CONTENTS

Abstract

It is vital that credit card companies are able to identify fraudulent credit card transactions so that customers are not charged for items that they did not purchase. Such problems can be tackled with Data Science and its importance, along with Machine Learning, cannot be overstated. This project intends to illustrate the modelling of a data set using machine learning with Credit Card Fraud Detection. The Credit Card Fraud Detection Problem includes modelling past credit card transactions with the data of the ones that turned out to be fraud. This model is then used to recognize whether a new transaction is fraudulent or not. Our objective here is to detect 100% of the fraudulent transactions while minimizing the incorrect fraud classifications. Credit Card Fraud Detection is a typical sample of classification. In this process, we have focused on analysing and preprocessing data sets as well as the deployment of multiple anomaly detection algorithms such as Local Outlier Factor and Isolation Forest algorithm on the PCA transformed Credit Card Transaction data.

Lists

List of Tables:-

List of Figures:-

List of Symbols:-

➔ ‘R’ - R Programming Language

INTRODUCTION

'Fraud' in credit card transactions is unauthorized and unwanted usage of an account by someone other than the owner of that account. Necessary prevention measures can be taken to stop this abuse and the behaviour of such fraudulent practices can be studied to minimize it and protect against similar occurrences in the future.In other words, Credit Card Fraud can be defined as a case where a person uses someone else's credit card for personal reasons while the owner and the card issuing authorities are unaware of the fact that the card is being used.

Fraud detection involves monitoring the activities of populations of users in order to estimate, perceive or avoid objectionable behaviour, which consist of fraud, intrusion, and defaulting.

This is a very relevant problem that demands the attention of communities such as machine learning and data science where the solution to this problem can be automated.

This problem is particularly challenging from the perspective of learning, as it is characterized by various factors such as class imbalance. The number of valid transactions far outnumber fraudulent ones. Also, the transaction patterns often change their statistical properties over the course of time.

These are not the only challenges in the implementation of a real-world fraud detection system, however. In real world examples, the massive stream of payment

requests is quickly scanned by automatic tools that determine which transactions to authorize.

Machine learning algorithms are employed to analyse all the authorized transactions and report the suspicious ones. These reports are investigated by professionals who contact the cardholders to confirm if the transaction was genuine or fraudulent.
The investigators provide feedback to the automated system which is used to train and update the algorithm to eventually improve the fraud-detection performance over time.

Fraud detection methods are continuously developed to defend criminals in adapting to their fraudulent strategies. These frauds are classified as:

- Credit Card Frauds: Online and Offline
- Card Theft
- Account Bankruptcy
- Device Intrusion
- Application Fraud
- Counterfeit Card

Telecommunication Fraud some of the currently used approaches to detection of such fraud are:

- Artificial Neural Network
- Fuzzy Logic
- Genetic Algorithm
- Logistic Regression
- Decision tree
- Support Vector Machines
- Bayesian Networks
- Hidden Markov Model
- K-Nearest Neighbour

PROBLEM STATEMENT:

Not al1 the doubtful transactions were considered fraudulent. It is commonly called as false positive (FP) which means that the case was not fraud although it was flagged as being potentially scam. This process of affirming each transaction those outliers from the cardholder's normal routine brings doubt about possible client disappointment. Additionally, the expenses related to exploring an enormous no. of false positives are high.

Motivation:

As of now, a considerable amount of time is given for examining countless genuine cases (FPs). On the off chance that the quantity of examination on FPs could be dropped down, scam analysts can invest more energy and time in genuine fraud transactions that restricts the losses to the FIs.

Literature Review

Fraud acts as the unlawful or criminal deception intended to result in financial or personal benefit. It is a deliberate act that is against the law, rule or policy with an aim to attain unauthorized financial benefit.

Numerous literatures pertaining to anomaly or fraud detection in this domain have been published already and are available for public usage. A comprehensive survey conducted by Clifton Phua and his associates have revealed that techniques employed in this domain include data mining applications, automated fraud detection, adversarial detection. In another paper, Suman, Research Scholar, GJUS&T at Hisar HCE presented techniques like Supervised and Unsupervised Learning for credit card fraud detection. Even though these methods and algorithms fetched an unexpected success in some areas, they failed to provide a permanent and consistent solution to fraud detection.

A similar research domain was presented by Wen-Fang YU and Na Wang where they used Outlier mining, Outlier detection mining and Distance sum algorithms to accurately predict fraudulent transactions in an emulation experiment of credit card transaction data set of one certain commercial bank. Outlier mining is a field of data mining which is basically used in monetary and internet fields. It deals with detecting objects that are detached from the main system i.e. the transactions that aren’t genuine. They have taken attributes of a customer's behaviour and based on the value of those attributes they’ve calculated the distance between the observed value of that attribute and its predetermined value. Unconventional techniques such as hybrid data

mining/complex network classification algorithm is able to perceive illegal instances in an actual card transaction data set, based on network reconstruction algorithm that allows creating representations of the deviation of one instance from a reference group have proved efficient typically on medium sized online transactions.

