CS 571 Data Preparation and Analysis

Project Report

On

Credit Card Fraud Detection

By

Name of Student	CWID
Pradyothan Govrineni	A20438408
Uttam Kotadiya	A20480934
Harsh Dungrani	A20514062
Sohaib Syed	A20439074

Under The Supervision

of

Prof. Jawahar Panchal

College of Computing

Illinois Institute of Technology

Chicago, Illinois

April 2023

CONTENT:

I. Abstract:

This project focuses on data preparation and management to develop a robust system for identifying fraudulent credit card transactions. The objective is to prevent financial loss due to fraudulent activities that cost billions of dollars annually. The Card Transactions dataset will serve as the primary source of data for the development of the system. The project will utilize a step-by-step approach, starting with exploratory data analysis, data cleaning, data transformation, and feature engineering to ensure the quality and accuracy of the data.

During the data preparation phase, the project will explore various data cleaning and transformation techniques, including handling missing values, dealing with outliers, and normalization to ensure the data is consistent and ready for the subsequent steps. The project will also use feature engineering to create new features that can better represent the underlying patterns in the data.

After data preparation, the project will use statistical analysis and visualization techniques to gain insights into the data and better understand the characteristics of fraudulent transactions. This understanding will help to identify the most appropriate modeling techniques.

The final stage of the project will focus on model evaluation and testing, where the developed system will undergo validation and testing using unseen data to ensure its robustness and reliability. The project aims to optimize the accuracy of the system while minimizing the number of false positives by understanding the characteristics of fraudulent transactions.

The success of the project will be determined by the interpretability of the developed system and its ability to identify fraudulent credit card transactions with high accuracy. The project aims to provide a comprehensive understanding of the data preparation and management process and its practical applications in preventing financial loss due to fraudulent credit card transactions.

II. Overview:

The main aim of this project is to develop a robust machine learning classifier system that is capable of accurately detecting fraudulent credit card transactions using the R programming language. To accomplish this goal, the project will follow a comprehensive and structured approach that involves various steps such as data preparation, exploratory data analysis, model training, validation, testing, and evaluation.

The Card Transactions dataset, which contains data on credit card transactions, will be used to build and test the classifier system. The project will leverage different machine learning algorithms, including logistic regression and decision trees, to identify the most effective approach for the task. Additionally, the classifier system will be designed to achieve high accuracy while minimizing the number of false positives.

The project will provide valuable knowledge and experience in the field of classification by implementing machine learning algorithms. Furthermore, the developed classifier system will have practical applications in preventing financial losses due to fraudulent credit card transactions. Specifically, it will enable credit card companies and financial institutions to detect fraudulent transactions in real-time and take appropriate measures to prevent further financial losses.

The success of the project will be determined by the accuracy and interpretability of the developed classifier system, as well as its ability to identify fraudulent credit card transactions with high confidence. Additionally, the project will provide insights into the importance of data preparation and management in machine learning projects, and the critical role played by exploratory data analysis in understanding data patterns and relationships.

III. Data Description:

For the project we considered two different datasets.

Dataset 1: https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud

Dataset 2: https://www.kaggle.com/datasets/mishra5001/credit-card

The first dataset consists of 31 columns and 284807 rows. The datatypes of the attributes in the dataset are 30 numeric and 1 integer type attributes. It contains only numeric input variables which are the result of a PCA transformation.

```
'data.frame':   284807 obs. of  31 variables:
 $ Time  : num  0 0 1 1 2 2 4 7 7 9 ...
 $ V1    : num  -1.36 1.192 -1.358 -0.966 -1.158 ...
 $ V2    : num  -0.0728 0.2662 -1.3402 -0.1852 0.8777 ...
 $ V3    : num  2.536 0.166 1.773 1.793 1.549 ...
 $ V4    : num  1.378 0.448 0.38 -0.863 0.403 ...
 $ V5    : num  -0.3383 0.06 -0.5032 -0.0103 -0.4072 ...
 $ V6    : num  0.4624 -0.0824 1.8005 1.2472 0.0959 ...
 $ V7    : num  0.2396 -0.0788 0.7915 0.2376 0.5929 ...
 $ V8    : num  0.0987 0.0851 0.2477 0.3774 -0.2705 ...
 $ V9    : num  0.364 -0.255 -1.515 -1.387 0.818 ...
 $ V10   : num  0.0908 -0.167 0.2076 -0.055 0.7531 ...
 $ V11   : num  -0.552 1.613 0.625 -0.226 -0.823 ...
 $ V12   : num  -0.6178 1.0652 0.0661 0.1782 0.5382 ...
 $ V13   : num  -0.991 0.489 0.717 0.508 1.346 ...
 $ V14   : num  -0.311 -0.144 -0.166 -0.288 -1.12 ...
 $ V15   : num  1.468 0.636 2.346 -0.631 0.175 ...
 $ V16   : num  -0.47 0.464 -2.89 -1.06 -0.451 ...
 $ V17   : num  0.208 -0.115 1.11 -0.684 -0.237 ...
 $ V18   : num  0.0258 -0.1834 -0.1214 1.9658 -0.0382 ...
 $ V19   : num  0.404 -0.146 -2.262 -1.233 0.803 ...
 $ V20   : num  0.2514 -0.0691 0.525 -0.208 0.4085 ...
 $ V21   : num  -0.01831 -0.22578 0.248 -0.1083 -0.00943 ...
 $ V22   : num  0.27784 -0.63867 0.77168 0.00527 0.79828 ...
 $ V23   : num  -0.11 0.101 0.909 -0.19 -0.137 ...
 $ V24   : num  0.0669 -0.3398 -0.6893 -1.1756 0.1413 ...
 $ V25   : num  0.129 0.167 -0.328 0.647 -0.206 ...
 $ V26   : num  -0.189 0.126 -0.139 -0.222 0.502 ...
 $ V27   : num  0.13356 -0.00898 -0.05535 0.06272 0.21942 ...
 $ V28   : num  -0.0211 0.0147 -0.0598 0.0615 0.2152 ...
 $ Amount: num  149.62 2.69 378.66 123.5 69.99 ...
 $ Class : int  0 0 0 0 0 0 0 0 0 0 ...
```

Fig 1: Structure of dataset 1

The dataset consists of 284807 rows where the classification of the credit card fraud is imbalanced with non-fraudulent transactions having 284315 samples while fraudulent transactions is only 492 samples which is what can be expected in real life as there would be more legitimate transactions as compared to illegitimate transactions.

'Target Class Distribution: 0: Non-Fraudulent, 1: Fraudulent'

0	1
284315	492

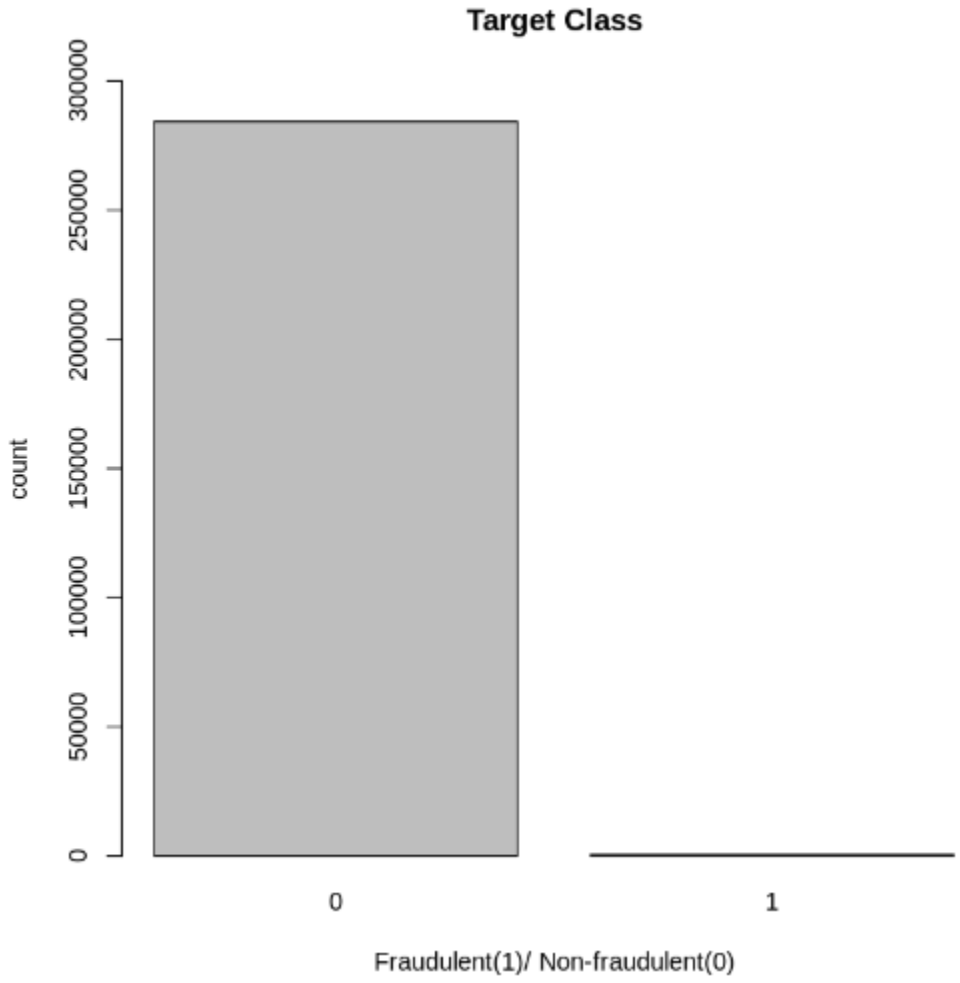


Fig 2: Target Class Countplot for dataset 1

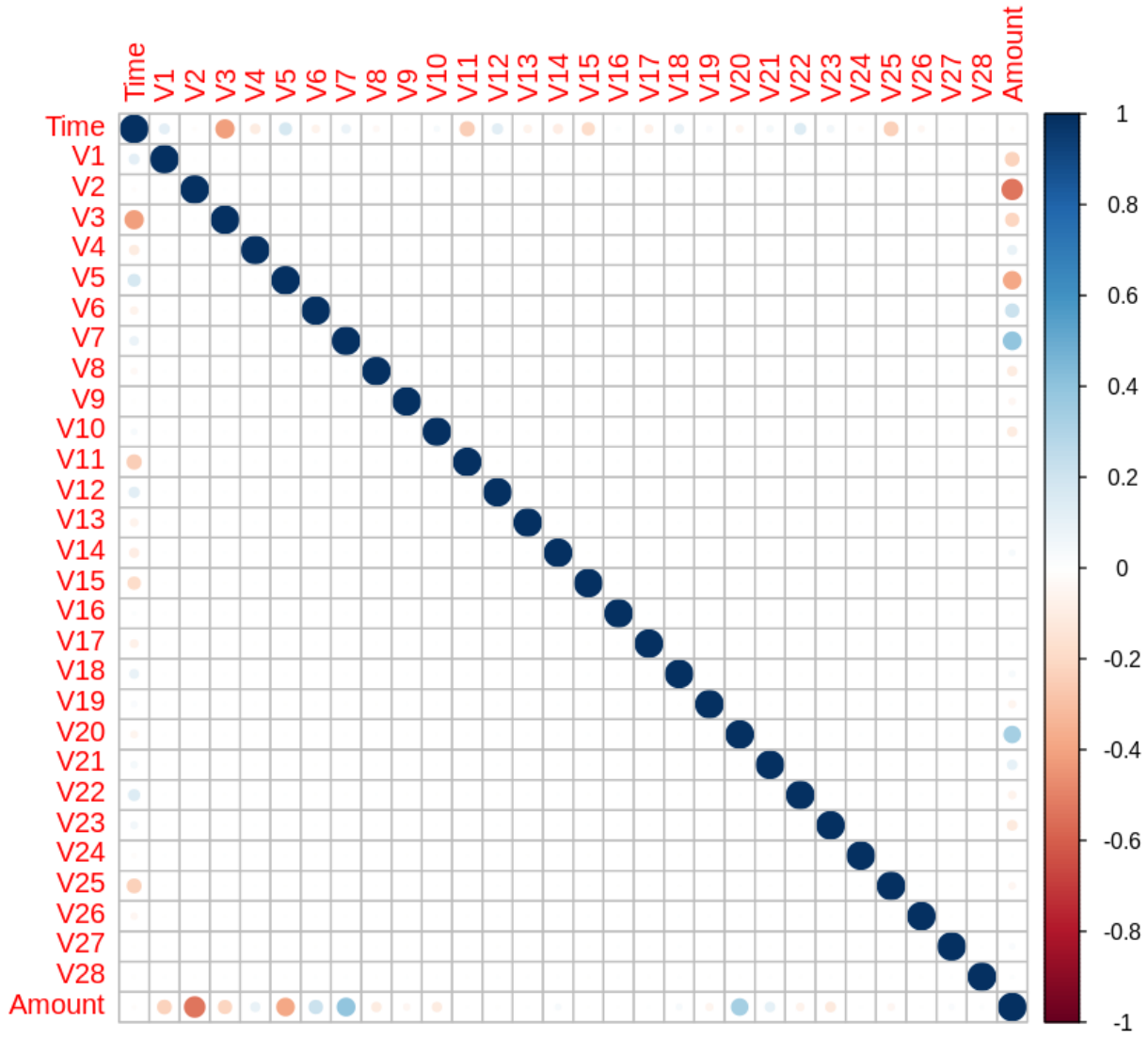


Fig 3: Correlation map between the predictors in dataset 1

The correlation map between the predictors above helps us in getting insights about how the different variables are dependent on each other. Amount and V2 are the variables which seem to have some negative correlation between each other while the rest of the correlations among them are weakly correlated.

Time	V1	V2	V3	V4	V5	V6	V7	V8	V9	V10	V11	V12	V13
0.0	-56.4075096	-72.71572756	-48.3255894	-5.68317120	-113.74330671	-26.1605059	-43.55724157	-73.21671846	-13.43406632	-24.58826244	-4.79747346	-18.6837146	-5.79188121
54201.5	-0.9203734	-0.59854991	-0.8903648	-0.84864012	-0.69159707	-0.7682956	-0.55407588	-0.20862974	-0.64309757	-0.53542573	-0.76249420	-0.4055715	-0.64853930
84692.0	0.0181088	0.06548556	0.1798463	-0.01984653	-0.05433583	-0.2741871	0.04010308	0.02235804	-0.05142873	-0.09291738	-0.03275735	0.1400326	-0.01356806
139320.5	1.3156417	0.80372387	1.0271955	0.74334129	0.61192644	0.3985649	0.57043607	0.32734586	0.59713903	0.45392345	0.73959341	0.6182380	0.66250496
172792.0	2.4549300	22.05772899	9.3825584	16.87534403	34.80166588	73.3016255	120.58949395	20.00720837	15.59499461	23.74513612	12.01891318	7.8483921	7.12688296

A matrix: 5 × 13 of type dbl

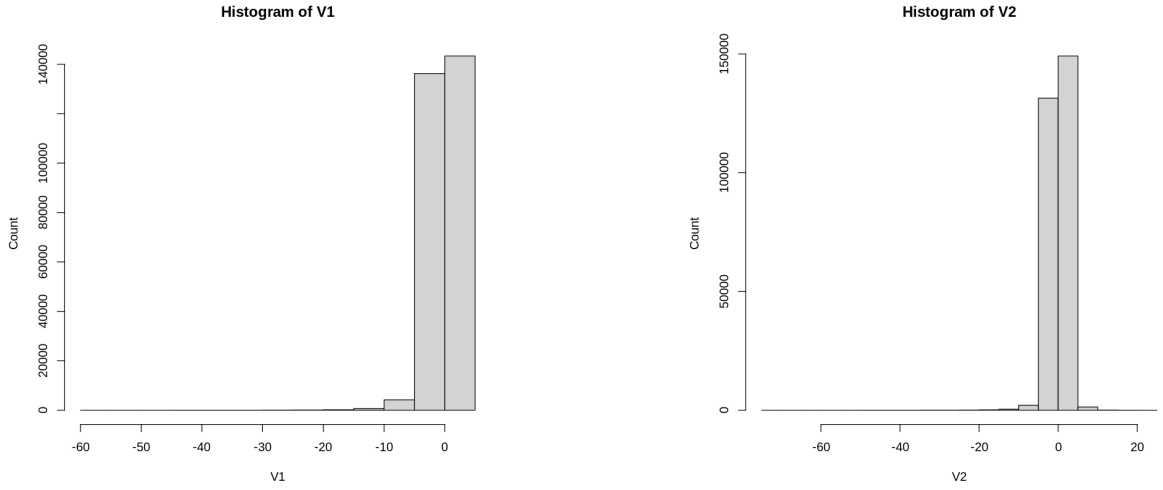
V14	V15	V16	V17	V18	V19	V20	V21	V22	V23	V24	V25	V26
-19.21432549	-4.49894468	-14.12985452	-25.16279937	-9.498745921	-7.213527430	-54.49772049	-34.83038214	-10.933143698	-44.80773520	-2.83662692	-10.2953971	-2.60455055
-0.42557401	-0.58288428	-0.46803677	-0.48374831	-0.498849799	-0.456298919	-0.21172136	-0.22839495	-0.542350373	-0.16184635	-0.35458614	-0.3171451	-0.32698393
0.05060132	0.04807155	0.06641332	-0.06567575	-0.003636312	0.003734823	-0.06248109	-0.02945017	0.006781943	-0.01119293	0.04097606	0.0165935	-0.05213911
0.49314985	0.64882081	0.52329631	0.39967498	0.500806747	0.458949356	0.13304084	0.18637720	0.528553635	0.14764206	0.43952660	0.3507156	0.24095217
10.52676605	8.87774160	17.31511152	9.25352625	5.041069185	5.591971427	39.42090425	27.20283916	10.503090090	22.52841169	4.58454914	7.5195887	3.51734561

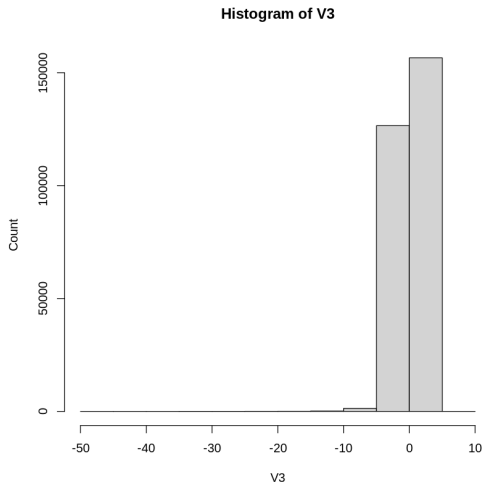
A matrix: 5 × 3 of type dbl

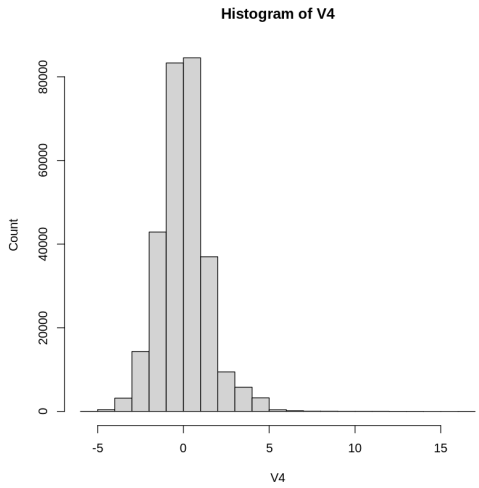
V27	V28	Amount
-22.565679321	-15.43008391	0.000
-0.070839529	-0.05295979	5.600
0.001342146	0.01124383	22.000
0.091045120	0.07827995	77.165
31.612198106	33.84780782	25691.160

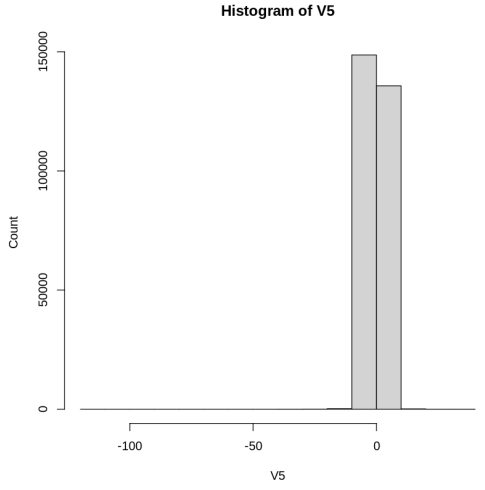
Fig 4: 5-number summary of each feature in dataset 1

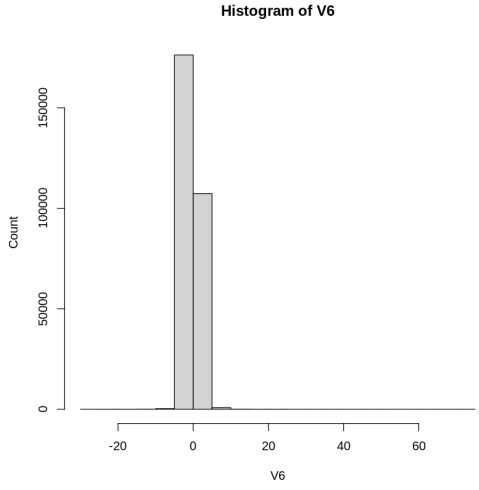
Unfortunately, due to confidentiality issues, they cannot provide the original features and more background information about the data. Features V1, V2, … V28 are the principal components obtained with PCA, the only features which have not been transformed with PCA are 'Time' and 'Amount'. Feature 'Time' contains the seconds elapsed between each transaction and the first transaction in the dataset. The histogram plots of the features in dataset 1 show the trend of variables that have PCA applied on them are distributed near to or around zero.

