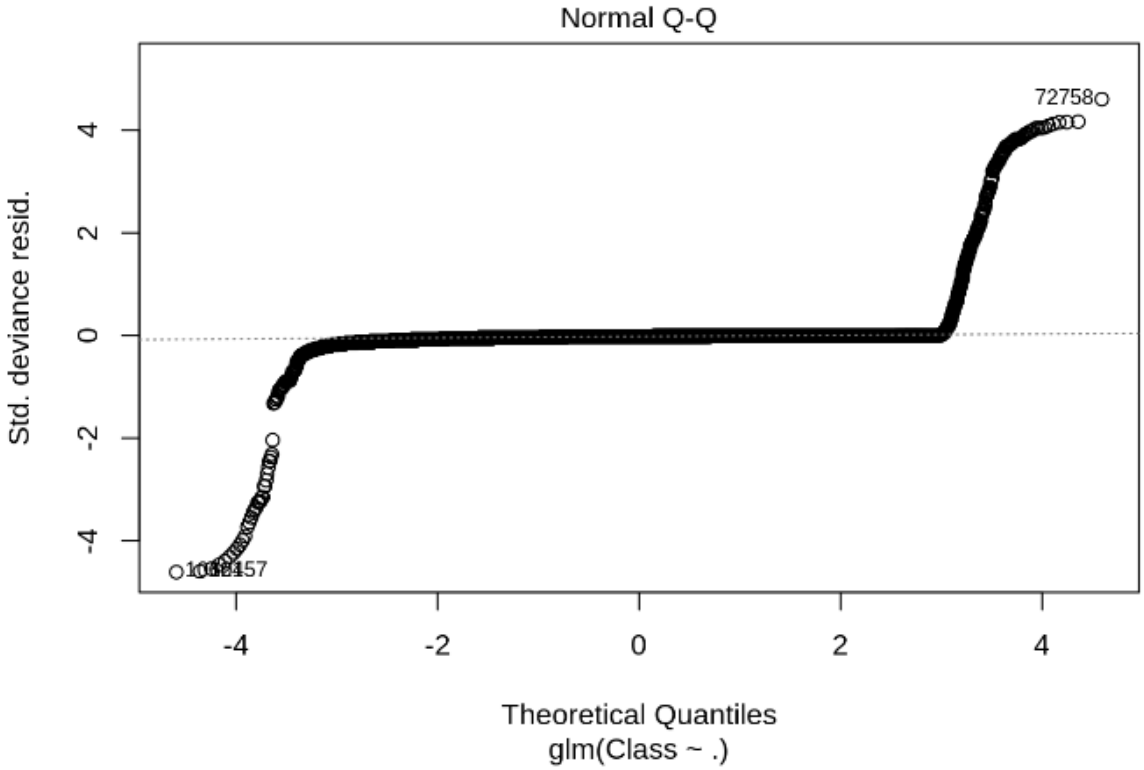
```
Deviance Residuals:
    Min       1Q   Median       3Q      Max
-4.6123  -0.0292  -0.0194  -0.0125   4.6031
```