There have also been efforts to progress from a completely new aspect. Attempts have been made to improve the alert feedback interaction in case of fraudulent transactions. In case of fraudulent transaction, the authorised system would be alerted and a feedback would be sent to deny the ongoing transaction.

Artificial Genetic Algorithm, one of the approaches that shed new light in this domain, countered fraud from a different direction.

It proved accurate in finding out the fraudulent transactions and minimizing the number of false alerts. Even so, it was accompanied by classification problems with variable misclassification costs.

REQUIREMENT ANALYSIS

Functional Requirements:

Functional requirements are the characteristics of the product. All the features expected from any development are mentioned as functional requirements.

1. The solution must be able to solve the problem with the highest possible accuracy.
2. The data must be preprocessed before being used.
3. Adequate amount of attention must be paid to bring out essential features by feature scaling.
4. Working dataset must not contain any null or missing values.
5. Predictions must provide realistic outcomes in tune with the fed data.

Non-Functional Requirements:

1. Work must be self-explanatory.
2. UI/UX must be appealing and easy to comprehend.
3. Graphs and illustrations must be properly organized.
4. Latency of model working must be minimal.

Software Requirements:

1. R Studio

   R-Studio is an integrated development environment (IDE) for R. It includes a console, syntax-highlighting editor that supports direct code execution, as well as tools for plotting, history, debugging and workspace management.

2. Spreadsheet

   A spreadsheet is a computer application for organization, analysis, and storage of data in tabular form. Spreadsheets were developed as computerized analogs of paper accounting worksheets. The program operates on data entered in cells of a table.

3. A Web Browser

   (Google chrome(recommended), Safari, Mozilla Firefox, Brave browser or Internet Explorer).A web browser (Google chrome(recommended), Safari, Mozilla Firefox, Brave browser or Internet Explorer)

4. Operating System

Windows 7 or higher, MAC OS X, Linux Distros (Ubuntu or CentOS)

5. Hosting

A shinyapp.io account for public hosting.