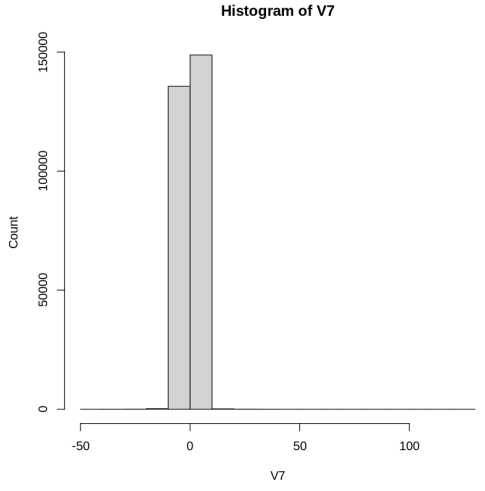


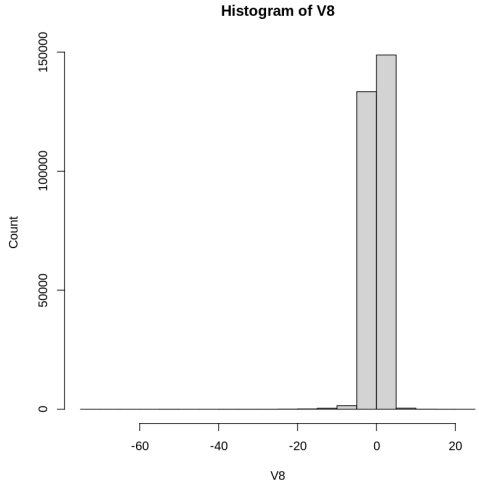

Histogram of V3
Count
0
50000
100000
150000
-50
-40
-30
-20
-10
0
10
V3

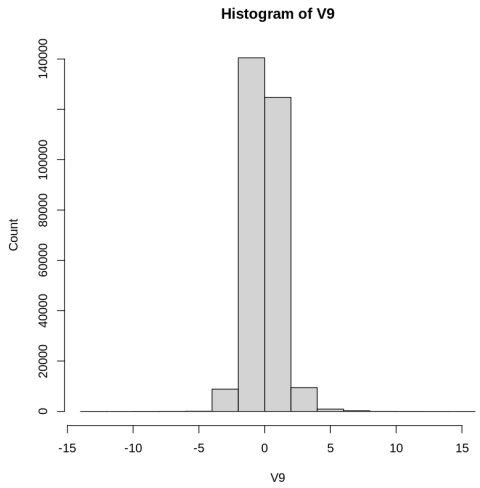

Histogram of V4
Count
0
20000
40000
60000
80000
-5
0
5
10
15
V4

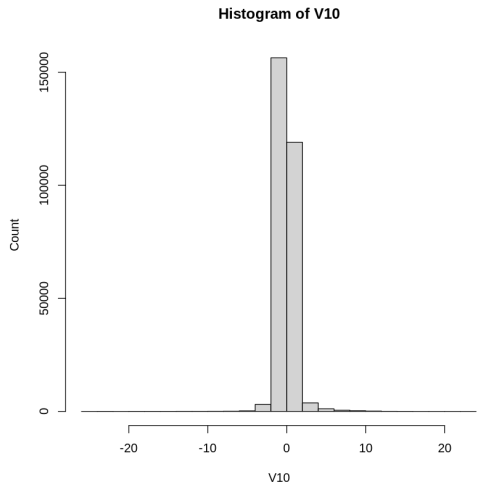

Histogram of V5
Count
0
50000
100000
150000
-100
-50
0
V5

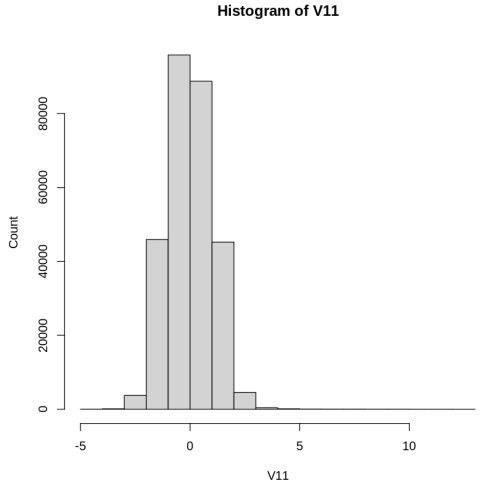

Histogram of V6
Count
0
50000
100000
150000
-20
0
20
40
60
V6

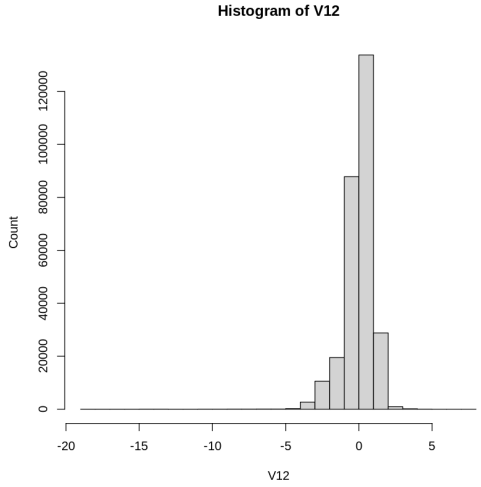

Histogram of V7
Count
0
50000
100000
150000
-50
0
50
100
V7

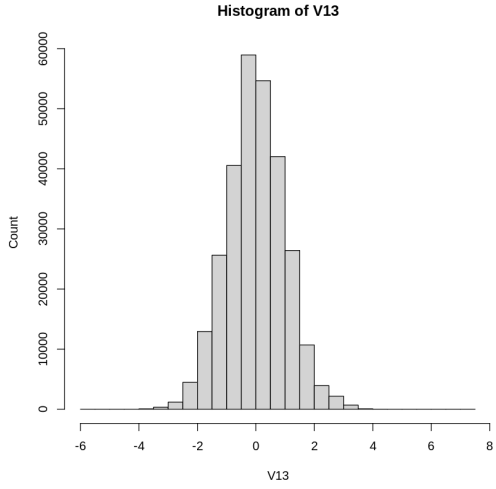

Histogram of V8
Count
0
50000
100000
150000
-60
-40
-20
0
20
V8

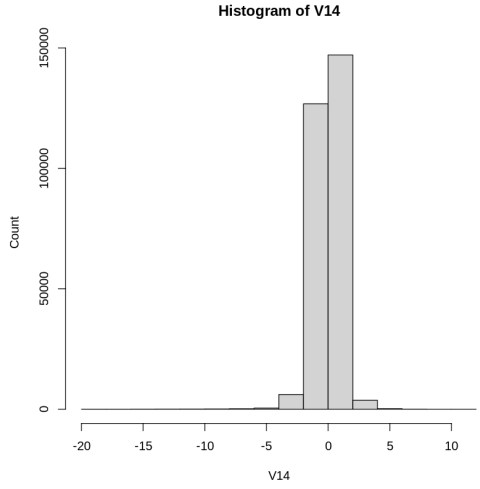

Histogram of V9
Count
0
20000
40000
60000
80000
100000
140000
-15
-10
-5
0
5
10
15
V9

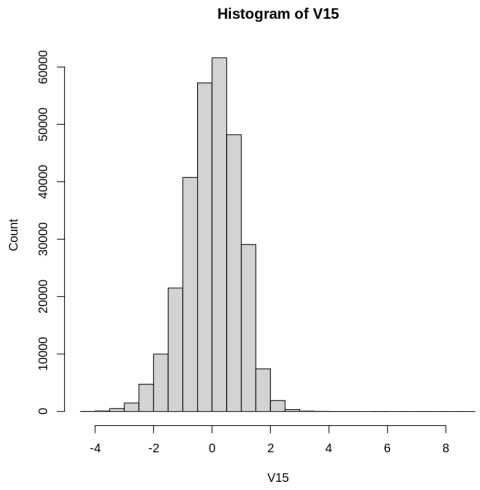

Histogram of V10
Count
0
50000
100000
150000
-20
-10
0
10
20
V10

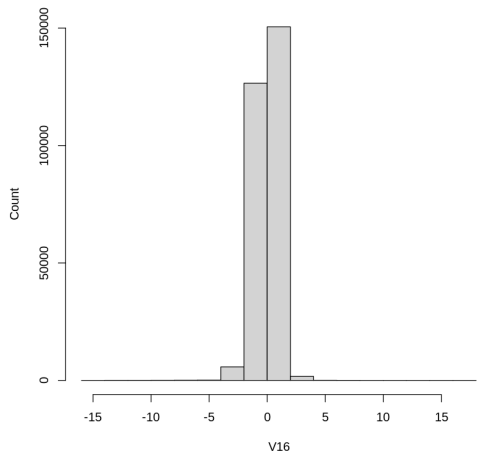

Histogram of V11
Count
0
20000
40000
60000
80000
-5
0
5
10
V11

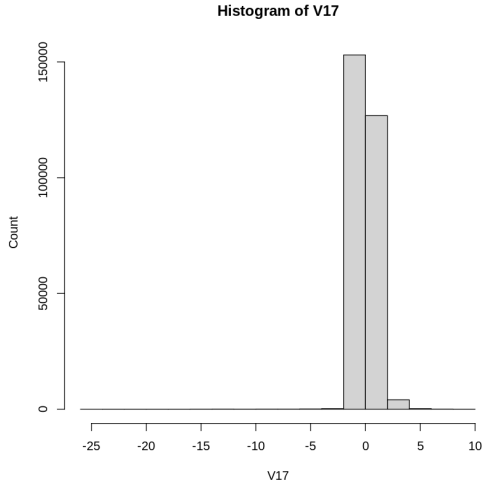

Histogram of V12
Count
0
20000
40000
60000
80000
100000
120000
-20
-15
-10
-5
0
5
V12

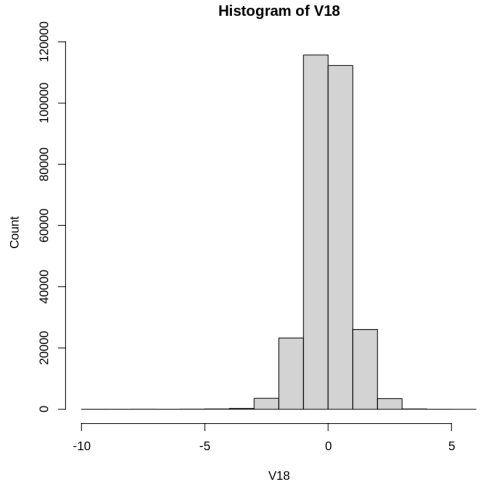

Histogram of V13
Count
0
10000
20000
30000
40000
50000
60000
-6
-4
-2
0
2
4
6
8
V13

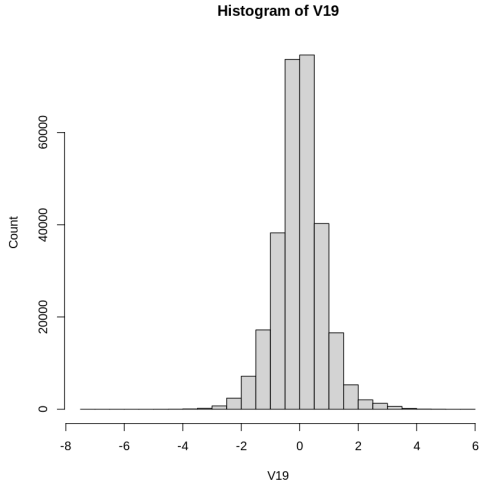

Histogram of V14
Count
0
50000
100000
150000
-20
-15
-10
-5
0
5
10
V14

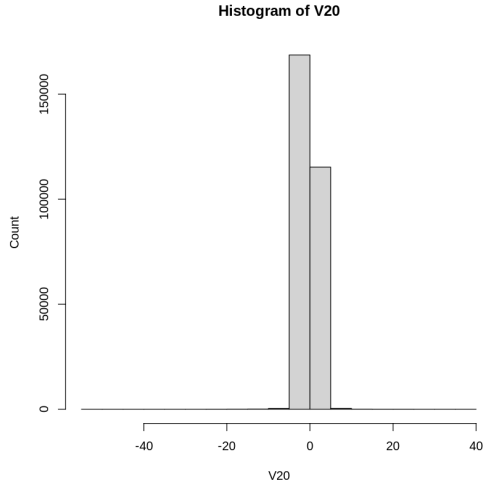

Histogram of V15
Count
0
10000
20000
30000
40000
50000
60000
-4
-2
0
2
4
6
8
V15

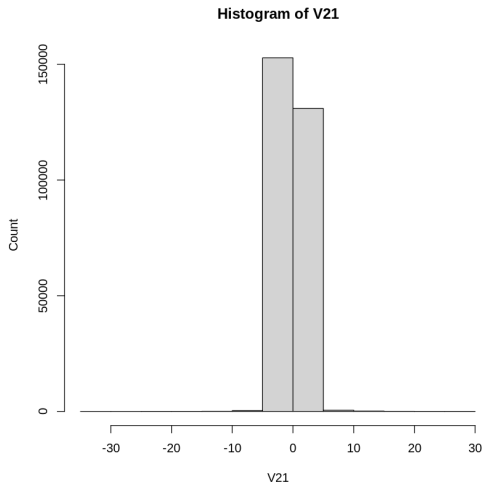

Histogram of V16
Count
0
50000
100000
150000
-15
-10
-5
0
5
10
15
V16

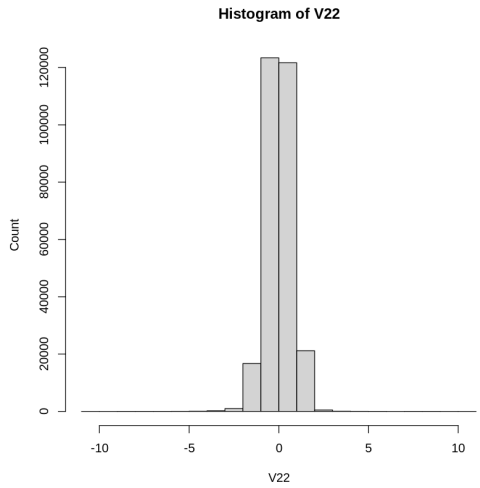

Histogram of V17
Count
0
50000
100000
150000
-25
-20
-15
-10
-5
0
5
10
V17

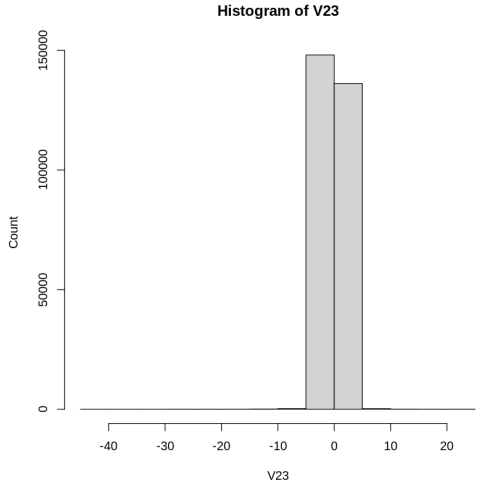

Histogram of V18
Count
0
20000
40000
60000
80000
100000
120000
-10
-5
0
5
V18

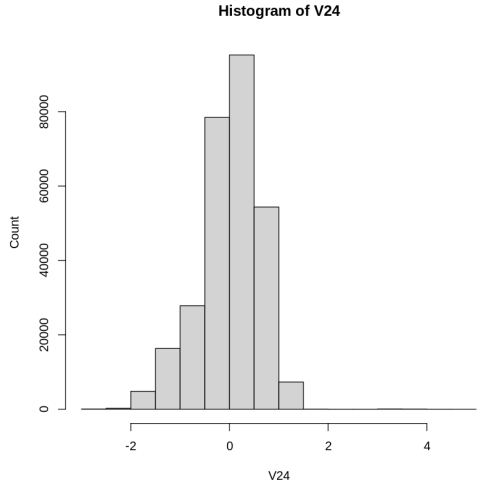

Histogram of V19
Count
0
20000
40000
60000
-8
-6
-4
-2
0
2
4
6
V19

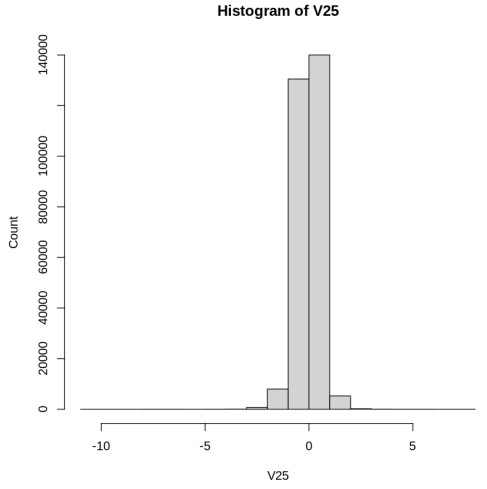

Histogram of V20
Count
0
50000
100000
150000
-40
-20
0
20
40
V20

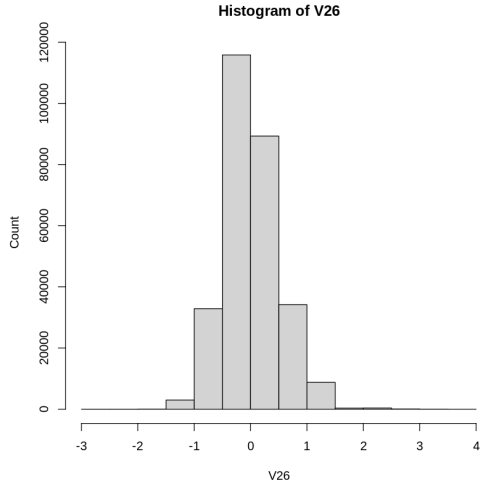

Histogram of V21
Count
0
50000
100000
150000
-30
-20
-10
0
10
20
30
V21


Histogram of V22
Count
0
20000
40000
60000
80000
100000
120000
-10
-5
0
5
10
V22


Histogram of V23
Count
0
50000
100000
150000
-40
-30
-20
-10
0
10
20
V23


Histogram of V24
Count
0
20000
40000
60000
80000
-2
0
2
4
V24


Histogram of V25
Count
0
20000
40000
60000
80000
100000
140000
-10
-5
0
5
V25


Histogram of V26
Count
0
20000
40000
60000
80000
100000
120000
-3
-2
-1
0
1
2
3
4
V26

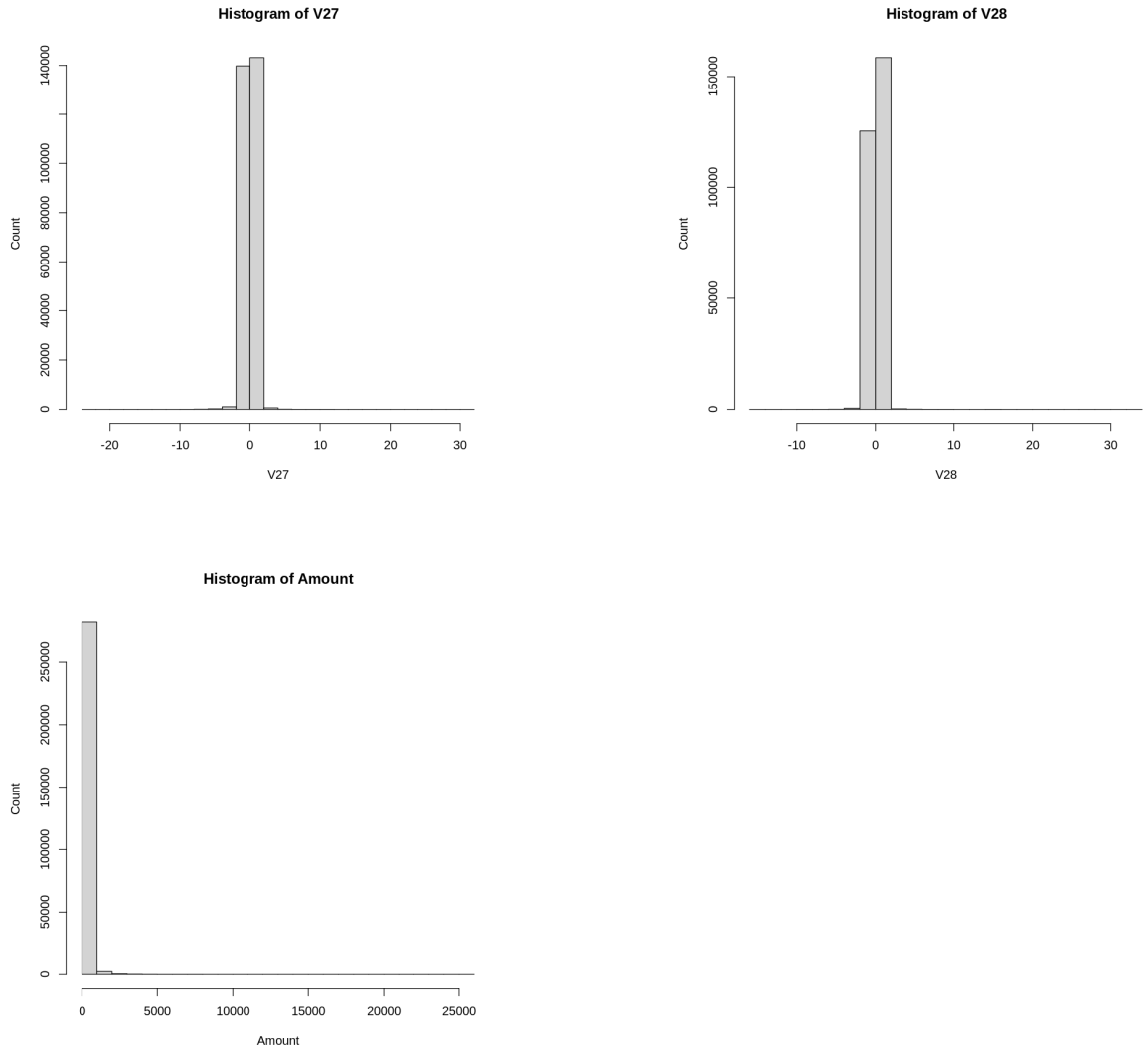


Fig 5: Histogram plots of each feature in dataset 1

The second dataset consists of 121 predictors and 1 target class to perform classification on credit card fraud detection. The dataset consists of 307511 rows where the classification of the credit card fraud is imbalanced with non-fraudulent transactions having 282686 samples while fraudulent transactions are 24825 only samples .The dataset consists of attributes which are of different data types. The distribution of which is as follows: 16 character type, 41 integer type and 65 numeric type attributes.