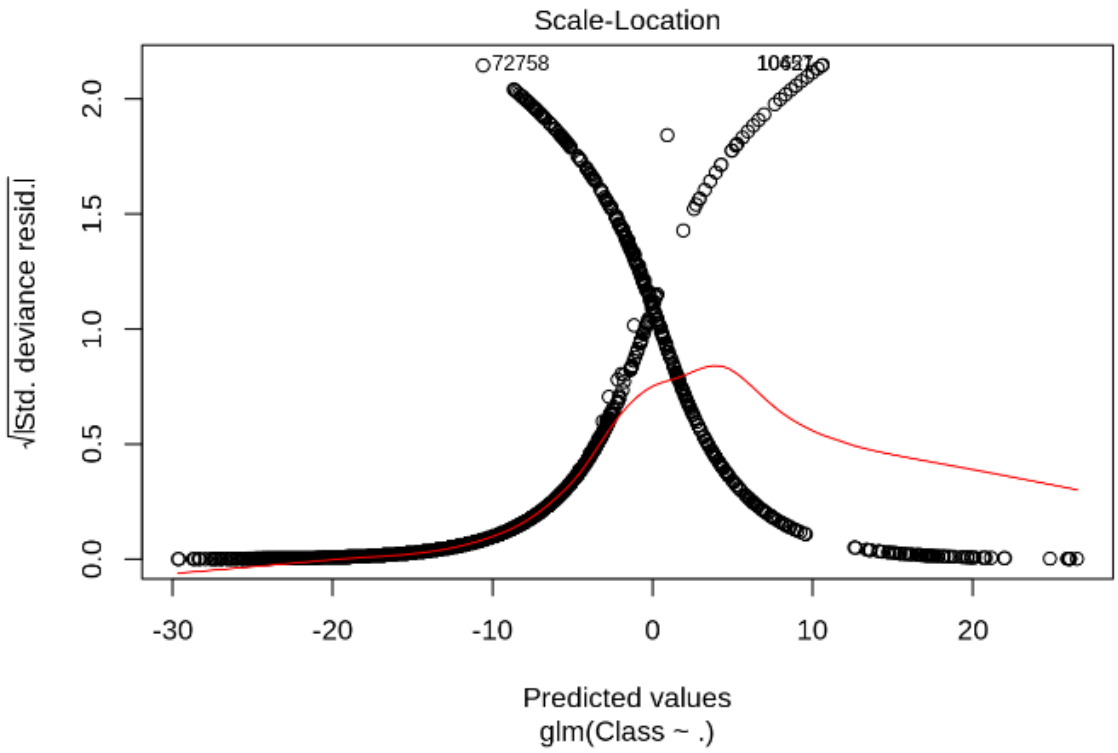
```
(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 5799.1  on 227845  degrees of freedom
Residual deviance: 1790.3  on 227815  degrees of freedom
AIC: 1852.3

Number of Fisher Scoring iterations: 12
```

On plotting the result, we get the below:

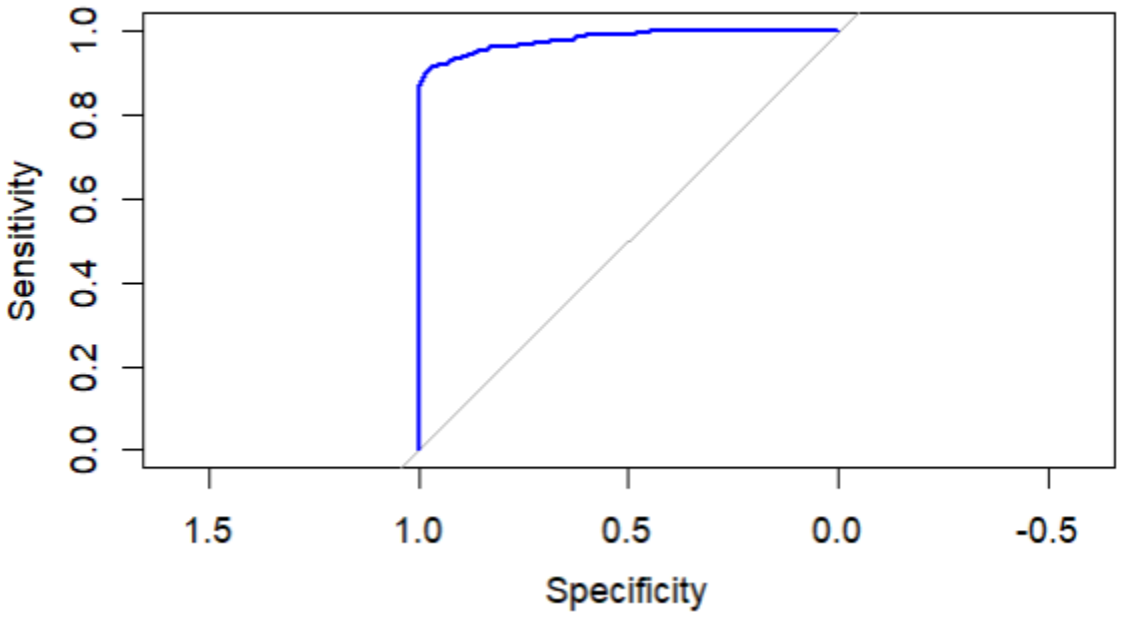



Residuals vs Fitted
72758
10621
Residuals
4
2
0
-2
-4
-30
-20
-10
0
10
20
Predicted values
glm(Class ~ .)