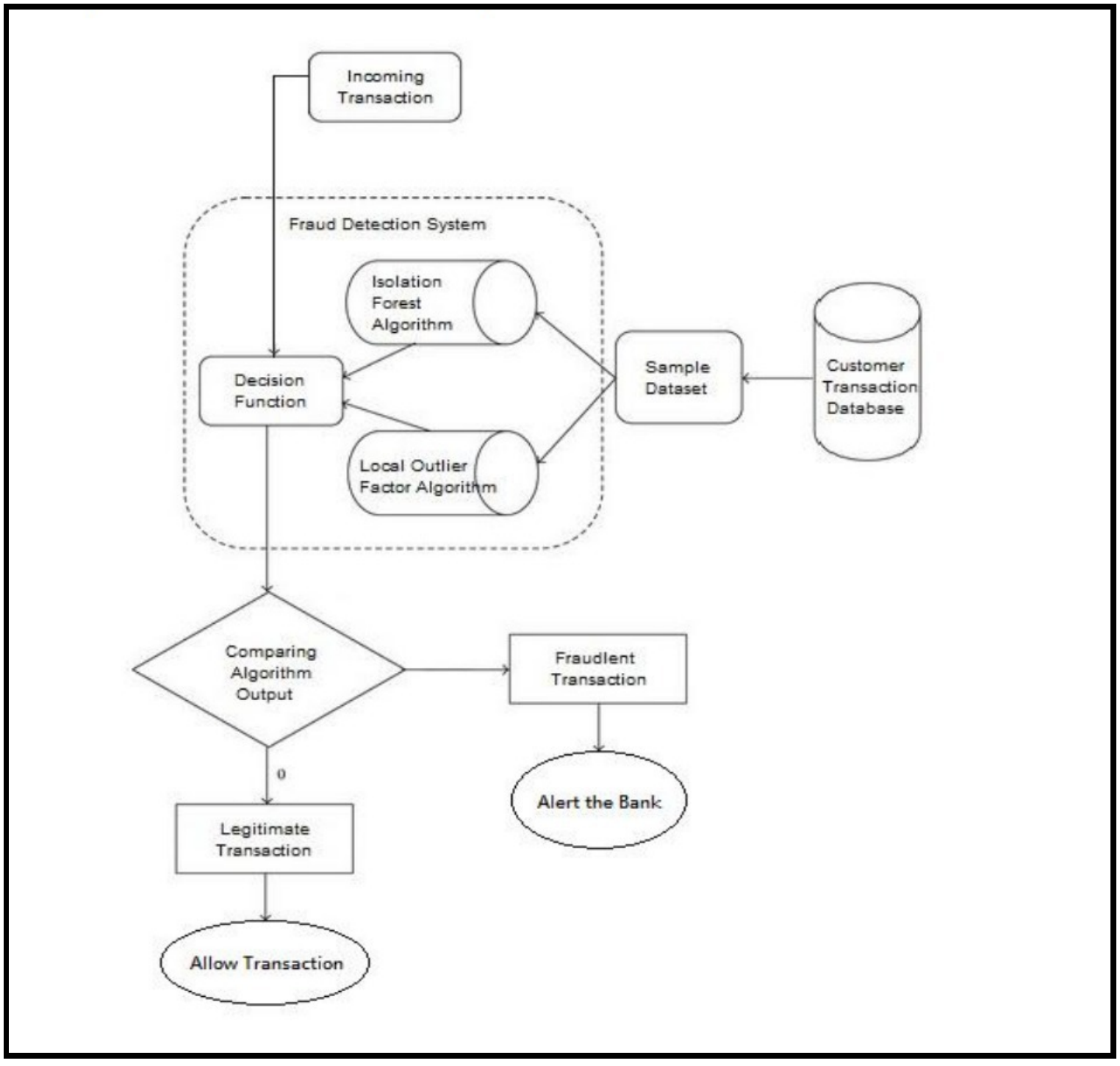
6. R Language

R is a programming language and free software environment for statistical computing and graphics supported by the R Core Team and the R Foundation for Statistical Computing. It is widely used among statisticians and data miners for developing statistical software and data analysis.

The project is to be developed in R language which should get executed in a Windows OS/ Mac/ Linux. R studio should be used as an IDE.

Design

Workflow Diagram

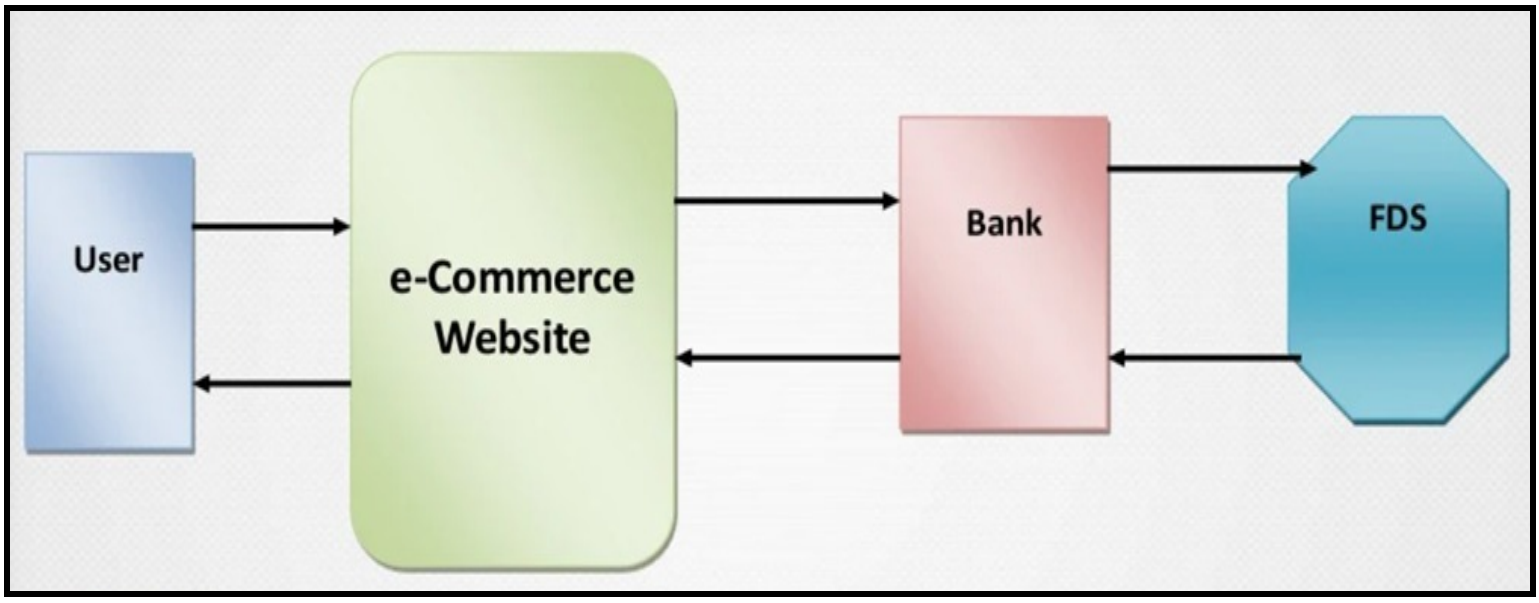


In this Diagram, the flow of the user transaction is shown:

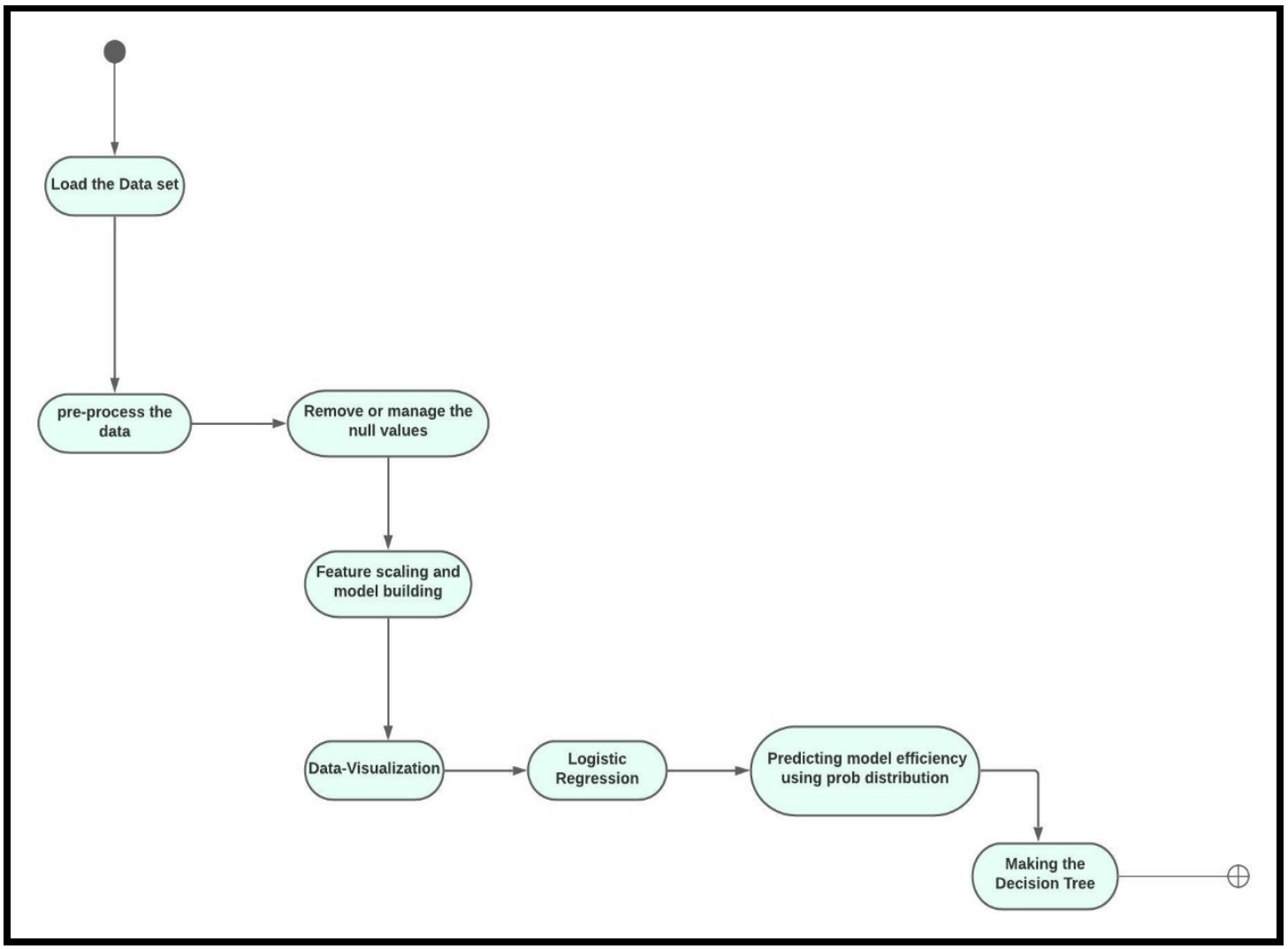
- When user makes a translation from their credit cards through any E-commerce websites, first the request goes to the bank
- Bank sends the details to the Fraud detection system, where a trained model with various algorithms are ready to test the incoming data
- The fraud detection system then analyses if the request made is legitimate or not
- if yes, the bank initiates the transaction
- else, the bank fails the transaction and blocks the card

➔ One more better explanation of this process is shown below by the

Block Diagram:



Flow Chart



This flowchart explains the overall flow of our model explained below:

1. Loading the data:

   We first import the libraries required for the project then we import our data set to R studio.

2. Pre-Processing the Data:

   In this step we basically transform the raw data into an understandable format. Because the data can be often incomplete, inconsistent, and/or lacking in certain behaviors or trends, and is likely to contain many errors, so it is important to remove the incomplete and inconsistent data.

3. Remove the Null Values:

   We first search all the null values that are in the dataset and remove corresponding rows because the null values create the data inconsistency and inaccuracy.

4. Feature Scaling and model building:

   Normalization is used when we want to bound our values between two numbers, typically, between [0,1] or [-1,1]. While Standardization transforms the data to have zero mean and a variance of 1, they make our data unitless. Refer to the below diagram, which shows how data looks after scaling in the X-Y plane.

5. Data Visualization:

   Data visualization is the graphical representation of information and data. By using visual elements like charts, graphs, and maps, data visualization tools provide an accessible way to see and understand trends, outliers, and patterns in data. In our project data visualization plays a major role, we create different graphs to analyse the data and different traits in it.