'Target Class Distribution: 0: Non-Fraudulent, 1: Fraudulent'

```
     0      1
282686  24825
```

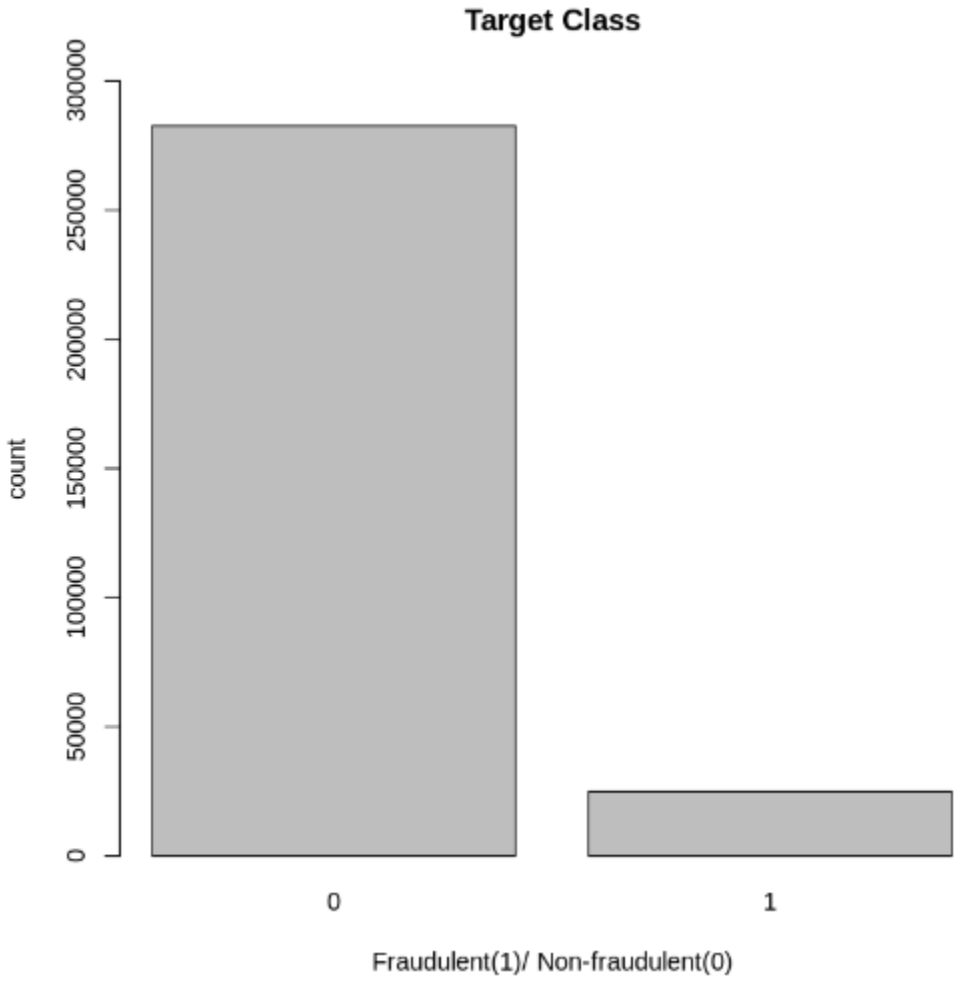


Fig 6: Target Class Countplot for dataset 1

The dataset consisted of quite a lot NA values and the number of those in each column is listed below.

SK_ID_CURR	0
TARGET	0
NAME_CONTRACT_TYPE	0
CODE_GENDER	0
FLAG_OWN_CAR	0
FLAG_OWN_REALTY	0
CNT_CHILDREN	0
AMT_INCOME_TOTAL	0
AMT_CREDIT	0
AMT_ANNUITY	12
AMT_GOODS_PRICE	278
NAME_TYPE_SUITE	0
NAME_INCOME_TYPE	0
NAME_EDUCATION_TYPE	0
NAME_FAMILY_STATUS	0
NAME_HOUSING_TYPE	0
REGION_POPULATION_RELATIVE	0
DAYS_BIRTH	0
DAYS_EMPLOYED	0
DAYS_REGISTRATION	0
DAYS_ID_PUBLISH	0
OWN_CAR_AGE	202929
FLAG_MOBIL	0
FLAG_EMP_PHONE	0
FLAG_WORK_PHONE	0
FLAG_CONT_MOBILE	0
FLAG_PHONE	0
FLAG_EMAIL	0
OCCUPATION_TYPE	0
CNT_FAM_MEMBERS	2
REGION_RATING_CLIENT	0
REGION_RATING_CLIENT_W_CITY	0
WEEKDAY_APPR_PROCESS_START	0
HOUR_APPR_PROCESS_START	0
REG_REGION_NOT_LIVE_REGION	0
REG_REGION_NOT_WORK_REGION	0
LIVE_REGION_NOT_WORK_REGION	0
REG_CITY_NOT_LIVE_CITY	0
REG_CITY_NOT_WORK_CITY	0
LIVE_CITY_NOT_WORK_CITY	0
ORGANIZATION_TYPE	0
EXT_SOURCE_1	173378
EXT_SOURCE_2	660
EXT_SOURCE_3	60965
APARTMENTS_AVG	156061
BASEMENTAREA_AVG	179943
YEARS_BEGINEXPLUATATION_AVG	150007
YEARS_BUILD_AVG	204488
COMMONAREA_AVG	214865
ELEVATORS_AVG	163891
ENTRANCES_AVG	154828
FLOORSMAX_AVG	153020
FLOORSMIN_AVG	208642
LANDAREA_AVG	182590
LIVINGAPARTMENTS_AVG	210199
LIVINGAREA_AVG	154350
NONLIVINGAPARTMENTS_AVG	213514
NONLIVINGAREA_AVG	169682
APARTMENTS_MODE	156061
BASEMENTAREA_MODE	179943
YEARS_BEGINEXPLUATATION_MODE	150007
YEARS_BUILD_MODE	204488
COMMONAREA_MODE	214865
ELEVATORS_MODE	163891
ENTRANCES_MODE	154828
FLOORSMAX_MODE	153020
FLOORSMIN_MODE	208642
LANDAREA_MODE	182590
LIVINGAPARTMENTS_MODE	210199
LIVINGAREA_MODE	154350
NONLIVINGAPARTMENTS_MODE	213514
NONLIVINGAREA_MODE	169682
APARTMENTS_MEDI	156061
BASEMENTAREA_MEDI	179943
YEARS_BEGINEXPLUATATION_MEDI	150007
YEARS_BUILD_MEDI	204488
COMMONAREA_MEDI	214865
ELEVATORS_MEDI	163891
ENTRANCES_MEDI	154828
FLOORSMAX_MEDI	153020
FLOORSMIN_MEDI	208642
LANDAREA_MEDI	182590
LIVINGAPARTMENTS_MEDI	210199
LIVINGAREA_MEDI	154350

```
NONLIVINGAPARTMENTS_MEDI          NONLIVINGAREA_MEDI
                  213514                      169682
      FONDKAPREMONT_MODE             HOUSETYPE_MODE
                       0                           0
          TOTALAREA_MODE          WALLSMATERIAL_MODE
                  148431                           0
     EMERGENCYSTATE_MODE      OBS_30_CNT_SOCIAL_CIRCLE
                       0                        1021
DEF_30_CNT_SOCIAL_CIRCLE      OBS_60_CNT_SOCIAL_CIRCLE
                    1021                        1021
DEF_60_CNT_SOCIAL_CIRCLE        DAYS_LAST_PHONE_CHANGE
                    1021                           1
          FLAG_DOCUMENT_2                 FLAG_DOCUMENT_3
                       0                           0
          FLAG_DOCUMENT_4                 FLAG_DOCUMENT_5
                       0                           0
          FLAG_DOCUMENT_6                 FLAG_DOCUMENT_7
                       0                           0
          FLAG_DOCUMENT_8                 FLAG_DOCUMENT_9
                       0                           0
         FLAG_DOCUMENT_10                FLAG_DOCUMENT_11
                       0                           0
         FLAG_DOCUMENT_12                FLAG_DOCUMENT_13
                       0                           0
         FLAG_DOCUMENT_14                FLAG_DOCUMENT_15
                       0                           0
         FLAG_DOCUMENT_16                FLAG_DOCUMENT_17
                       0                           0
         FLAG_DOCUMENT_18                FLAG_DOCUMENT_19
                       0                           0
         FLAG_DOCUMENT_20                FLAG_DOCUMENT_21
                       0                           0
AMT_REQ_CREDIT_BUREAU_HOUR     AMT_REQ_CREDIT_BUREAU_DAY
                   41519                       41519
AMT_REQ_CREDIT_BUREAU_WEEK     AMT_REQ_CREDIT_BUREAU_MON
                   41519                       41519
 AMT_REQ_CREDIT_BUREAU_QRT     AMT_REQ_CREDIT_BUREAU_YEAR
                   41519                       41519
```

Fig 7:Count of NA values for each column in dataset 2

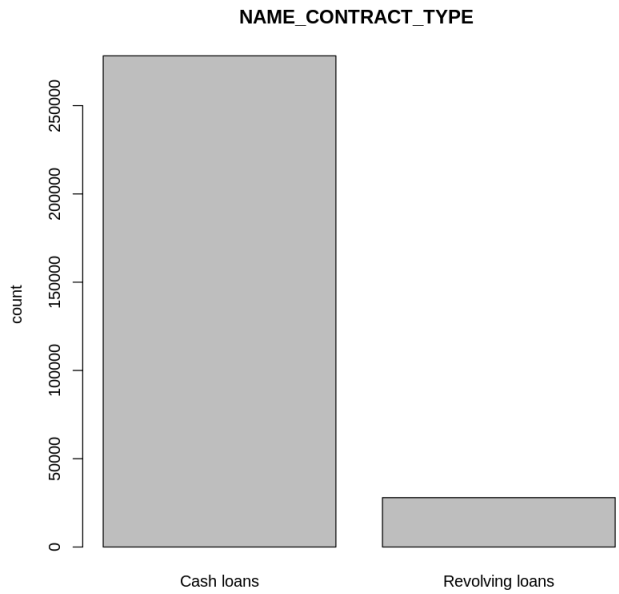


Fig 8: Count plot of feature 'NAME_CONTRACT_TYPE'

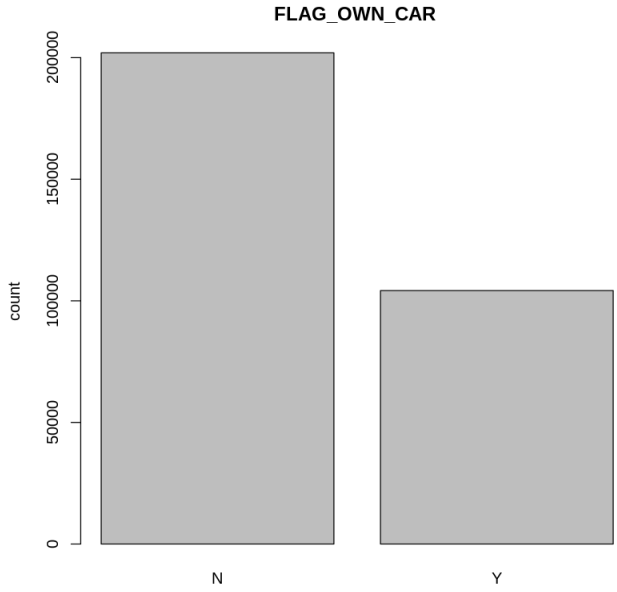


Fig 9: Count plot of feature 'FLAG_OWN_CAR'

Fig 10: Count plot of feature 'FLAG_OWN_REALTY'

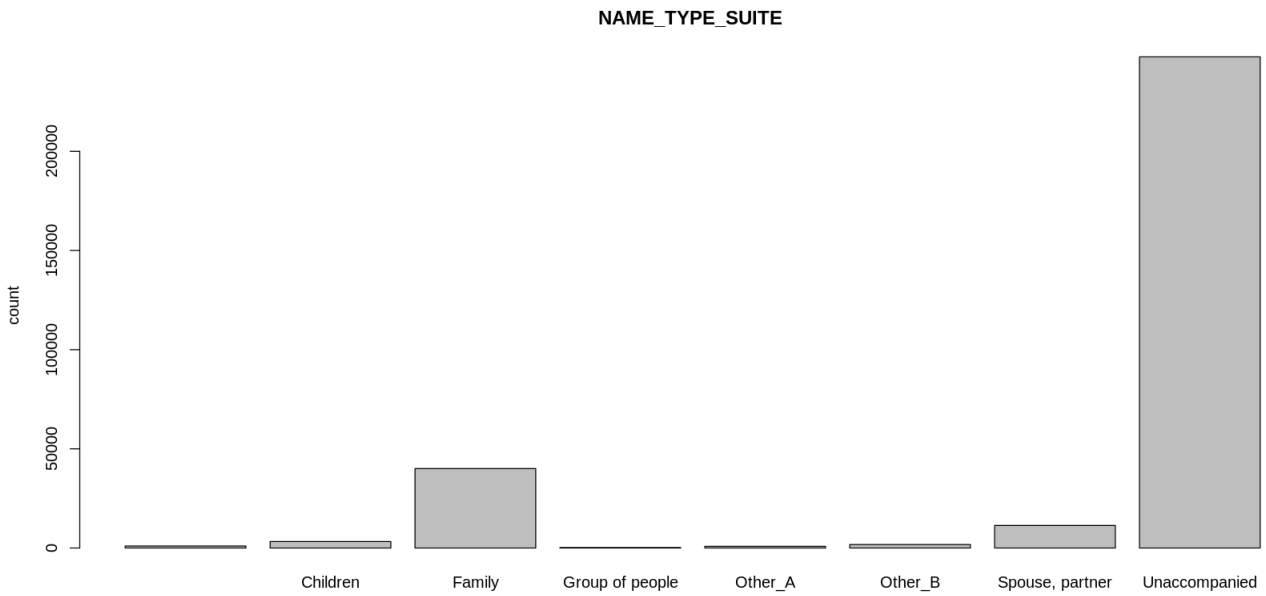


Fig 11: Count plot of feature 'NAME_TYPE_SUITE'

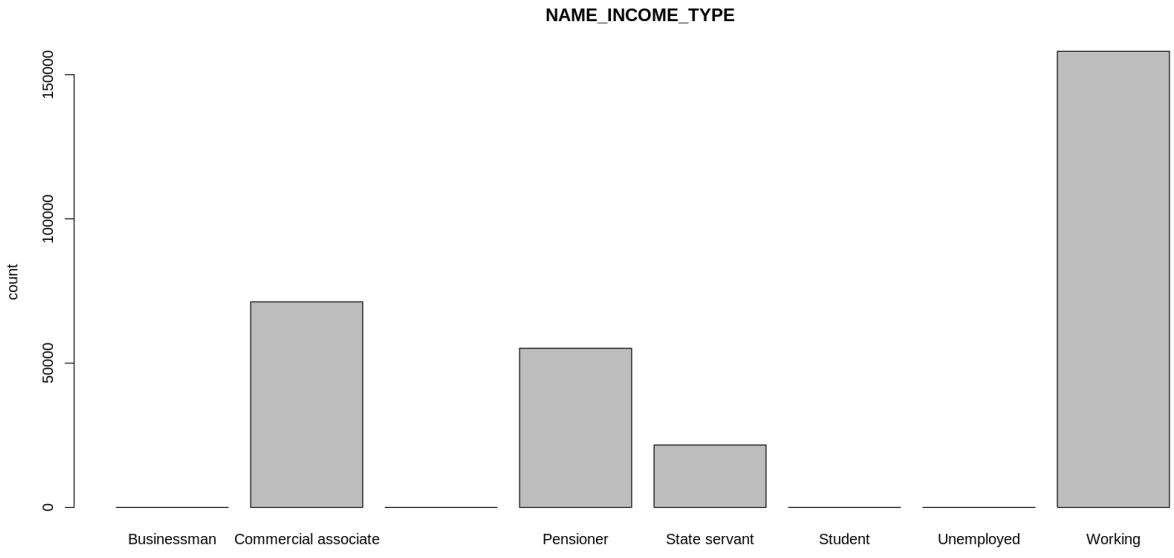


Fig 12: Count plot of feature 'NAME_INCOME_TYPE'

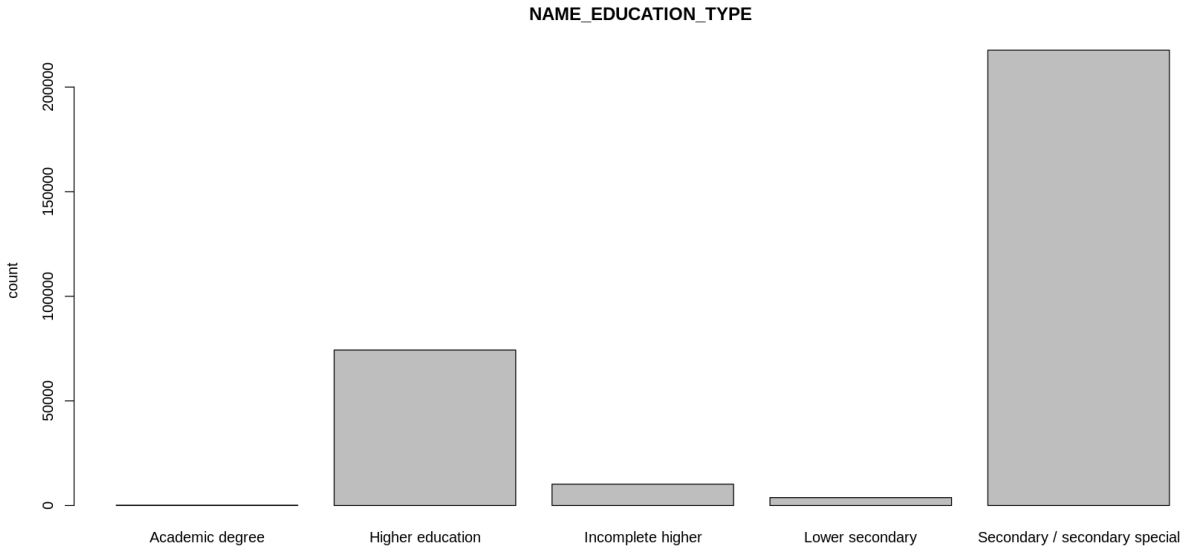


Fig 13: Count plot of feature 'NAME_EDUCATION_TYPE'