Normal Q-Q
72758
Std. deviance resid.
4
2
0
-2
-4
-4
-2
0
2
4
Theoretical Quantiles
glm(Class ~ .)



When we plot the ROC curve, we get the below:



We get the below as area under curve:

```
Data: model1_predictions in 227452 controls (cc_train_set$Class 0) < 394 cases
 (cc_train_set$Class 1).
Area under the curve: 0.9777
>
```

Now, let us apply this model in test set and see the Confusion Matrix.

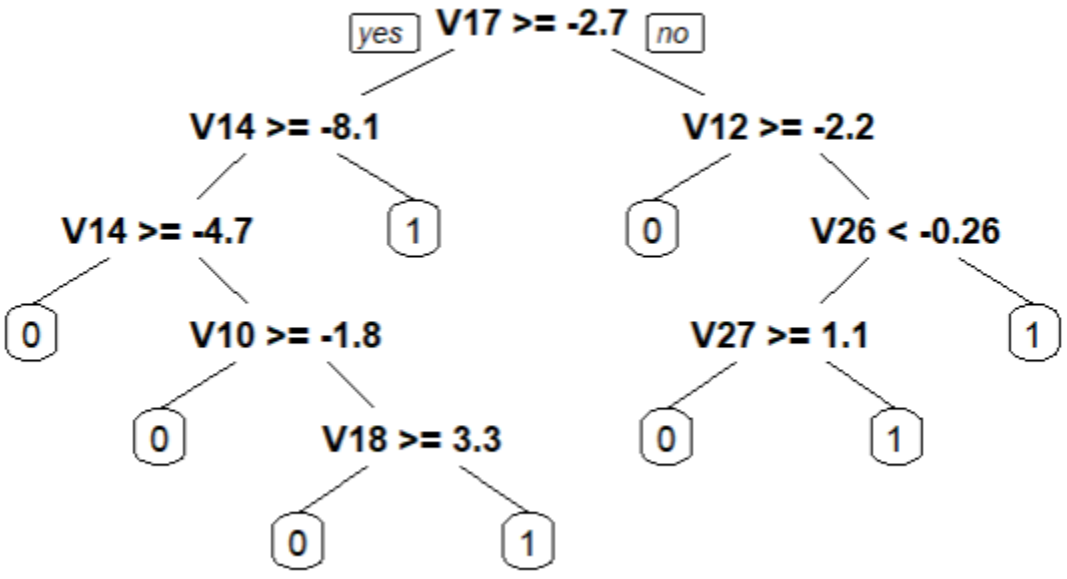
```
> table(cc_test_set$Class, pred)
   pred
        0     1
  0 56856     7
  1    42    56
>
```

We have 56856 true positives, 7 false positives,42 false negatives and 56true negatives.