6. Logistic Regression:

   Logistic regression is a process of modeling the probability of a discrete outcome given an input variable. The most common logistic regression models a binary outcome; something that can take two values such as true/false, yes/no, and so on. Multinomial logistic regression can model scenarios where there are more than two possible discrete outcomes. Logistic regression is a useful analysis method for classification problems, where you are trying to determine if a new sample fits best into a category. As aspects of cyber security are classification problems, such as attack detection, logistic regression is a useful analytic technique.We use logical regression to create the summary of logistic models and to point graphs of logistic models, and we will make ROC curves.

7. Predicting model efficiency using problem distribution:

   Predictive modeling is a statistical technique using machine learning and data mining to predict and forecast likely future outcomes with the aid of historical and existing data. It works by analyzing current and historical data and projecting what it learns on a model generated to forecast likely outcomes. Predictive modeling can be used to predict just about anything, from TV ratings and a customer's next purchase to credit risks and corporate earnings.

8. Making the Decision tree:

   At last, we will make a decision tree. A decision tree is a decision support tool that uses a tree-like model of decisions and their possible consequences, including chance event outcomes, resource costs, and utility. It is one way to display an algorithm that only contains conditional control statements; we use it to see when our model is successful or not.

Methodology

The approach that this paper proposes, uses the latest machine learning algorithms to detect anomalous activities, called outliers.

First of all, we obtained our dataset from Kaggle, a data analysis website which provides datasets. Inside this dataset, there are 31 columns out of which 28 are named as v1-v28 to protect sensitive data. The other columns represent Time, Amount and Class. Time shows the time gap between the first transaction and the following one. Amount is the amount of money transacted. Class 0 represents a valid transaction and 1 represents a fraudulent one. We plot different graphs to check for inconsistencies in the dataset and to visually comprehend it:

Then we will explore the data, data exploration is an approach similar to initial data analysis, whereby a data analyst uses visual exploration to understand what is in a dataset and the characteristics of the data, rather than through traditional data management systems, we helps users to make better decisions on where to dig deeper into the data and to take a broad understanding of the business when asking more detailed questions later. With a user-friendly interface, anyone across an organization can familiarize themselves with the data, discover patterns, and generate thoughtful questions that may spur on deeper, valuable analysis.

After data exploration comes Data cleaning, and data cleaning is the process of fixing or removing incorrect, corrupted, incorrectly formatted, duplicate, or incomplete data within a dataset. When combining multiple data sources, there are many opportunities

for data to be duplicated or mislabeled. If data is incorrect, outcomes and algorithms are unreliable, even though they may look correct.

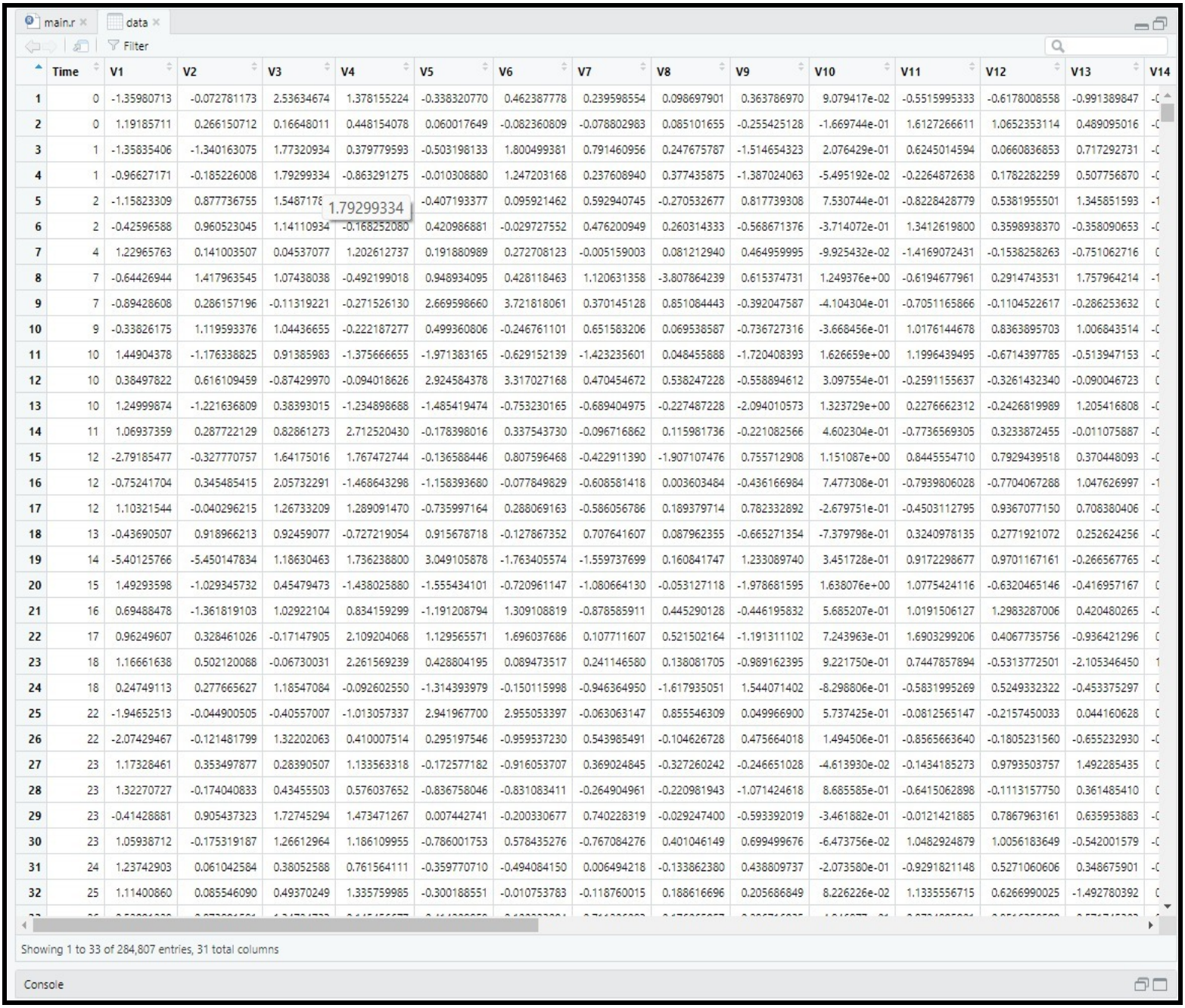
Then the preprocessing of data is finished, we will set prediction parameters and start with Data visualization, in data visualization we will create different graphs to see where are the differences and analyse if we find a lead, then we will implement logistic regression to make further predictions.