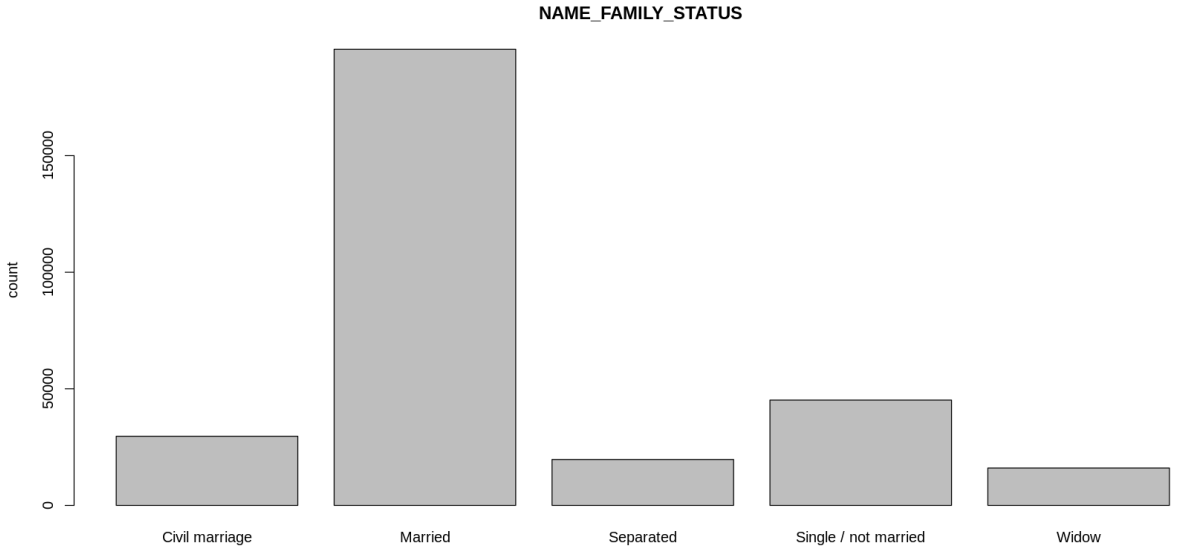


Fig 14: Count plot of feature 'NAME_FAMILY_STATUS'

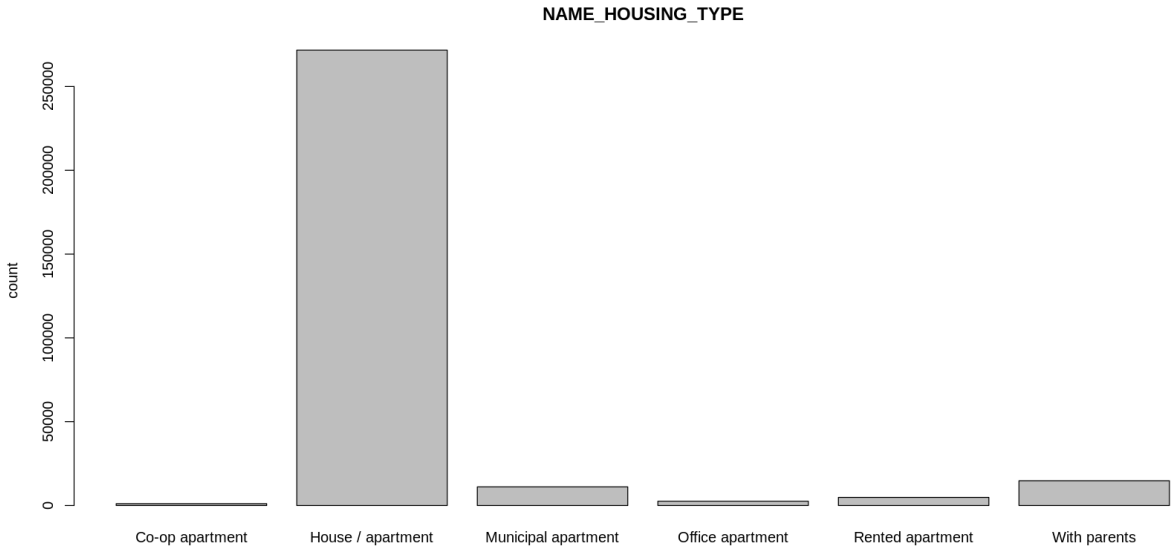


Fig 15: Count plot of feature 'NAME_HOUSING_TYPE'

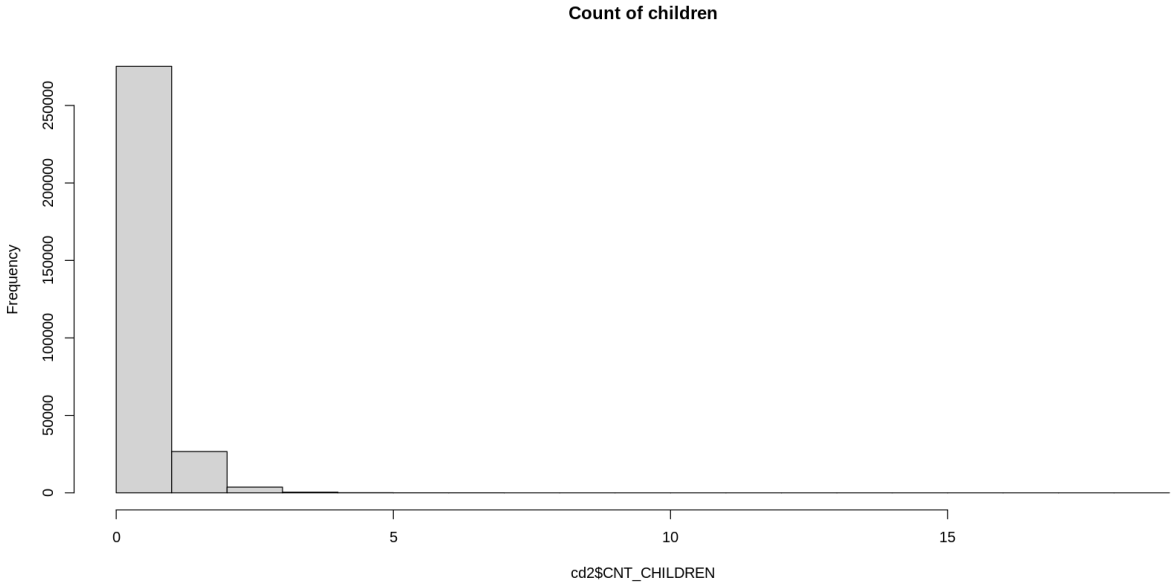


Fig 16: Histogram plot of feature 'CNT_CHILDREN'

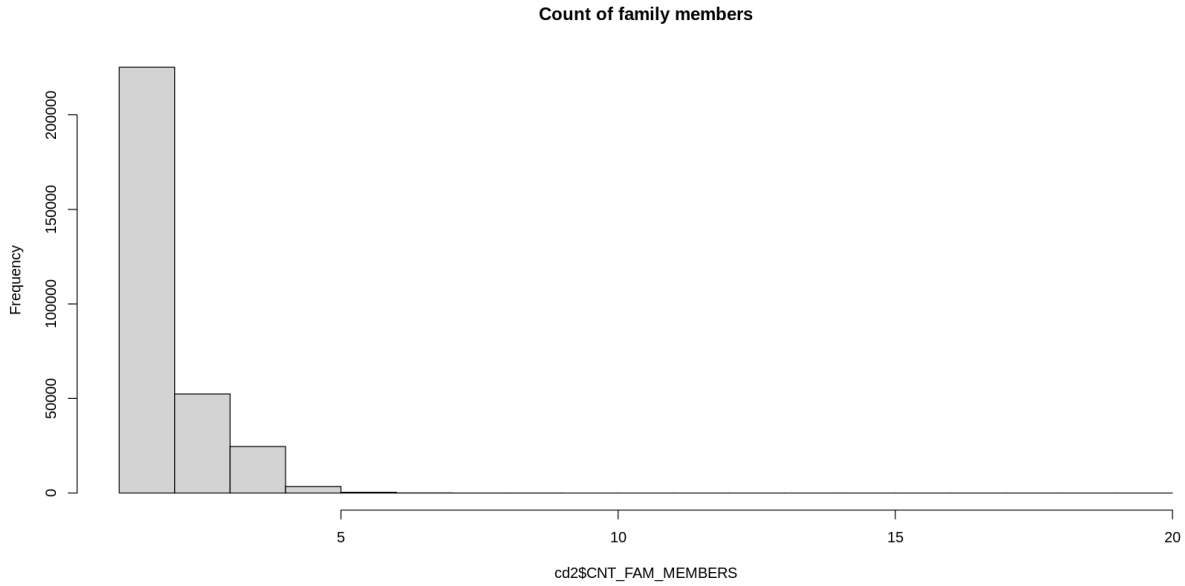


Fig 17: Histogram plot of feature 'CNT_FAM_MEMBERS'

IV. Data Preprocessing:

The dataset 1 had already been preprocessed and thus required next to no preprocessing. Only the attribute Amount was scaled and then the dataset was partitioned into 80% training set and 20% testing set for fitting different machine learning models. However, due to the highly imbalanced data with 99.82% non-fraudulent and 0.001% fraudulent transactions we needed to balance the dataset so that the models fitted are not biased to the majority class. For this purpose, we employed synthetic minority oversampling technique (SMOTE) and tried to synthesize the minority class i.e. the fraudulent transactions to be able to get a balanced dataset and fit models on it and compare their performance.

The dataset 2 consisted of a larger number of predictors of different data types and also consisted of a lot of NA values. The attributes which had a large number of NA values were dropped from the dataset and then the rows consisting of any attributes with NA values were omitted from the dataset to form a clean dataset. After cleaning the dataset consisted of 306199 rows and 63 columns. This new dataset consisted of 11 character data type, 40 integer data type and 12 numeric

data type attributes. The character data type attributes were transformed to integer data type so that they can be used for synthesizing new samples for the minority class using SMOTE and be usable to fit different machine learning models. The target class distribution consisted of 281431 non-fraudulent and 24768 fraudulent transactions.

V. Model Training:

After we have standardized our entire dataset, we will split our dataset into a training set as well as a test set with a split ratio of 0.80. This means that 80% of our data will be attributed to the train_data whereas 20% will be attributed to the test data.

We use two methods in model training like logistic regression method and Decision Tree method both the method only has two outcomes yes or no, so we clearly identified.

In this section of the credit card fraud detection project, we will fit our first model. We will begin with logistic regression. A logistic regression is used for modeling the outcome probability of a class such as pass or fail, positive or negative and in our case – fraud or not fraud. We proceed to implement this model.

In this picture we clearly identified that in logistic regression fitting probabilities numerically 0 or 1. The summary of the fitted logistic model can be seen below.

```
Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -12.52800   10.30537  -1.216   0.2241
V1           -0.17299    1.27381  -0.136   0.8920
V2            1.44512    4.23062   0.342   0.7327
V3            0.17897    0.24058   0.744   0.4569
V4            3.13593    7.17768   0.437   0.6622
V5            1.49014    3.80369   0.392   0.6952
V6           -0.12428    0.22202  -0.560   0.5756
V7            1.40903    4.22644   0.333   0.7388
V8           -0.35254    0.17462  -2.019   0.0435 *
V9            3.02176    8.67262   0.348   0.7275
V10          -2.89571    6.62383  -0.437   0.6620
V11          -0.09769    0.28270  -0.346   0.7297
V12           1.97992    6.56699   0.301   0.7630
V13          -0.71674    1.25649  -0.570   0.5684
V14           0.19316    3.28868   0.059   0.9532
V15           1.03868    2.89256   0.359   0.7195
V16          -2.98194    7.11391  -0.419   0.6751
V17          -1.81809    4.99764  -0.364   0.7160
V18           2.74772    8.13188   0.338   0.7354
V19          -1.63246    4.77228  -0.342   0.7323
V20          -0.69925    1.15114  -0.607   0.5436
V21          -0.45082    1.99182  -0.226   0.8209
V22          -1.40395    5.18980  -0.271   0.7868
V23           0.19026    0.61195   0.311   0.7559
V24          -0.12889    0.44701  -0.288   0.7731
V25          -0.57835    1.94988  -0.297   0.7668
V26           2.65938    9.34957   0.284   0.7761
V27          -0.45396    0.81502  -0.557   0.5775
Chunk 12: cars
```

Fig 18: Summary of logistic model on imbalanced dataset 1

We also plot logistic regression plots which can be observed below.

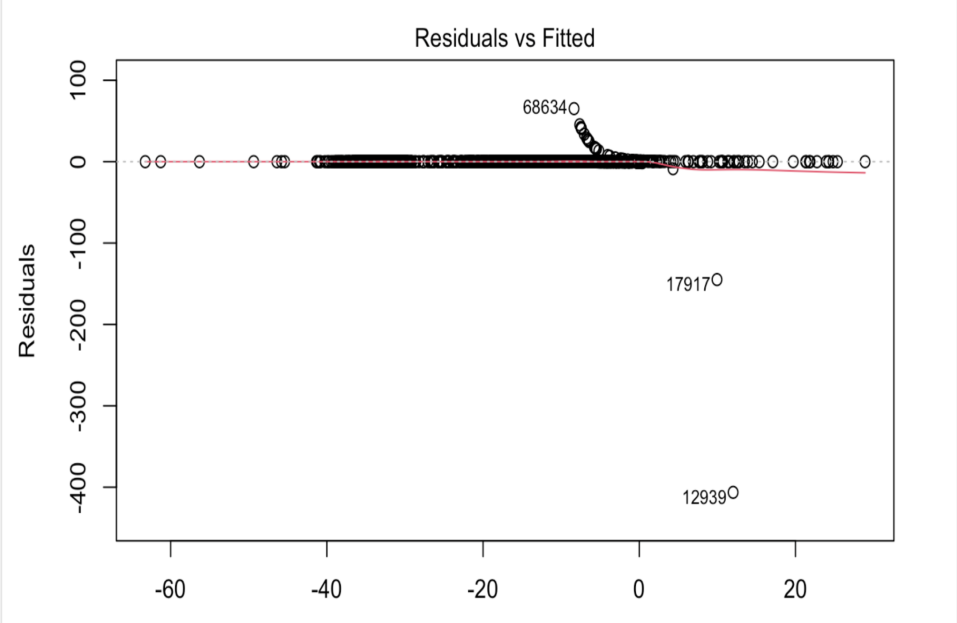


Fig 19: Residuals vs Fitted plot

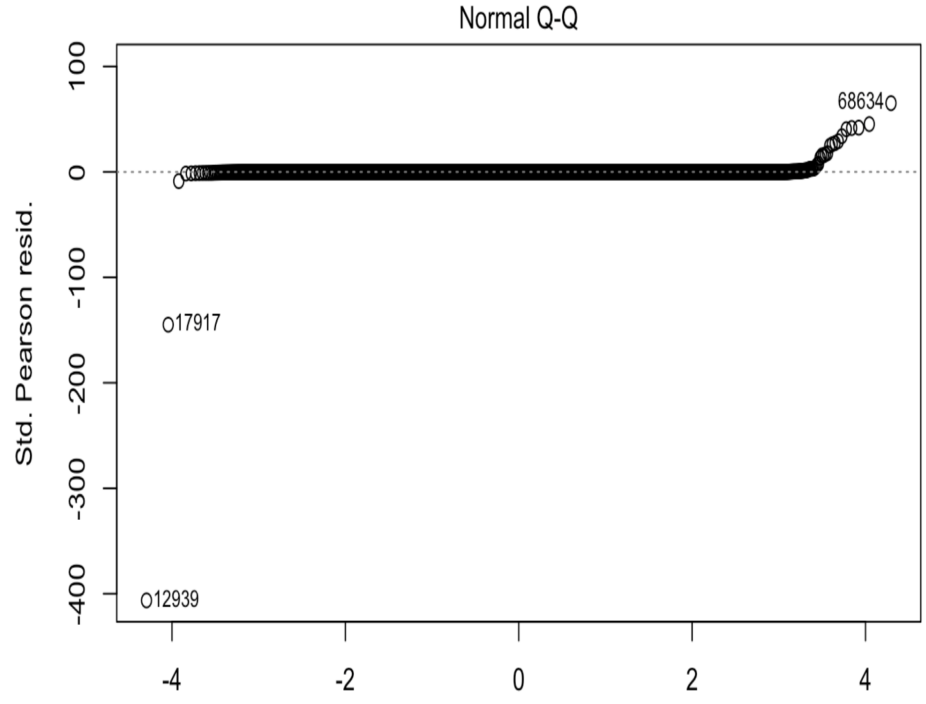


Fig 20: Normal Q-Q plot

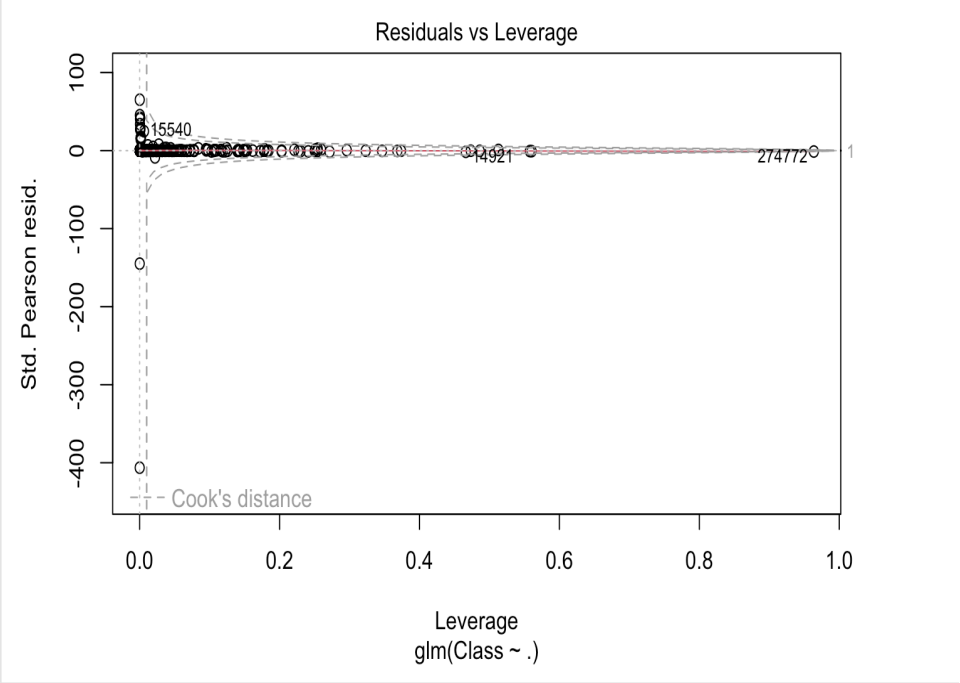


Fig 21: Residuals vs Leverage plot

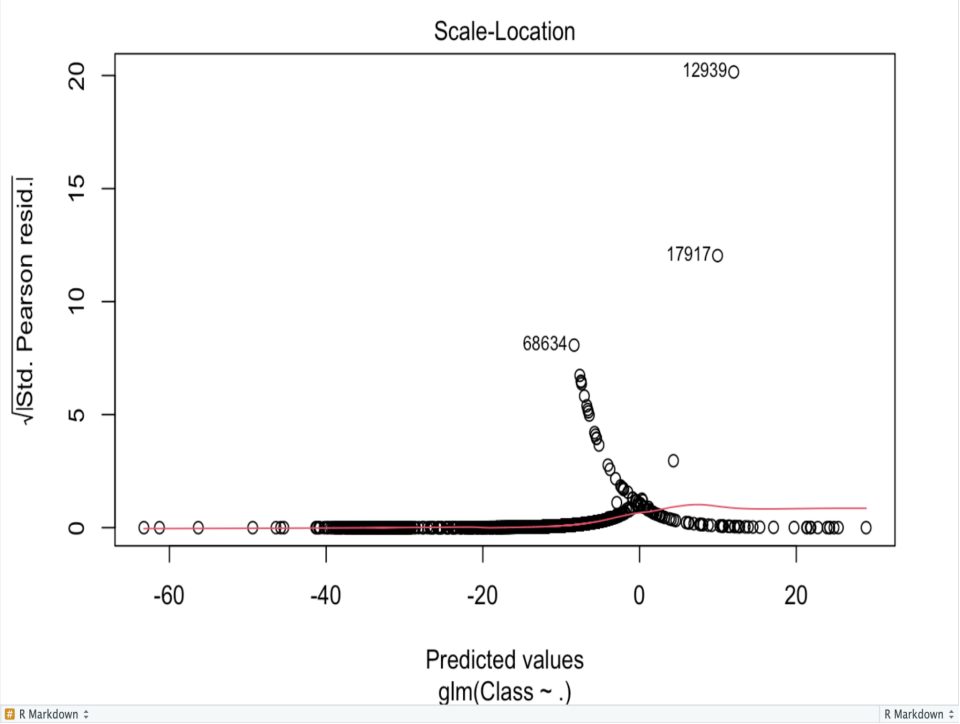


Fig 22: Scale-Location plot

In order to assess the performance of our model, we will delineate the ROC curve. ROC is also known as Receiver Optimistic Characteristics. For this, we will first import the ROC package and then plot our ROC curve to analyze its performance.