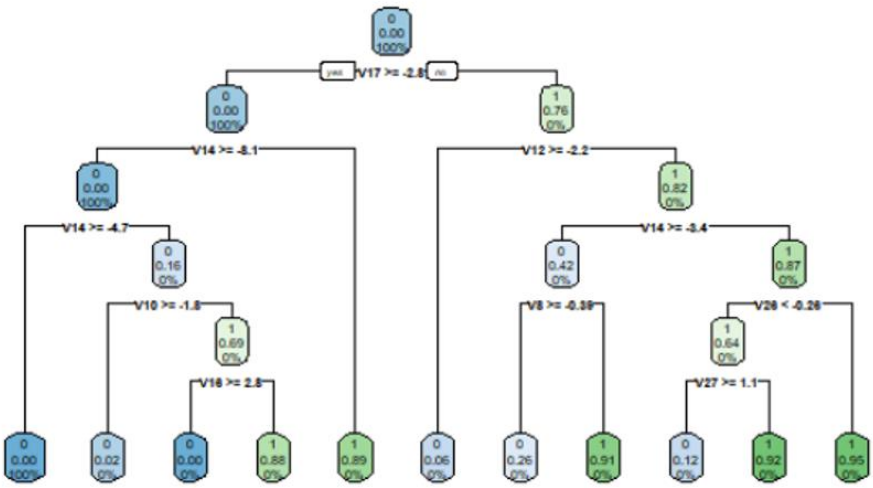
7.2 Model 2: Decision tree model

Decision Trees are useful supervised Machine learning algorithms that have the ability to perform both regression and classification tasks. It is characterized by nodes and branches, where the tests on each attribute are represented at the nodes, the outcome of this procedure is represented at the branches and the class labels are represented at the leaf nodes.

We have used the rpart and rpart.plot libraries in this model. We first apply this model to our train set. The result is as below:



When we apply this model to the test set, we get the below:



The Confusion Matrix is as below:

```
> confusionMatrix(cc_test_set$Class, predicted_val)
Confusion Matrix and Statistics

          Reference
Prediction     0     1
         0 56843    20
         1    21    77

               Accuracy : 0.9993
                 95% CI : (0.999, 0.9995)
    No Information Rate : 0.9983
    P-Value [Acc > NIR] : 1.066e-10

                  Kappa : 0.7894

 Mcnemar's Test P-Value : 1

            Sensitivity : 0.9996
            Specificity : 0.7938
         Pos Pred Value : 0.9996
         Neg Pred Value : 0.7857
             Prevalence : 0.9983
         Detection Rate : 0.9979
   Detection Prevalence : 0.9983
      Balanced Accuracy : 0.8967

       'Positive' Class : 0

>
```

We have 56843 true positives, 20 false positives, 21 false negatives and 77 true negatives.

We see that the number of false negatives has reduced in this model.

7.3 Model 3: Random Forest Model

Random forest model is classification algorithm consisting of many decision trees. It uses bagging and feature randomness when building each individual tree to try to create an uncorrelated forest of trees whose prediction by committee is more accurate than that of any individual tree.

We will first build this model on our train set, and then apply it on the test set.

On plotting the result, we get the below:

The Confusion Matrix is as below:

```
Confusion Matrix and Statistics

          Reference
Prediction     0     1
         0 56856     7
         1    50    48

               Accuracy : 0.999
                 95% CI : (0.9987, 0.9992)
    No Information Rate : 0.999
    P-Value [Acc > NIR] : 0.6395

                  Kappa : 0.627

 Mcnemar's Test P-Value : 2.651e-08

            Sensitivity : 0.9991
            Specificity : 0.8727
         Pos Pred Value : 0.9999
         Neg Pred Value : 0.4898
             Prevalence : 0.9990
         Detection Rate : 0.9982
   Detection Prevalence : 0.9983
      Balanced Accuracy : 0.9359

       'Positive' Class : 0
```

8 RESULTS

From the 3 models, the results can be summarized as below:

1. Logistic Regression

Area under curve: 0.9777

True Positives: 56856

True Negatives: 56

2. Decision Tree Model

True Positives: 56843

True Negatives: 77

3. Random Forest Model

True Positives: 56856

True Negatives: 48

The mean of Class field of all the 3 models are shown below:

```
> Model1
[1] 0.9991398
> Model2
[1] 0.9992802
> Model3
[1] 0.9989993
```

9 CONCLUSION

After trying several models, we found that Random model is apparently good in predicting fraud case. We got maximum accuracy with this model with maximum number of True Positives. In this case as specificity is more important, this model would be recommended.

10 REFERENCES

Confusion Matrix in R | A Complete Guide | DigitalOcean

https://www.statology.org/scale-function-in-r/

https://data-flair.training/blogs/data-science-machine-learning-project-credit-card-fraud-detection/

x----------x