And finally we will test the data, and calculate the efficiency of our model and conclude with the help of the decision tree.

Implementation

1. DATA EXPLORATION:

We have imported the dataset from kaggle, the preview of our data set is shown below:



main.r × data ×

Filter

	Time	V1	V2	V3	V4	V5	V6	V7	V8	V9	V10	V11	V12	V13	V14
1	0	-1.35980713	-0.072781173	2.53634674	1.378155224	-0.338320770	0.462387778	0.239598554	0.098697901	0.363786970	9.079417e-02	-0.5515995333	-0.6178008558	-0.991389847	-0
2	0	1.19185711	0.266150712	0.16648011	0.448154078	0.060017649	-0.082360809	-0.078802983	0.085101655	-0.255425128	-1.669744e-01	1.6127266611	1.0652353114	0.489095016	-0
3	1	-1.35835406	-1.340163075	1.77320934	0.379779593	-0.503198133	1.800499381	0.791460956	0.247675787	-1.514654323	2.076429e-01	0.6245014594	0.0660836853	0.717292731	-0
4	1	-0.96627171	-0.185226008	1.79299334	-0.863291275	-0.010308880	1.247203168	0.237608940	0.377435875	-1.387024063	-5.495192e-02	-0.2264872638	0.1782282259	0.507756870	-0
5	2	-1.15823309	0.877736755	1.5487178	1.79299334	-0.407193377	0.095921462	0.592940745	-0.270532677	0.817739308	7.530744e-01	-0.8228428779	0.5381955501	1.345851593	-1
6	2	-0.42596588	0.960523045	1.14110934	-0.168252080	0.420986881	-0.029727552	0.476200949	0.260314333	-0.568671376	-3.714072e-01	1.3412619800	0.3598938370	-0.358090653	-0
7	4	1.22965763	0.141003507	0.04537077	1.202612737	0.191880989	0.272708123	-0.005159003	0.081212940	0.464959995	-9.925432e-02	-1.4169072431	-0.1538258263	-0.751062716	0
8	7	-0.64426944	1.417963545	1.07438038	-0.492199018	0.948934095	0.428118463	1.120631358	-3.807864239	0.615374731	1.249376e+00	-0.6194677961	0.2914743531	1.757964214	-1
9	7	-0.89428608	0.286157196	-0.11319221	-0.271526130	2.669598660	3.721818061	0.370145128	0.851084443	-0.392047587	-4.104304e-01	-0.7051165866	-0.1104522617	-0.286253632	0
10	9	-0.33826175	1.119593376	1.04436655	-0.222187277	0.499360806	-0.246761101	0.651583206	0.069538587	-0.736727316	-3.668456e-01	1.0176144678	0.8363895703	1.006843514	-0
11	10	1.44904378	-1.176338825	0.91385983	-1.375666655	-1.971383165	-0.629152139	-1.423235601	0.048455888	-1.720408393	1.626659e+00	1.1996439495	-0.6714397785	-0.513947153	-0
12	10	0.38497822	0.616109459	-0.87429970	-0.094018626	2.924584378	3.317027168	0.470454672	0.538247228	-0.558894612	3.097554e-01	-0.2591155637	-0.3261432340	-0.090046723	0
13	10	1.24999874	-1.221636809	0.38393015	-1.234898688	-1.485419474	-0.753230165	-0.689404975	-0.227487228	-2.094010573	1.323729e+00	0.2276662312	-0.2426819989	1.205416808	-0
14	11	1.06937359	0.287722129	0.82861273	2.712520430	-0.178398016	0.337543730	-0.096716862	0.115981736	-0.221082566	4.602304e-01	-0.7736569305	0.3233872455	-0.011075887	-0
15	12	-2.79185477	-0.327770757	1.64175016	1.767472744	-0.136588446	0.807596468	-0.422911390	-1.907107476	0.755712908	1.151087e+00	0.8445554710	0.7929439518	0.370448093	-0
16	12	-0.75241704	0.345485415	2.05732291	-1.468643298	-1.158393680	-0.077849829	-0.608581418	0.003603484	-0.436166984	7.477308e-01	-0.7939806028	-0.7704067288	1.047626997	-1
17	12	1.10321544	-0.040296215	1.26733209	1.289091470	-0.735997164	0.288069163	-0.586056786	0.189379714	0.782332892	-2.679751e-01	-0.4503112795	0.9367077150	0.708380406	-0