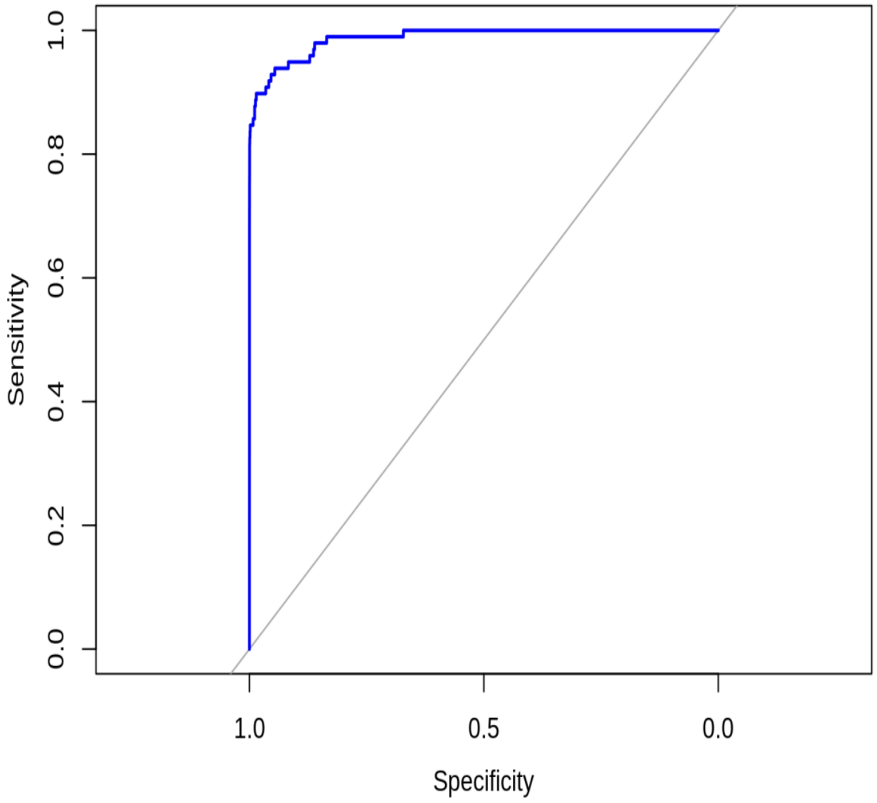


Fig 23: ROC Curve of logistic model

The performance of logistic model on the imbalance dataset:

```
Confusion Matrix and Statistics

          Reference
Prediction     0     1
         0 56857    29
         1     6    69

               Accuracy : 0.9994
                 95% CI : (0.9991, 0.9996)
    No Information Rate : 0.9983
    P-Value [Acc > NIR] : 1.953e-13

                  Kappa : 0.7974

 Mcnemar's Test P-Value : 0.0002003

            Sensitivity : 0.9999
            Specificity : 0.7041
         Pos Pred Value : 0.9995
         Neg Pred Value : 0.9200
             Prevalence : 0.9983
         Detection Rate : 0.9982
   Detection Prevalence : 0.9987
      Balanced Accuracy : 0.8520

       'Positive' Class : 0
```

Fig 24: Confusion Matrix of imbalanced logistic model

A decision tree is a flowchart-like structure in which each internal node represents a test on a feature (e.g. patient if alive or dead), and each leaf node is a class label made after all tests. A decision tree aims to learn ways to split datasets based on conditions. So, it is non-parametric.

We will implement a decision tree algorithm. to plot the outcomes of a decision. These outcomes are basically a consequence through which we can conclude as to what class the object belongs to. We will now implement our decision tree model and will plot it using the rpart.plot() function. We will specifically use the recursive parting to plot the decision tree.

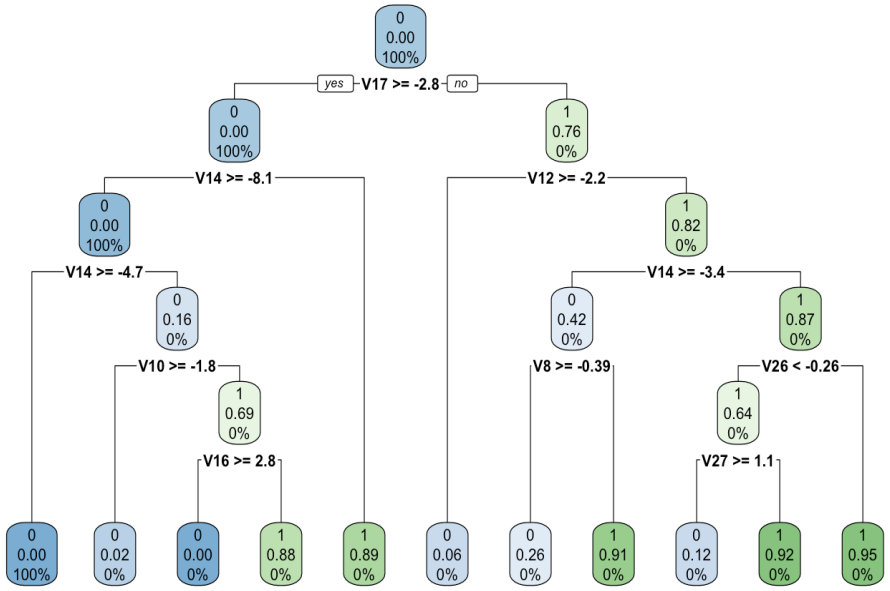


Fig 25: Decision tree model plot on imbalanced dataset 1

The ROC curve for the Decision Tree Model can be seen below.

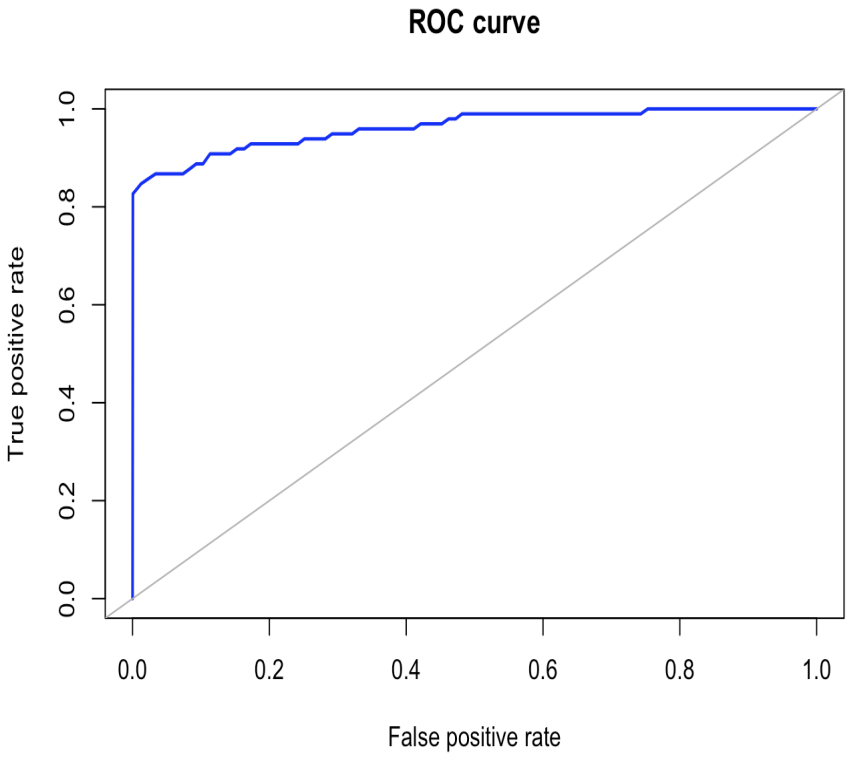


Fig 26: ROC Curve of decision tree model

The performance of decision tree model on the imbalance dataset:

```
Confusion Matrix and Statistics

          Reference
Prediction     0     1
         0 56845    21
         1    18    77

               Accuracy : 0.9993
                 95% CI : (0.9991, 0.9995)
    No Information Rate : 0.9983
    P-Value [Acc > NIR] : 9.766e-12

                  Kappa : 0.7976

 Mcnemar's Test P-Value : 0.7488

            Sensitivity : 0.9997
            Specificity : 0.7857
         Pos Pred Value : 0.9996
         Neg Pred Value : 0.8105
             Prevalence : 0.9983
         Detection Rate : 0.9980
   Detection Prevalence : 0.9983
      Balanced Accuracy : 0.8927

       'Positive' Class : 0
```

Fig 27: Confusion Matrix of imbalanced decision tree model

From this point we display the results of the models with SMOTE technique applied. The reason for using SMOTE as stated earlier was to tackle the imbalance issue. To compare to models without SMOTE, we use the same models. After applying SMOTE the dataset consisted of 227437 samples of non-fraudulent and 127199 samples of fraudulent cases. We create such a training dataset where there is an approximate 65/35 split between positive and negative classes. We only apply this SMOTE technique after splitting the dataset only on the training set, so that the model can appropriately learn before predicting on an out-of-sample dataset.

```
Coefficients:
             Estimate Std. Error  z value Pr(>|z|)
(Intercept) -3.876602   0.017112 -226.538  < 2e-16 ***
V1           0.540604   0.020400   26.500  < 2e-16 ***
V2           0.609282   0.036567   16.662  < 2e-16 ***
V3           0.388265   0.016450   23.602  < 2e-16 ***
V4           0.785808   0.011143   70.518  < 2e-16 ***
V5           0.654587   0.026304   24.886  < 2e-16 ***
V6          -0.550354   0.017442  -31.553  < 2e-16 ***
V7          -0.443284   0.032190  -13.771  < 2e-16 ***
V8          -0.333063   0.011040  -30.169  < 2e-16 ***
V9          -0.277926   0.013739  -20.229  < 2e-16 ***
V10         -0.688587   0.018307  -37.613  < 2e-16 ***
V11          0.560093   0.011854   47.248  < 2e-16 ***
V12         -0.993184   0.017238  -57.617  < 2e-16 ***
V13         -0.464032   0.009116  -50.904  < 2e-16 ***
V14         -1.207337   0.019703  -61.278  < 2e-16 ***
V15          0.042258   0.009977    4.236 2.28e-05 ***
V16         -0.560462   0.019083  -29.370  < 2e-16 ***
V17         -0.714938   0.024052  -29.724  < 2e-16 ***
V18         -0.256191   0.016189  -15.825  < 2e-16 ***
V19          0.207370   0.014095   14.712  < 2e-16 ***
V20         -0.820236   0.032258  -25.427  < 2e-16 ***
V21          0.011311   0.013203    0.857   0.3916
V22          0.517977   0.017223   30.075  < 2e-16 ***
V23          0.475796   0.032290   14.735  < 2e-16 ***
V24         -0.100368   0.018719   -5.362 8.24e-08 ***
V25          0.229613   0.020809   11.034  < 2e-16 ***
V26         -0.475567   0.023431  -20.296  < 2e-16 ***
V27          0.055122   0.031393    1.756   0.0791 .
V28          0.666343   0.040983   16.259  < 2e-16 ***
Amount       1.888255   0.088095   21.434  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 462908  on 354635  degrees of freedom
Residual deviance:  93672  on 354606  degrees of freedom
AIC: 93732
```

Fig 28: Summary of logistic model on SMOTE dataset 1

We plot the logistic model trained with a balanced dataset.

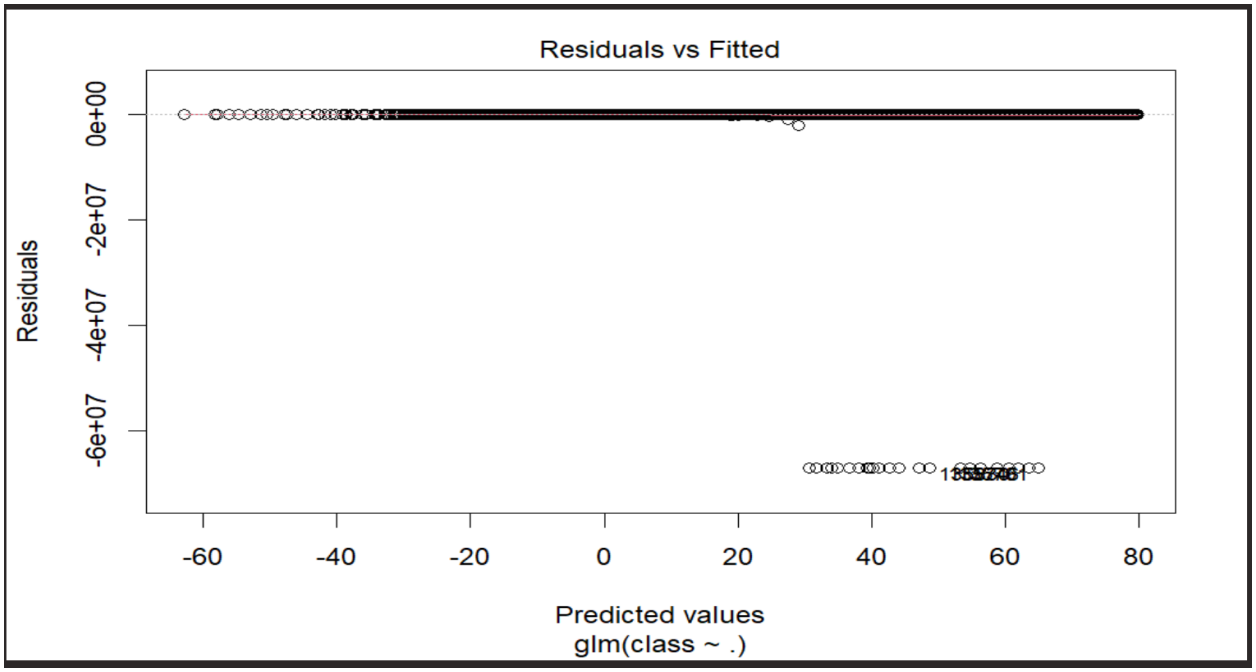


Fig 29: Residuals vs Fitted plot

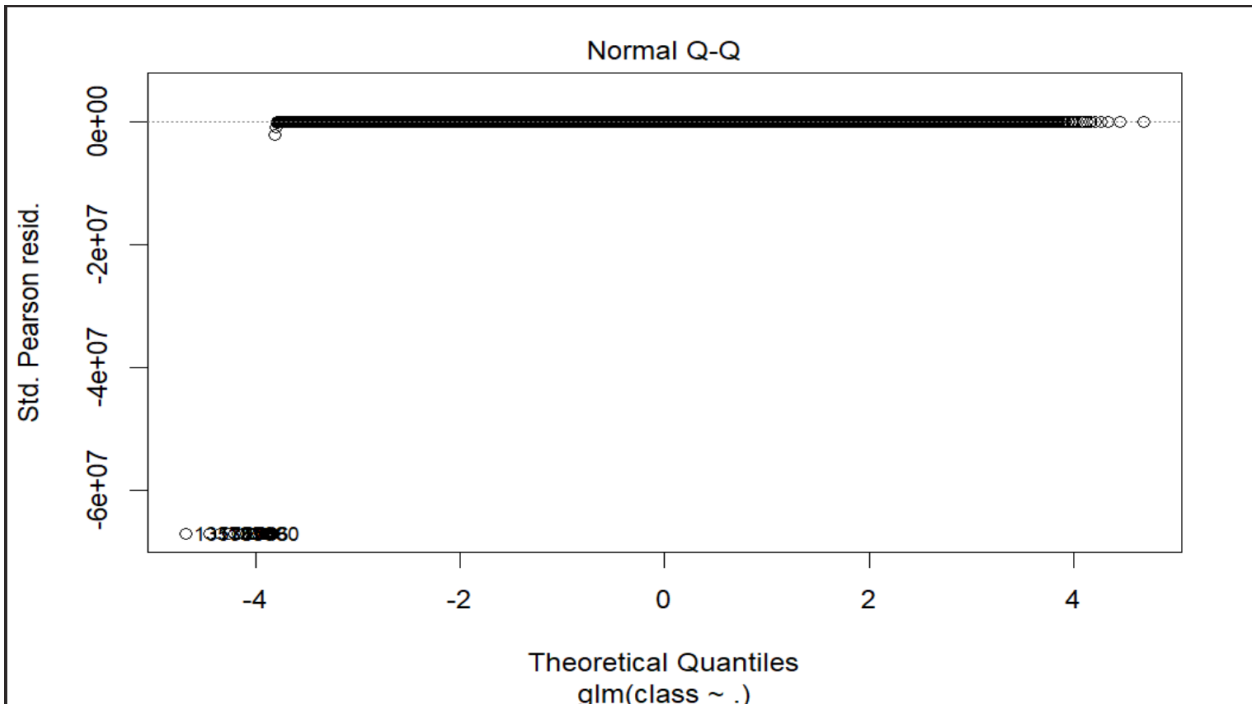


Fig 30: Normal Q-Q plot

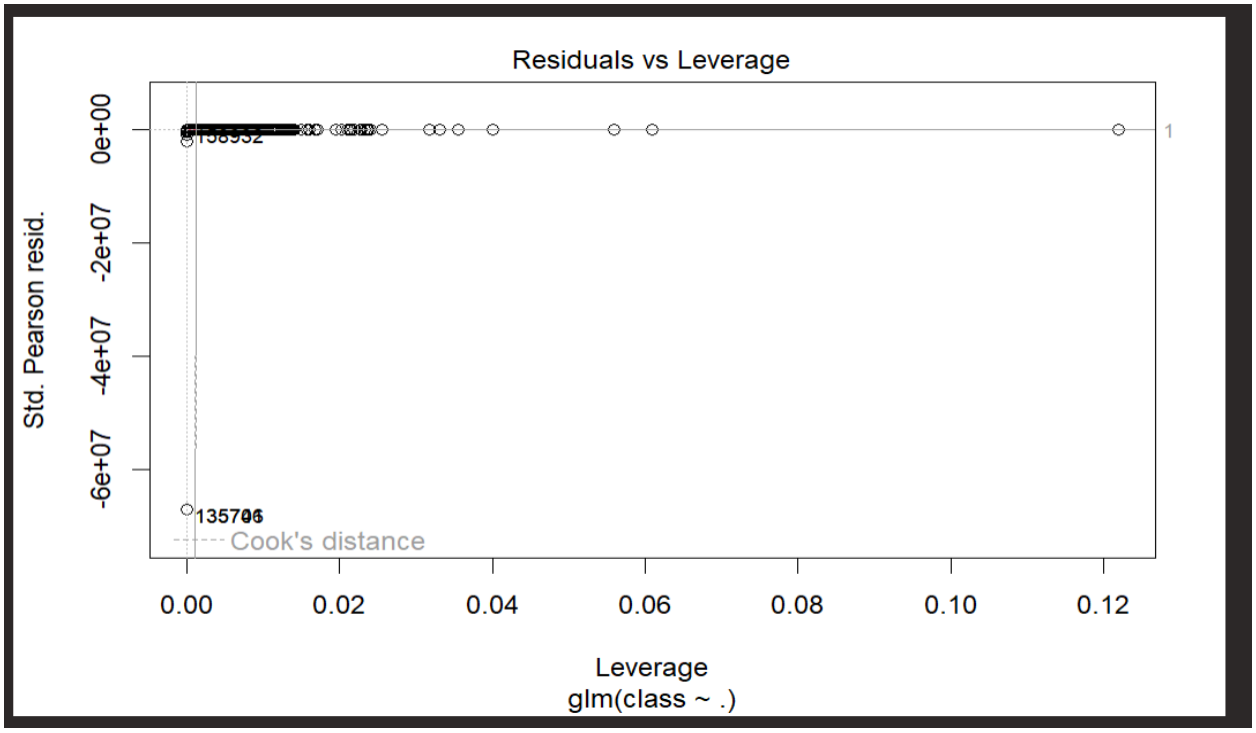


Fig 31: Residual vs Leverage plot

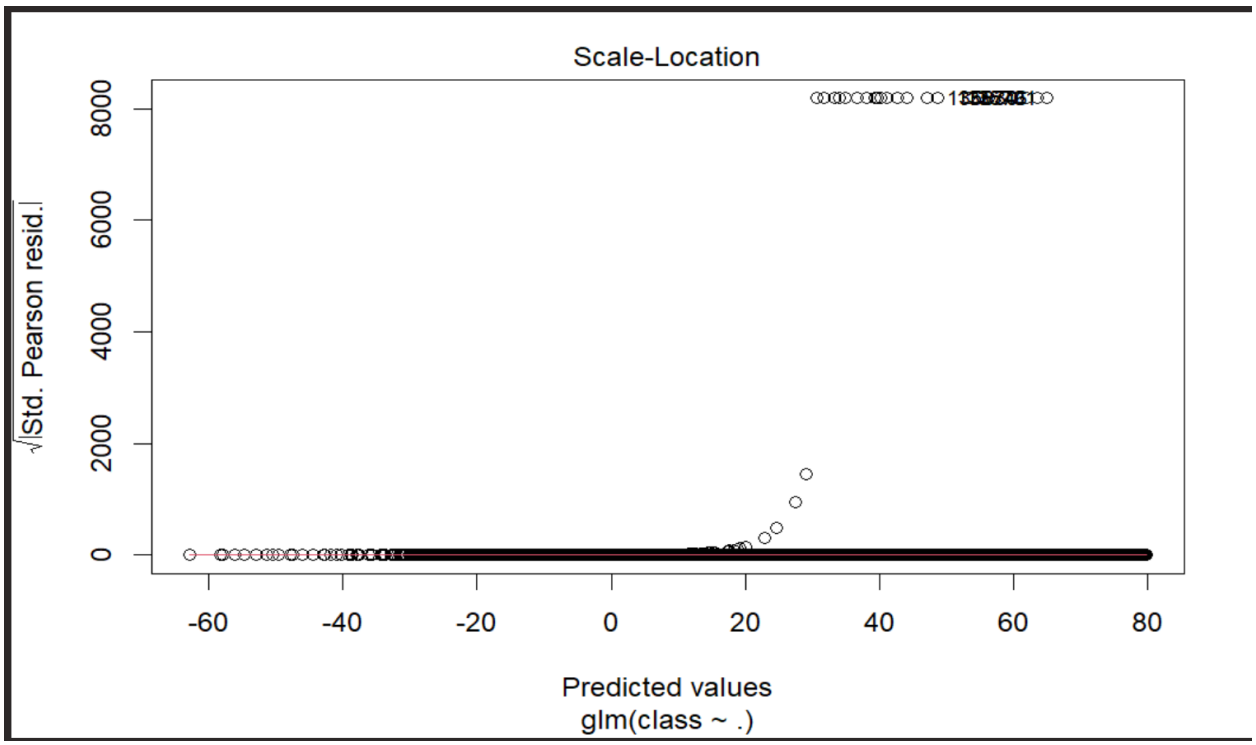


Fig 32: Scale-Location plot

Interestingly, the Residual plots show a cluster around X=30 to X= 60. The Q-Q plot shows a cluster around X=-4. There is a possibility of the model fitting on the synthetic dataset, while the true observations are being treated as the outlier which is actually an unintended behavior.

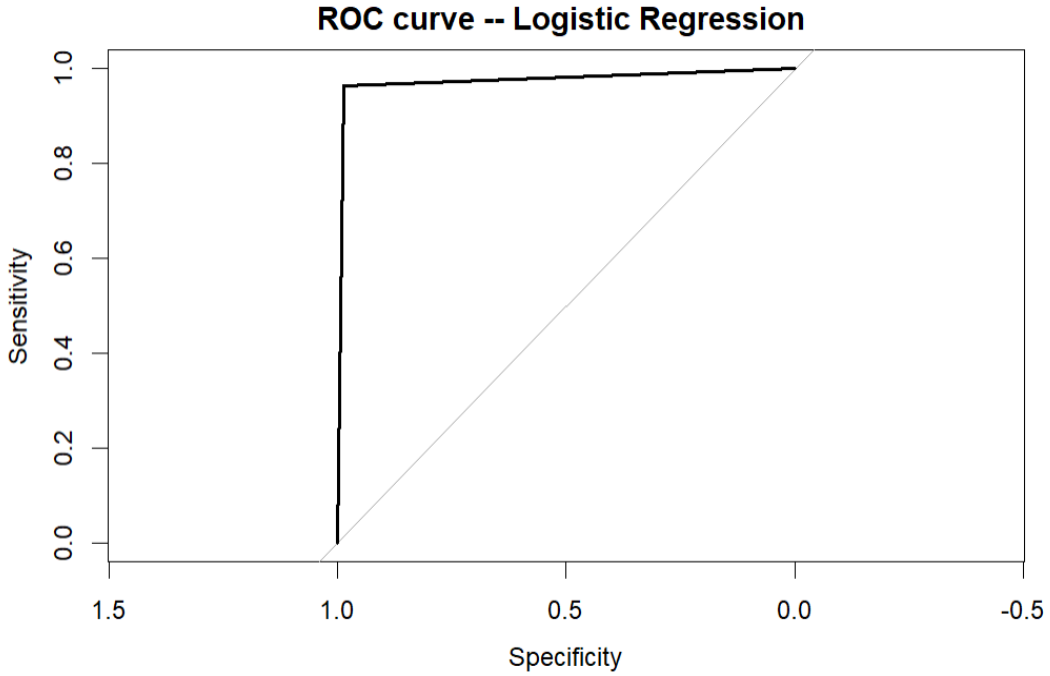


Fig 33: ROC Curve of balanced logistic model

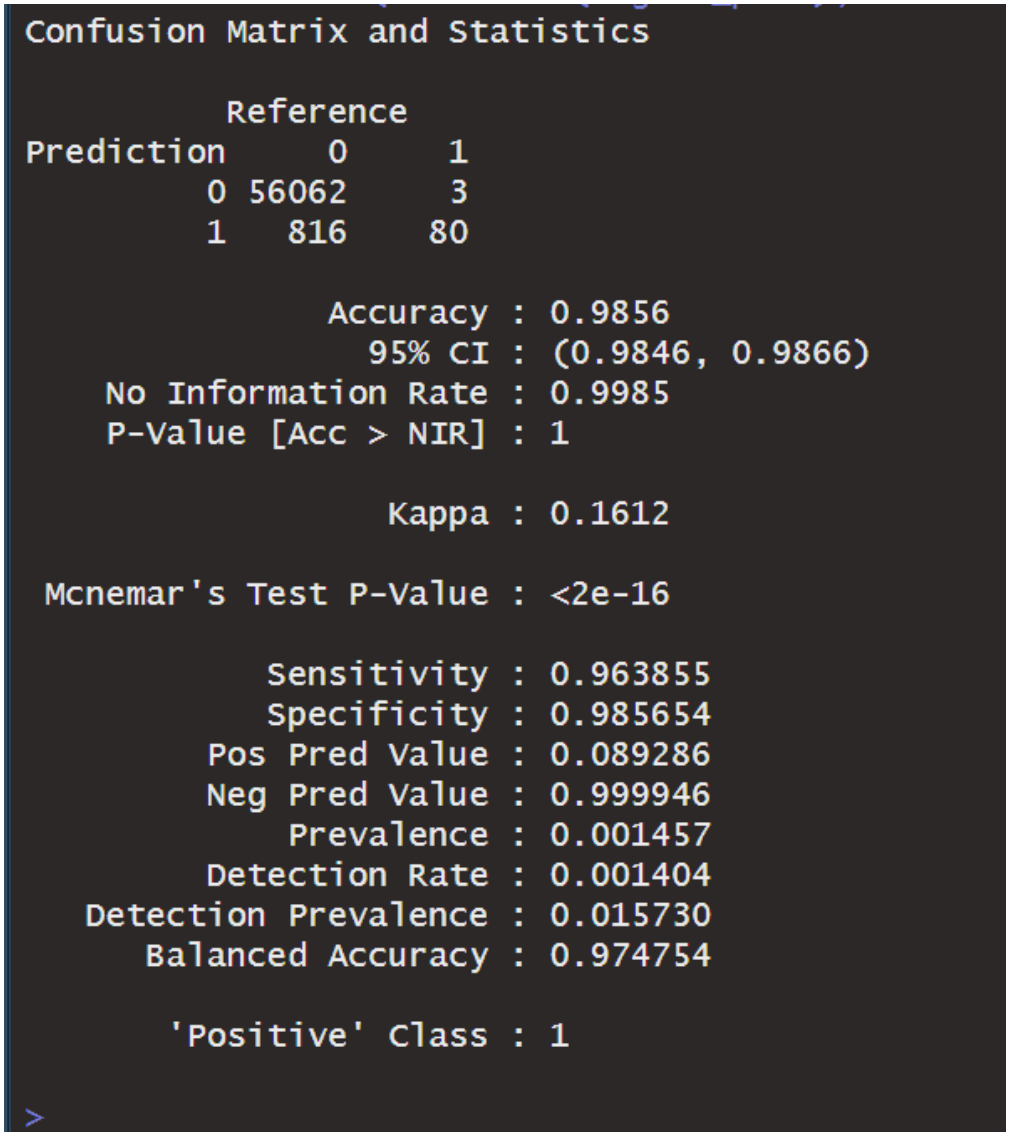


Fig 34: Confusion Matrix of balanced logistic model

For the decision tree model the following tree was the result of using SMOTE:

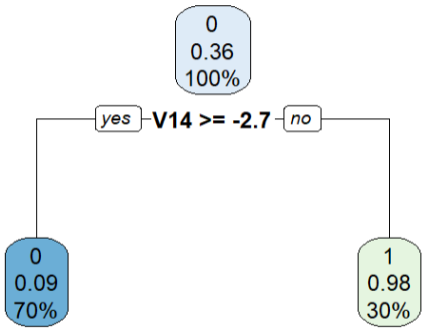


Fig 35: Decision tree model plot on balanced model

With 98% confidence, the tree is able to split on a single feature V14. The ROC below plot follows a similar confidence in its predictions.

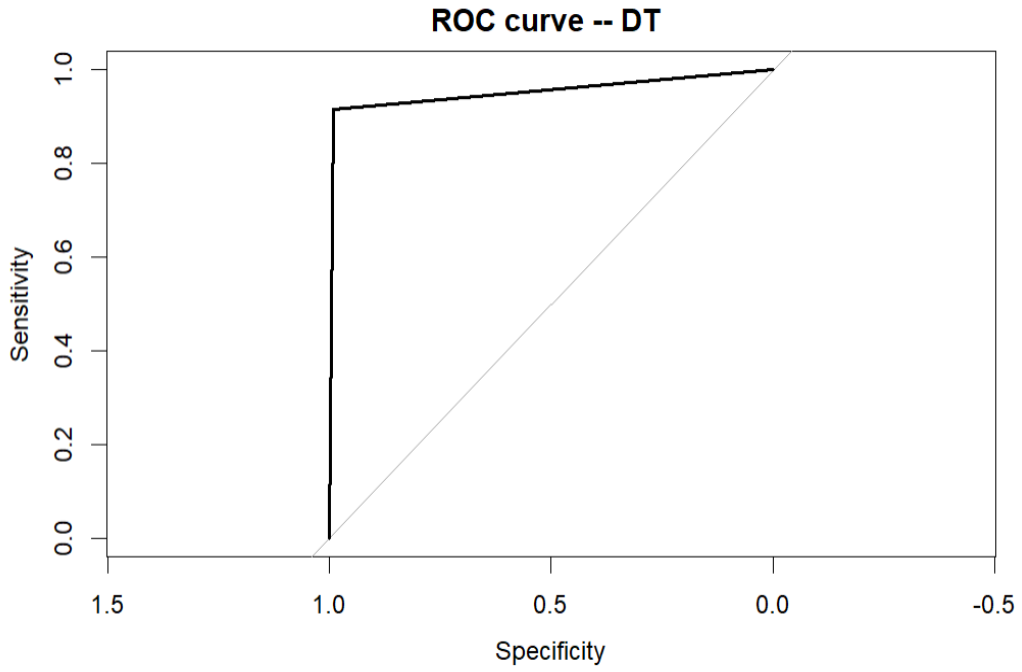


Fig 36: ROC Curve of balanced Decision tree model

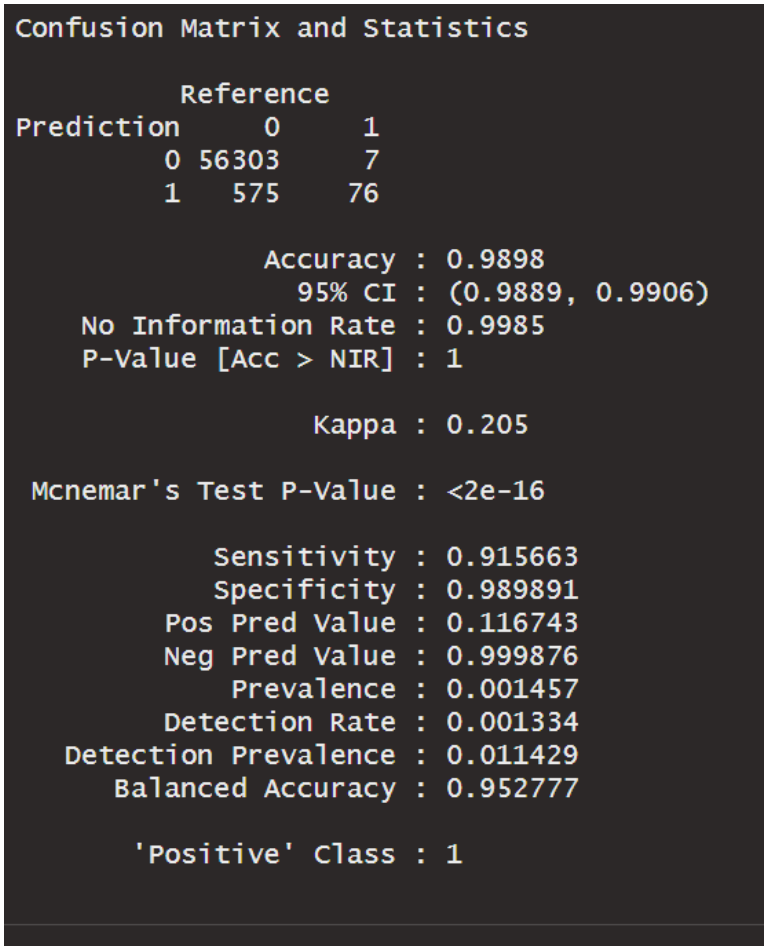


Fig 37: Confusion Matrix of balanced Decision tree model

For dataset 2, the same logistic and decision tree models are fitted on an imbalanced dataset and then on the balanced dataset. The dataset was splitted into 80/20 train/test set and the distribution of the target class was as seen below. The train set consisted of 225145 non-fraudulent and 19815 fraudulent transactions and the test set consisted of 56286 non-fraudulent and 4953 fraudulent transactions when the imbalanced dataset was splitted into the two sets.

The logistic model fitted on the imbalanced data had the following plots.

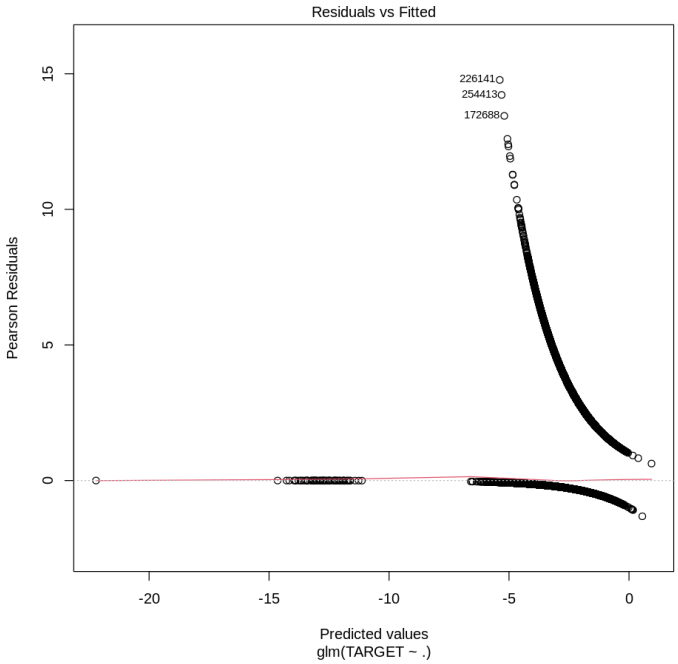


Fig 38: Residuals vs Fitted plot

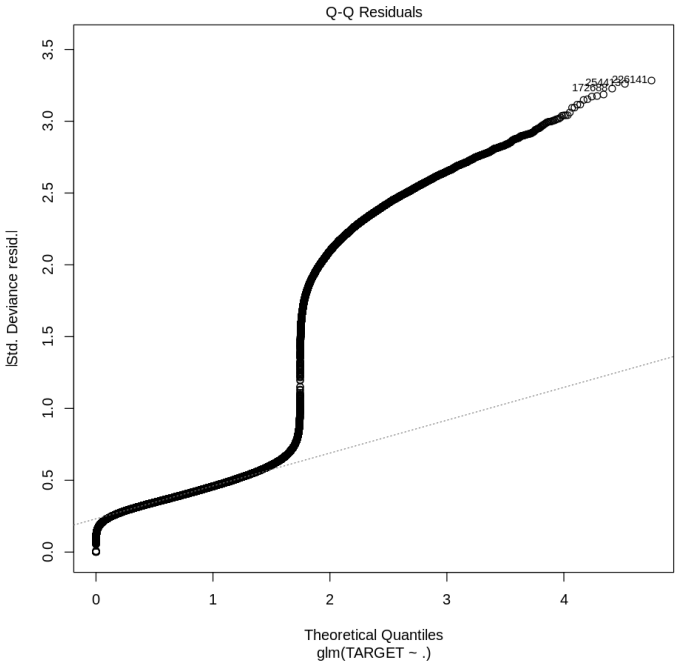


Fig 39: Q-Q Residuals plot

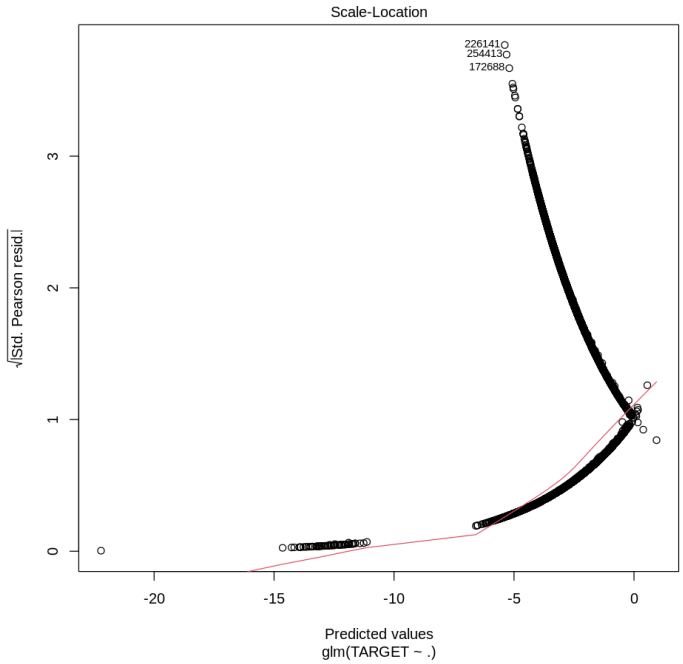


Fig 40: Scale-Location plot

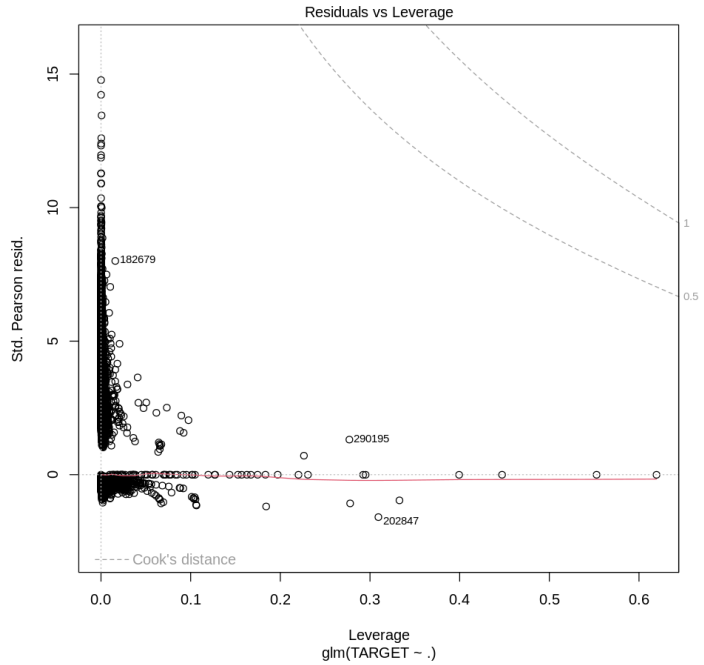


Fig 41: Residual vs Leverage plot

The ROC curve of this fitted logistic model can be seen below.

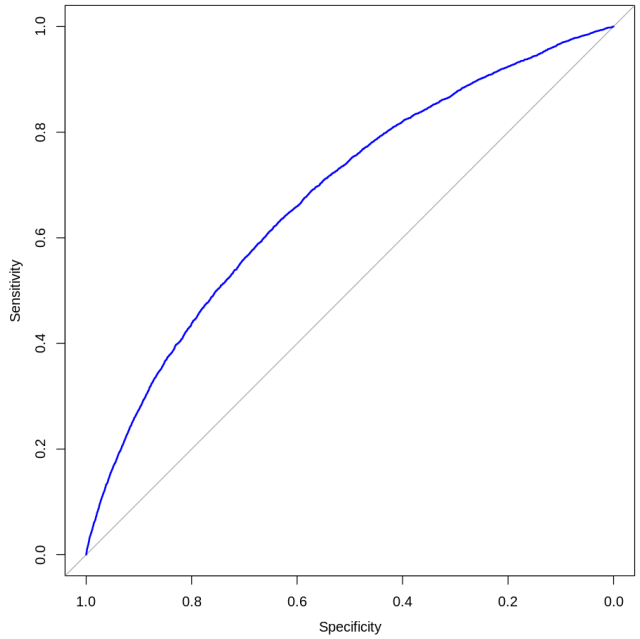


Fig 42: ROC Curve of logistic model

The logistic model had a test accuracy of 0.9191 but from the confusion matrix it can be inferred that the model is highly biased and incapable of classifying the fraudulent datasets as it could only classify 1 sample out of the 4953 fraudulent samples in the test set.

```
Confusion Matrix and Statistics

          Reference
Prediction     0     1
         0 56282  4952
         1     4     1

               Accuracy : 0.9191
                 95% CI : (0.9169, 0.9212)
    No Information Rate : 0.9191
    P-Value [Acc > NIR] : 0.5215

                  Kappa : 2e-04

 Mcnemar's Test P-Value : <2e-16

            Sensitivity : 0.9999289
            Specificity : 0.0002019
         Pos Pred Value : 0.9191299
         Neg Pred Value : 0.2000000
             Prevalence : 0.9191202
         Detection Rate : 0.9190549
   Detection Prevalence : 0.9999184
      Balanced Accuracy : 0.5000654

       'Positive' Class : 0
```

Fig 43: Logistic model performance on test data

The decision tree model fitted on the imbalanced dataset 2 consisted of only one node and thus, the dataset was not convenient for creating machine learning models and performing the classification task.