18	13	-0.43690507	0.918966213	0.92459077	-0.727219054	0.915678718	-0.127867352	0.707641607	0.087962355	-0.665271354	-7.379798e-01	0.3240978135	0.2771921072	0.252624256	-0
19	14	-5.40125766	-5.450147834	1.18630463	1.736238800	3.049105878	-1.763405574	-1.559737699	0.160841747	1.233089740	3.451728e-01	0.9172298677	0.9701167161	-0.266567765	-0
20	15	1.49293598	-1.029345732	0.45479473	-1.438025880	-1.555434101	-0.720961147	-1.080664130	-0.053127118	-1.978681595	1.638076e+00	1.0775424116	-0.6320465146	-0.416957167	0
21	16	0.69488478	-1.361819103	1.02922104	0.834159299	-1.191208794	1.309106819	-0.878585911	0.445290128	-0.446195832	5.685207e-01	1.0191506127	1.2983287006	0.420480265	-0
22	17	0.96249607	0.328461026	-0.17147905	2.109204068	1.129565571	1.696037686	0.107711607	0.521502164	-1.191311102	7.243963e-01	1.6903299206	0.4067735756	-0.936421296	0
23	18	1.16661638	0.502120088	-0.06730031	2.261569239	0.428604195	0.089473517	0.241146580	0.138081705	-0.989162395	9.221750e-01	0.7447857894	-0.5313772501	-2.105346450	1
24	18	0.24749113	0.277665627	1.18547084	-0.092602550	-1.314393979	-0.150115998	-0.946364950	-1.617935051	1.544071402	-8.298806e-01	-0.5831995269	0.5249332322	-0.453375297	0
25	22	-1.94652513	-0.044900505	-0.40557007	-1.013057337	2.941967700	2.955053397	-0.063063147	0.855546309	0.049966900	5.737425e-01	-0.0812565147	-0.2157450033	0.044160628	0
26	22	-2.07429467	-0.121481799	1.32202063	0.410007514	0.295197546	-0.959537230	0.543985491	-0.104626728	0.475664018	1.494506e-01	-0.8565663640	-0.1805231560	-0.655232930	-0
27	23	1.17328461	0.353497877	0.28390507	1.133563318	-0.172577182	-0.916053707	0.369024845	-0.327260242	-0.246651028	-4.613930e-02	-0.1434185273	0.9793503757	1.492285435	0
28	23	1.32270727	-0.174040833	0.43455503	0.576037652	-0.836758046	-0.831083411	-0.264904961	-0.220981943	-1.071424618	8.685585e-01	-0.6415062898	-0.1113157750	0.361485410	0
29	23	-0.41428881	0.905437323	1.72745294	1.473471267	0.007442741	-0.200330677	0.740228319	-0.029247400	-0.593392019	-3.461882e-01	-0.0121421885	0.7867963161	0.635953883	-0
30	23	1.05938712	-0.175319187	1.26612964	1.186109955	-0.786001753	0.578435276	-0.767084276	0.401046149	0.699499676	-6.473756e-02	1.0482924879	1.0056183649	-0.542001579	-0
31	24	1.23742903	0.061042584	0.38052588	0.761564111	-0.359770710	-0.494084150	0.006494218	-0.133862380	0.438809737	-2.073580e-01	-0.9291821148	0.5271060606	0.348675901	-0
32	25	1.11400860	0.085546090	0.49370249	1.335759985	-0.300188551	-0.010753783	-0.118760015	0.188616696	0.205686849	8.226226e-02	1.1335556715	0.6266990025	-1.492780392	0

Showing 1 to 33 of 284,807 entries, 31 total columns

Console

About the Dataset:

The dataset contains transactions made by credit cards in September 2013 by European cardholders.

This dataset presents transactions that occurred in two days, where we have 492 frauds out of 284,807 transactions. The dataset is highly unbalanced, the positive class (frauds) account for 0.172% of all transactions.

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

Given the class imbalance ratio, we recommend measuring the accuracy using the Area Under the Precision-Recall Curve (AUPRC). Confusion matrix accuracy is not meaningful for unbalanced classification.