Fig 44: Decision tree model plot on imbalanced dataset 2

After applying the SMOTE technique, dataset consisting of 281431 non-fraudulent and 272448 fraudulent transactions was obtained. Again a 80/20 split of the train/test set was applied and models were fitted on this balanced dataset 2.

The plots of the logistic model on the balanced dataset can be seen below.

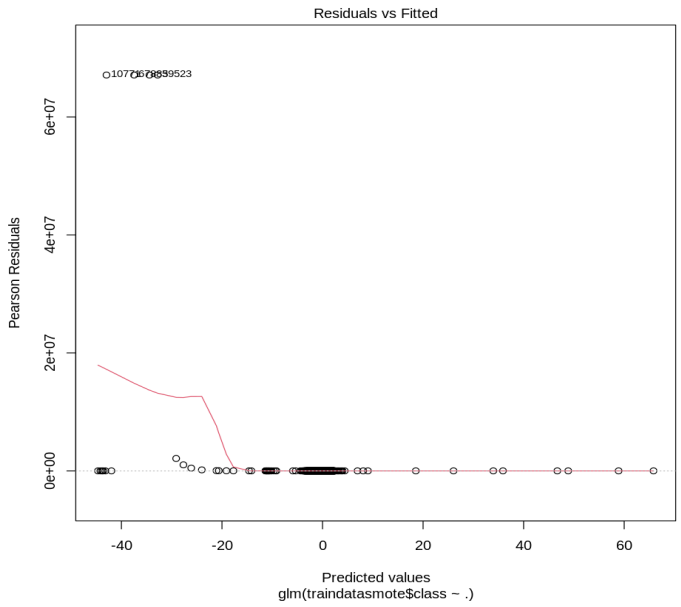


Fig 45: Residuals vs Fitted plot

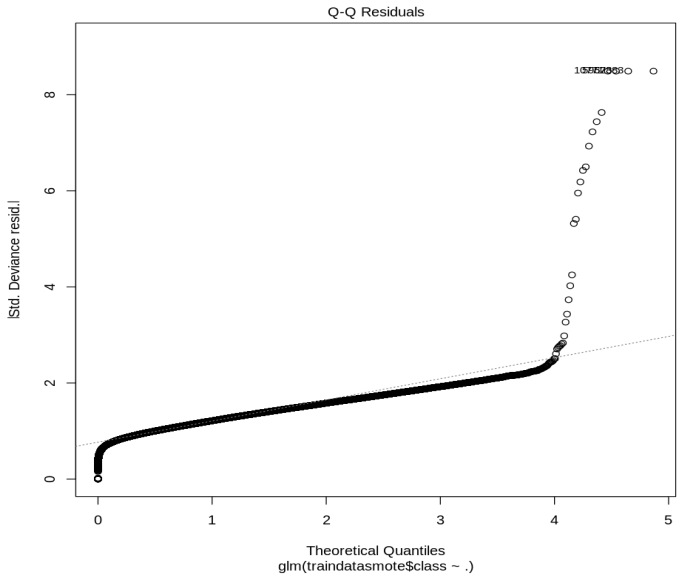


Fig 46: Q-Q Residuals plot

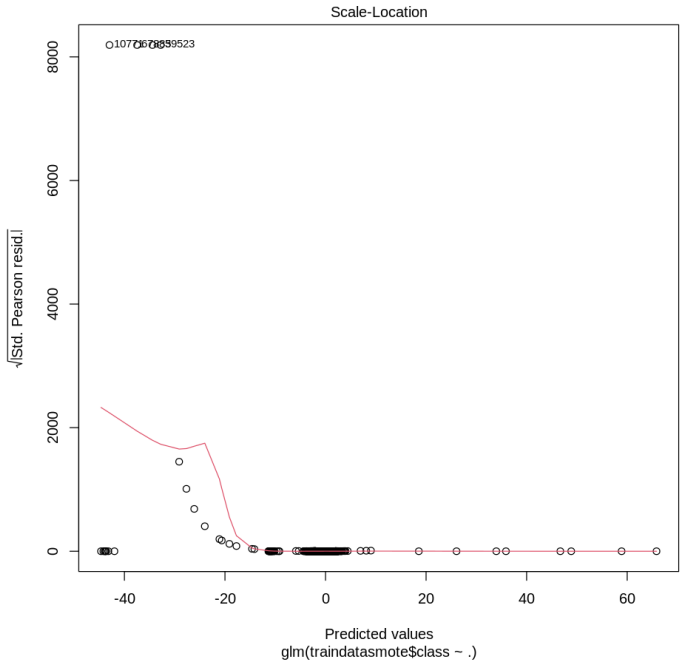


Fig 47: Scale-Location plot

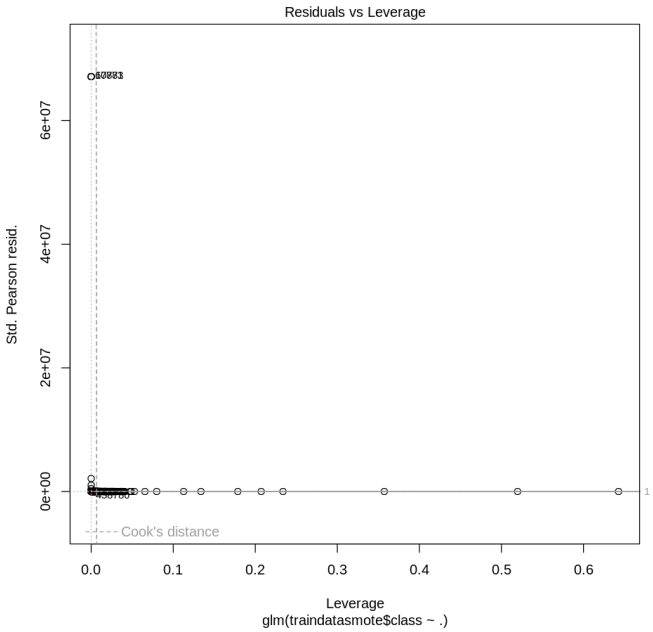


Fig 48: Residuals vs Fitted plot

The ROC curve and AUC score of the logistic model can be observed below.

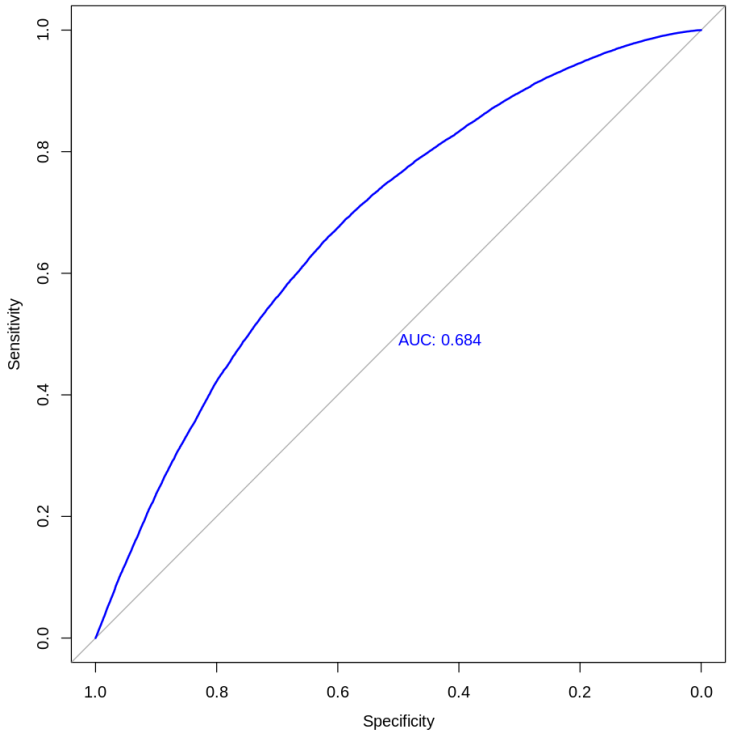


Fig 49: ROC Curve of logistic model

The logistic model had an accuracy of 0.6365 in performing the classification task and was not biased as observed in the logistic model fitted on an imbalanced dataset.

```
Confusion Matrix and Statistics

          Reference
Prediction     1     2
         1 36096 20082
         2 20190 34407

               Accuracy : 0.6365
                 95% CI : (0.6336, 0.6393)
    No Information Rate : 0.5081
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.2727

 Mcnemar's Test P-Value : 0.5939

            Sensitivity : 0.6413
            Specificity : 0.6314
         Pos Pred Value : 0.6425
         Neg Pred Value : 0.6302
             Prevalence : 0.5081
         Detection Rate : 0.3258
   Detection Prevalence : 0.5071
      Balanced Accuracy : 0.6364

       'Positive' Class : 1
```

Fig 50: Logistic model performance on test data

The decision tree model on the balanced dataset 2 formed the following tree.

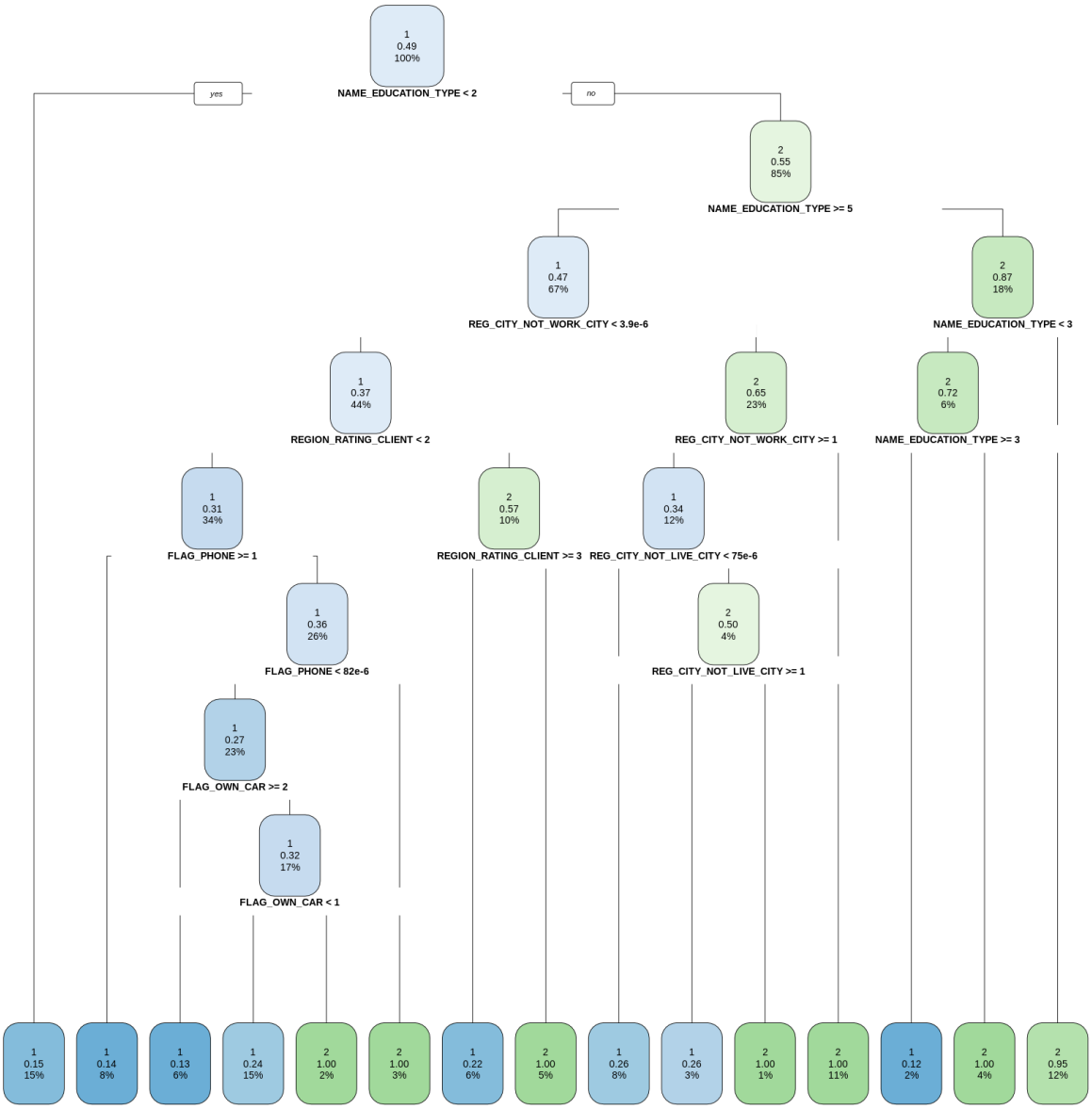


Fig 51: Decision tree model plot on balanced train dataset 2

On the basis of variable importance given by the fitted decision tree the attributes with highest importance can be observed below.

Variable	Importance
<chr>	<dbl>
NAME_EDUCATION_TYPE	4.950860e+04
REG_CITY_NOT_WORK_CITY	3.303368e+04
LIVE_CITY_NOT_WORK_CITY	2.682830e+04
REGION_RATING_CLIENT	2.267733e+04
FLAG_PHONE	1.725110e+04
REGION_RATING_CLIENT_W_CITY	1.699629e+04
REG_CITY_NOT_LIVE_CITY	1.244447e+04
FLAG_OWN_CAR	1.242587e+04
REGION_POPULATION_RELATIVE	3.878306e+03
CNT_CHILDREN	3.172776e+03
OBS_60_CNT_SOCIAL_CIRCLE	1.370675e+03
OBS_30_CNT_SOCIAL_CIRCLE	1.364034e+03
CNT_FAM_MEMBERS	1.346038e+03
FLAG_OWN_REALTY	1.344775e+03
NAME_INCOME_TYPE	1.312754e+03
REG_REGION_NOT_WORK_REGION	1.123527e+03
LIVE_REGION_NOT_WORK_REGION	1.086017e+03
REG_REGION_NOT_LIVE_REGION	5.515178e+02
HOUR_APPR_PROCESS_START	1.447517e+02
FLAG_DOCUMENT_8	1.440828e+02
FLAG_WORK_PHONE	1.429277e+02
NAME_HOUSING_TYPE	1.198994e+02
AMT_GOODS_PRICE	1.077132e+02
AMT_CREDIT	9.760190e+01
AMT_INCOME_TOTAL	4.770814e+01
DAYS_BIRTH	4.253216e+01
FLAG_EMAIL	4.144638e+01
FLAG_DOCUMENT_14	3.696890e+01
AMT_ANNUITY	3.076038e+01
FLAG_DOCUMENT_5	1.712356e+01
FLAG_DOCUMENT_13	1.418557e+01
FLAG_DOCUMENT_11	8.649265e+00
FLAG_EMP_PHONE	3.651006e+00
FLAG_DOCUMENT_15	2.696562e+00
FLAG_DOCUMENT_2	9.272529e-01
DAYS_LAST_PHONE_CHANGE	5.992359e-01

Fig 52: Decision tree model variable importance

The decision tree model had an accuracy of 0.8732 in performing the classification. The model performance on the test set can be observed below.

```
Confusion Matrix and Statistics

          Reference
Prediction     1     2
         1 55630 13389
         2   656 41100

               Accuracy : 0.8732
                 95% CI : (0.8712, 0.8752)
    No Information Rate : 0.5081
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.7454

 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.9883
            Specificity : 0.7543
         Pos Pred Value : 0.8060
         Neg Pred Value : 0.9843
             Prevalence : 0.5081
         Detection Rate : 0.5022
   Detection Prevalence : 0.6231
      Balanced Accuracy : 0.8713

       'Positive' Class : 1
```

Fig 53: Decision tree model performance on test data

The ROC curve and the AUC score of the decision tree model on the balanced dataset 2 can be seen below.

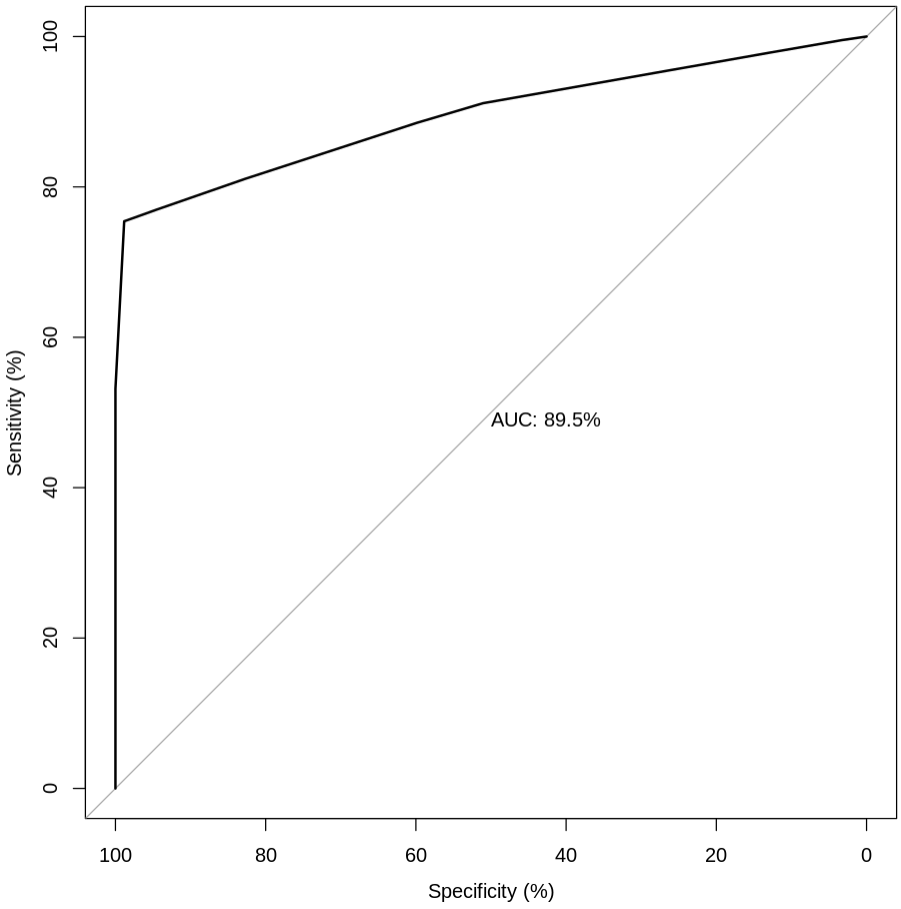


Fig 54: ROC Curve of Decision tree model

We tried to perform feature selection using the Boruta package in R. Boruta is an all relevant feature selection wrapper algorithm, capable of working with any classification method that outputs variable importance measure; by default, Boruta uses Random Forest. The method performs a top-down search for relevant features by comparing original attributes' importance with importance achievable at random, estimated using their permuted copies, and progressively eliminating

irrelevant features to stabilize that test. The boruta package required too much computation time. The entire balanced dataset was fitted into boruta but it took too long and no output was obtained. Then, the train set consisting of 80% samples from the entire dataset was used but still it wasn't possible to fit the boruta model. After that the test set consisting of 20% samples from the entire dataset was used and the following output was obtained which was not very convincing.

```
Boruta performed 10 iterations in 1.234625 hours.
 No attributes deemed important.
 No attributes deemed unimportant.
 63 tentative attributes left: AMT_ANNUITY, AMT_CREDIT,
AMT_GOODS_PRICE, AMT_INCOME_TOTAL, CNT_CHILDREN and 58 more;
```

Fig 55: Boruta feature selection output on balanced test set for dataset 2

So, as observed the boruta model performed 10 interactions in over an hour but couldn't deem any attributes as important or unimportant out of the total 63 tentative attributes. Then we selected the attributes which had shown importance on the decision tree variable importance measure and fitted the boruta model but still the output was not convincing.

```
Boruta performed 10 iterations in 58.74955 mins.
 No attributes deemed important.
 No attributes deemed unimportant.
 18 tentative attributes left: CNT_CHILDREN, CNT_FAM_MEMBERS,
FLAG_OWN_CAR, FLAG_OWN_REALTY, FLAG_PHONE and 13 more;
```

Fig 56: Boruta feature selection output on balanced test set for dataset 2 with features selected based on variable importance by decision tree model

After this, a new dataframe consisting of features selected based on the variable importance of the decision tree model was created and again logistic regression and decision tree models were fitted to compare them with the previously fitted models on the dataset containing all the attributes left after removing those attributes which had a large number of null values. The selected attributes were NAME_EDUCATION_TYPE,REG_CITY_NOT_WORK_CITY,LIVE_CITY_NO

T_WORK_CITY,REGION_RATING_CLIENT,FLAG_PHONE,REGION_RATING_CLIENT_W_CITY,REG_CITY_NOT_LIVE_CITY,FLAG_OWN_CAR,REGION_POPULATION_RELATIVE,CNT_CHILDREN,OBS_60_CNT_SOCIAL_CIRCLE,OBS_30_CNT_SOCIAL_CIRCLE,CNT_FAM_MEMBERS,FLAG_OWN_REALTY,NAME_INCOME_TYPE,REG_REGION_NOT_WORK_REGION,LIVE_REGION_NOT_WORK_REGION,REG_REGION_NOT_LIVE_REGION.

The fitted logistic model showed an accuracy score of 0.5848 lower but similar to the previous model which had an accuracy score of 0.6365 even after the reduction in the number of predictors used for performing the classification.

```
Confusion Matrix and Statistics

          Reference
Prediction     1     2
         1 33813 23524
         2 22465 30973

               Accuracy : 0.5848
                 95% CI : (0.5819, 0.5877)
    No Information Rate : 0.508
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.1692

 Mcnemar's Test P-Value : 8.075e-07

            Sensitivity : 0.6008
            Specificity : 0.5683
         Pos Pred Value : 0.5897
         Neg Pred Value : 0.5796
             Prevalence : 0.5080
         Detection Rate : 0.3052
   Detection Prevalence : 0.5176
      Balanced Accuracy : 0.5846

       'Positive' Class : 1
```

Fig 57: Logistic model performance on test data

The logistic model plots on the model fitted on the new dataframe are as seen below.

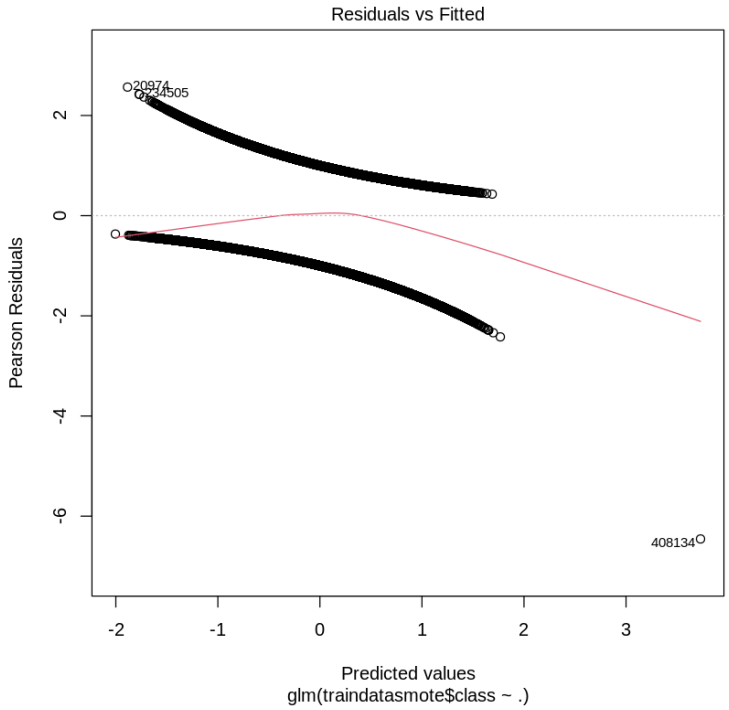


Fig 58: Residuals vs Fitted plot

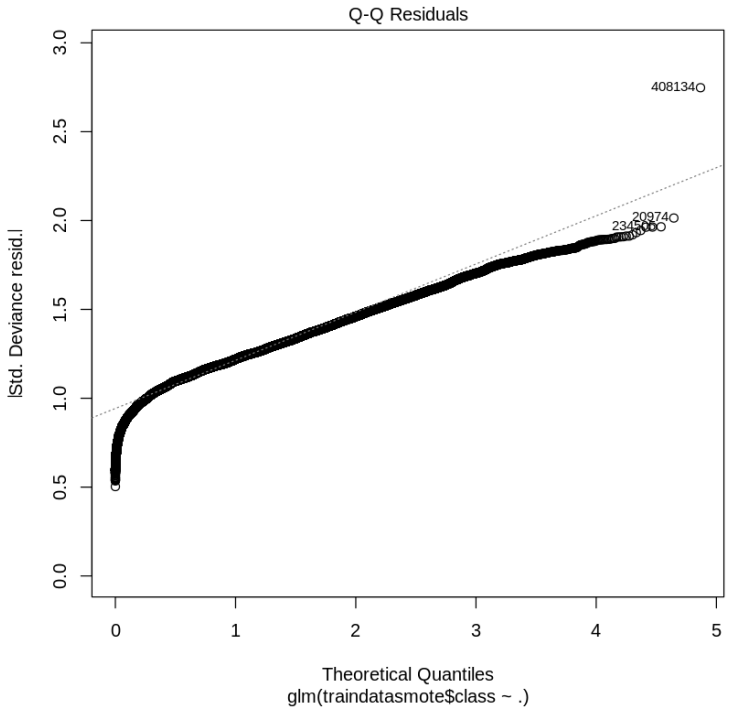


Fig 59: Q-Q Residuals plot

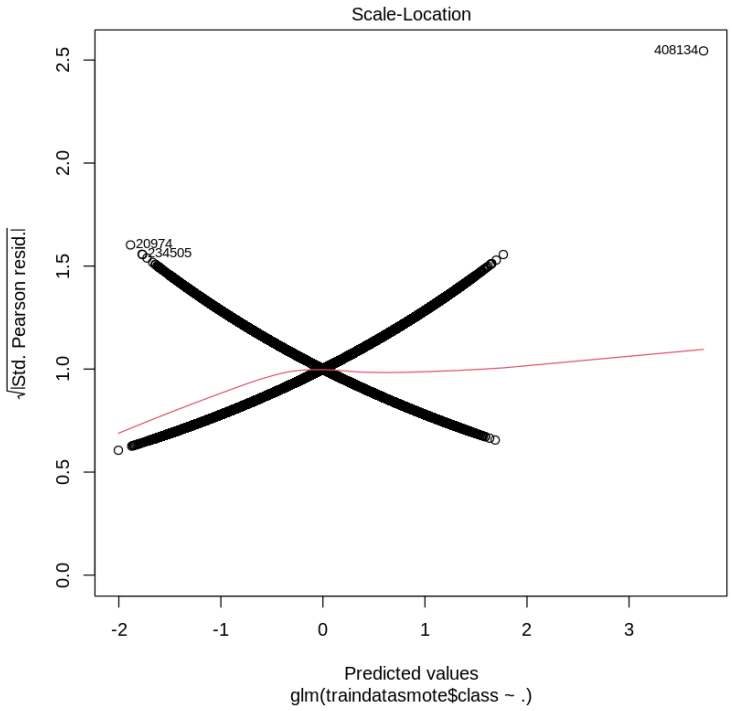


Fig 60: Scale-Location plot

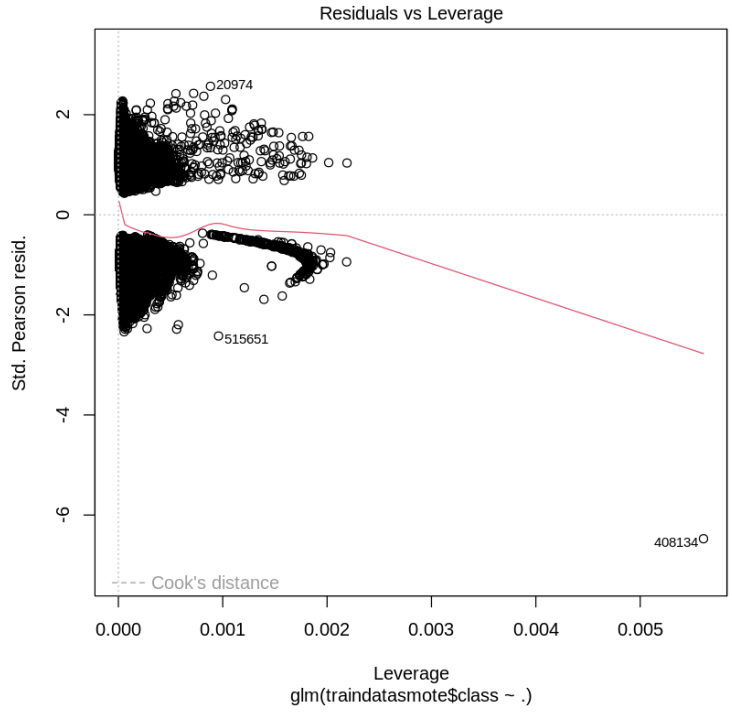


Fig 61: Residuals vs Leverage plot

The ROC curve and the AUC score of this model can be seen below.

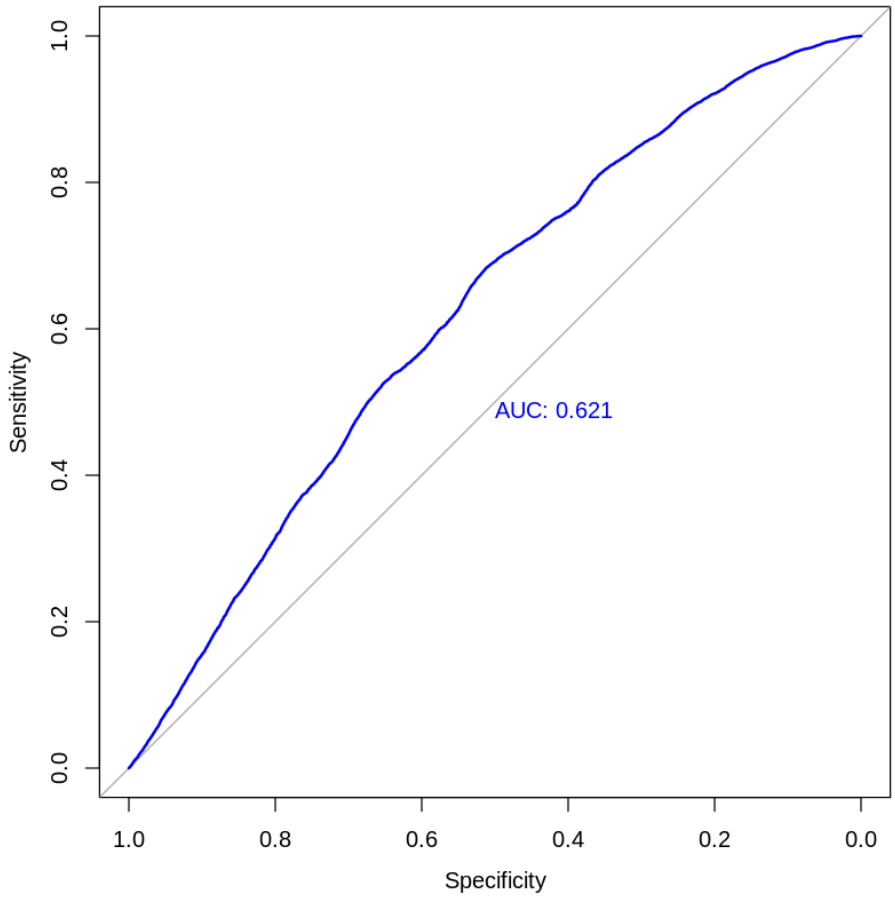


Fig 62: ROC Curve of Logistic model

The AUC score saw a small decrease from 0.684 to 0.621.

The decision tree model fitted on the new dataframe had the following structure.

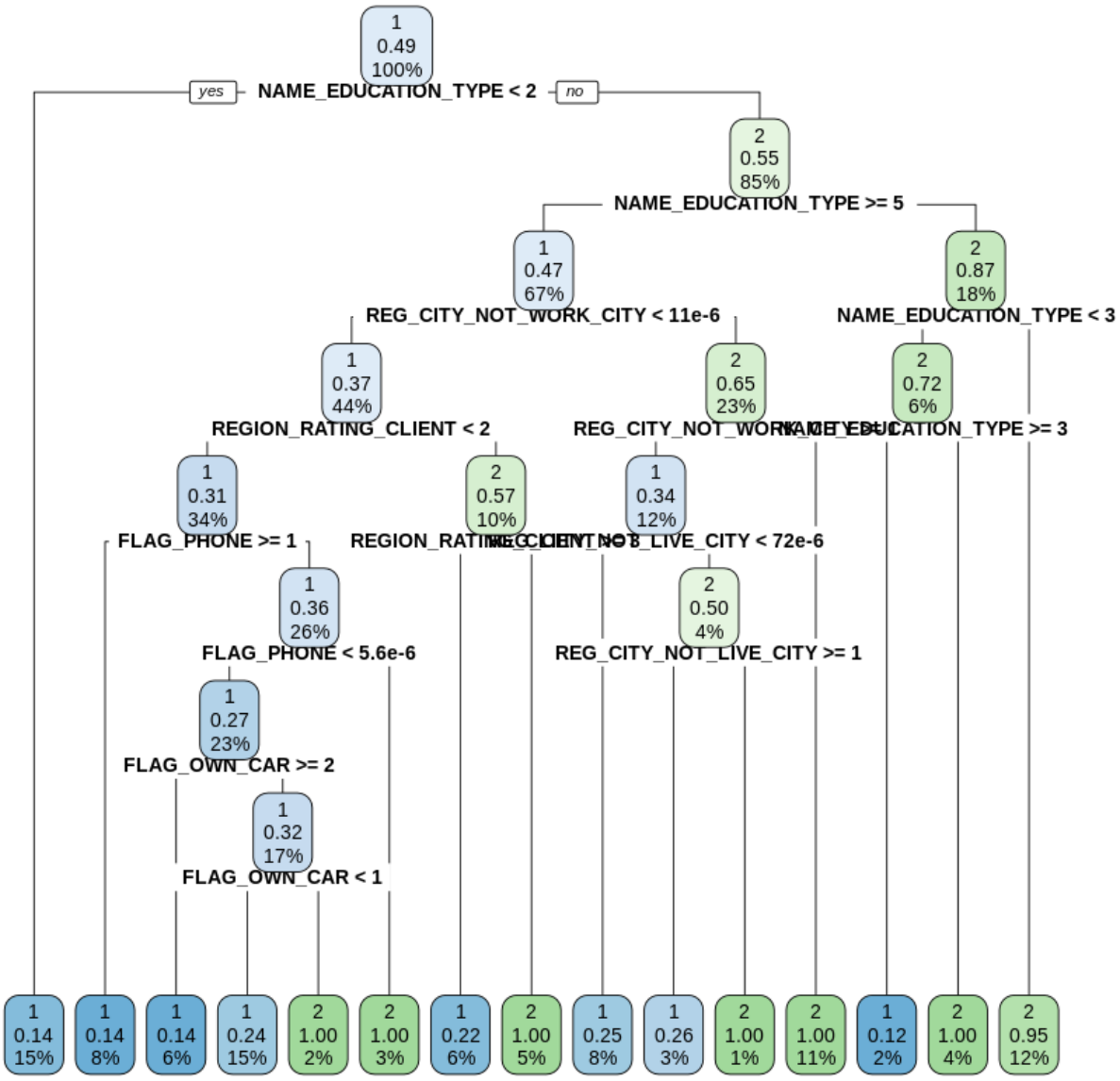


Fig 63: Decision tree model on balanced data with selected features

The accuracy score of this model was 0.873 and performance of the model can be observed below.

```
Confusion Matrix and Statistics

          Reference
Prediction     1     2
         1 55589 13383
         2   689 41114

               Accuracy : 0.873
                 95% CI : (0.871, 0.8749)
    No Information Rate : 0.508
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.7449

 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.9878
            Specificity : 0.7544
         Pos Pred Value : 0.8060
         Neg Pred Value : 0.9835
             Prevalence : 0.5080
         Detection Rate : 0.5018
   Detection Prevalence : 0.6226
      Balanced Accuracy : 0.8711

       'Positive' Class : 1
```

Fig 64: Decision tree model performance on test data

The ROC curve and the AUC score of this model can be observed below.

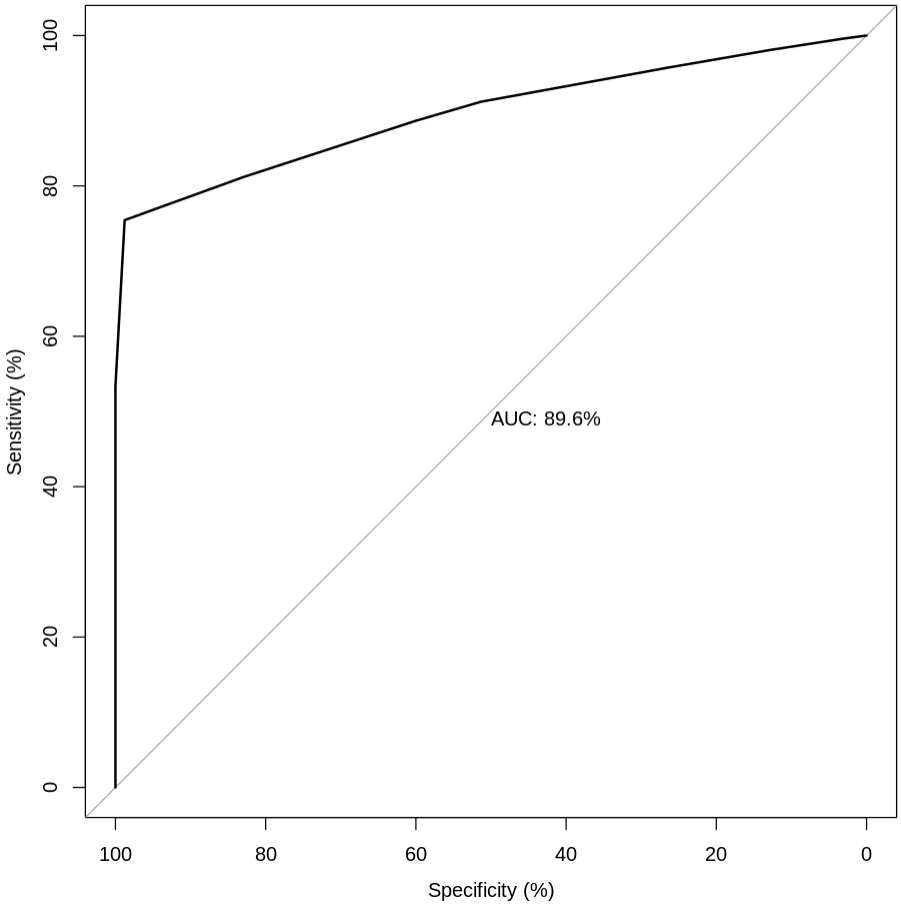


Fig 65: ROC Curve of Decision tree model

The AUC score of the decision tree model on the dataset consisting of selected features was 89.6% which is similar to the AUC score of 89.5% for the previous decision model fitted on a balanced dataset with all the features.

VI. Results:

The dataset 1 was highly imbalanced and hence, the logistic and decision tree models built had a biased performance for the majority class and this was resolved using SMOTE technique to balance the dataset and improve the performance of both the models on both the classes involved in the binary classification problem. The dataset 2 required some preprocessing to be done to use the dataset for fitting classification models which included handling null values and handling different data types present in the dataset. The logistic and decision tree models built on the processed imbalanced dataset were unsuccessful in performing classification as it was observed that even after an accuracy of 90% the logistic model was only able to correctly classify the majority class and had the worst performance for minority class. Decision tree model was also highly biased. After applying SMOTE, the models fitted on the balanced dataset had some acceptable performance for both the classes. The most notable change was the decrease in False Negatives in exchange for an increase in False Positives. In this domain, we decided this was an important improvement, as for fraud, it is better to be more safe than unsafe. Further, the variable importance from decision tree model and boruta was employed to perform feature selection but boruta was not successful. Upon selecting the 18 features from 62 total features based on the variable importance by decision tree model, the logistic and decision tree models fitted on this balanced dataset with 18 features had similar output as before. The accuracy score and AUC score of the logistic model decreased by 5.17% and 6.3% respectively. The accuracy and AUC score for both the decision tree models was 87.3% and 89.5% respectively.

VII. Future work:

For future work, Random Forest, XGBoost, LightGBM and SVM models can be fitted and the performance of these models can be compared. Boruta can be employed using appropriate computational resources to get some meaningful outputs. Other feature selection algorithms like recursive feature elimination (RFE) algorithm can be used to find the relevant features for the classification task.

VIII. References:

[1] MLG - Machine Learning Group - ULB. (2016). Credit Card Fraud Detection Dataset. Kaggle. https://www.kaggle.com/mlg-ulb/creditcardfraud

[2] Mishra, A. (2020). Credit Card Dataset. Kaggle. https://www.kaggle.com/mishra5001/credit-card

[3] Inscribe.AI. (n.d.). Credit Card Fraud Detection. https://www.inscribe.ai/fraud-detection/credit-fraud-detection#:~:text=What%20is%20credit%20card%20fraud,fraud%20and%20stop%20fraudulent%20transactions

[4] Chargeback Gurus. (2021). Credit Card Fraud Detection: The Essential Guide. https://www.chargebackgurus.com/blog/credit-card-fraud-detection

[5] Janiobachmann. (2018). Credit Fraud: Dealing with Imbalanced Datasets. Kaggle. Retrieved April 28, 2023, https://www.kaggle.com/janiobachmann/credit-fraud-dealing-with-imbalanced-datasets.

[6] Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, and W. Philip Kegelmeyer. 2002. SMOTE: synthetic minority over-sampling technique. J. Artif. Int. Res. 16, 1 (January 2002), 321–357. https://dl.acm.org/doi/10.5555/1622407.1622416

[7] Boruta package: https://cran.r-project.org/web/packages/Boruta/Boruta.pdf

Github Link: https://github.com/pgovrineni/DPAprojectSpring2023