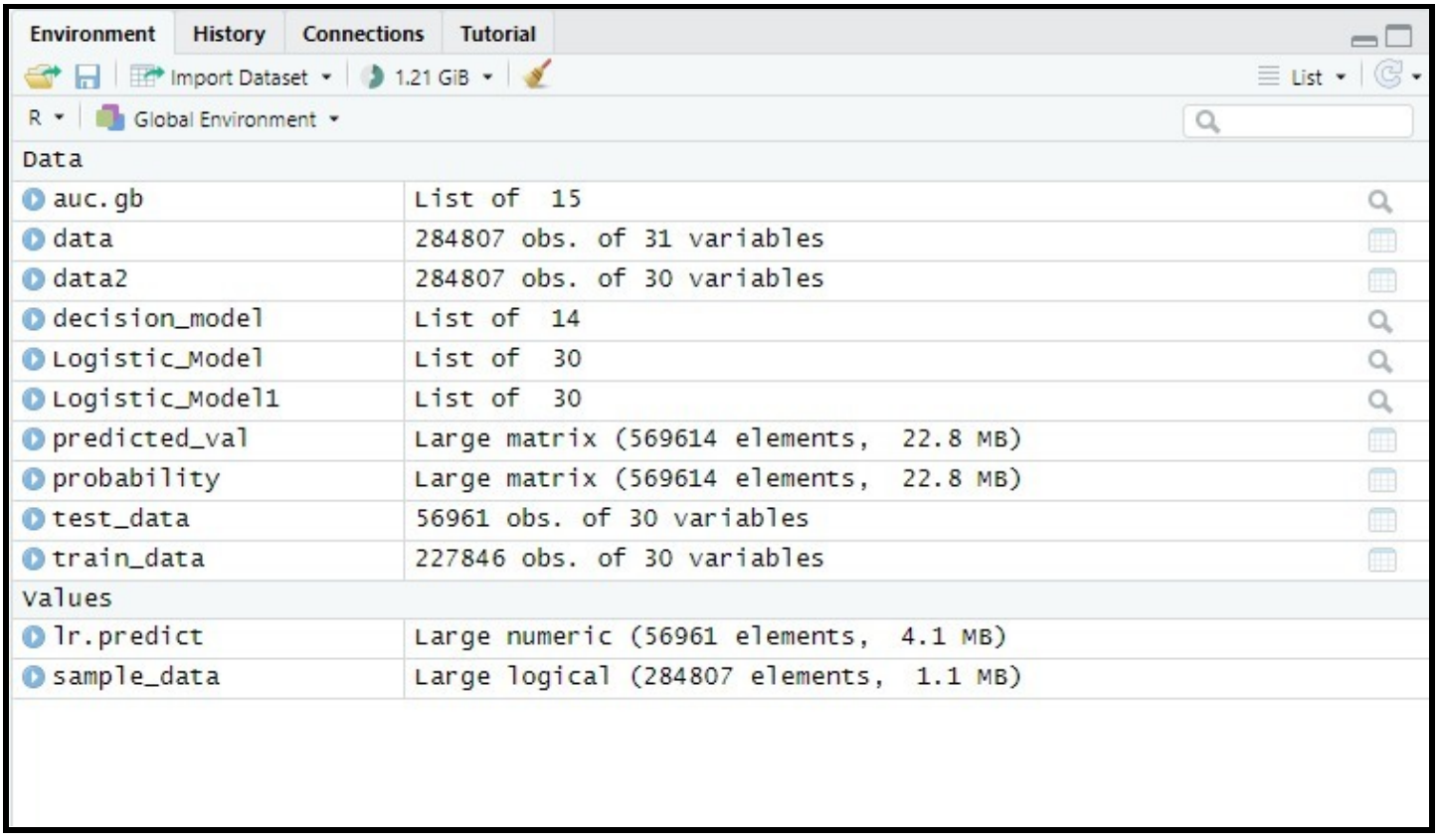
About the variables:

We have divided our data into 2 parts: Test data and Train data.

Train data is used to train the model how to detect the fraud and generally consists of 80% of the dataset.

Test data is used to test the model if it is correctly able to detect the fraud or not, we have used the rest 20% of the dataset in the train data set.

We also have created three lists of decision and logistic models for data visualization purposes.



Exploring the Data:

After importing we have looked at the data set, like :

- Summarised the data.
- Calculated its mean, variance and Standard deviation.
- Interquartile Range.
- Looked at the head and tail of the data (first 5 and last5 rows of the data).
- Calculating the Fraud and Non-Fraud ratio.

```
> data <- read.csv("C:/Users/divya/Desktop/DS_mini_Project_(2)/creditcard.csv")
>
> # Data Exploration
> data.table(data)
          Time          V1          V2         V3         V4          V5          V6          V7          V8         V9         V10
    1:       0  -1.3598071 -0.07278117  2.5363467  1.3781552 -0.33832077  0.46238778  0.23959855  0.09869790  0.3637870  0.09079417
    2:       0   1.1918571  0.26615071  0.1664801  0.4481541  0.06001765 -0.08236081 -0.07880298  0.08510165 -0.2554251 -0.16697441
    3:       1  -1.3583541 -1.34016307  1.7732093  0.3797796 -0.50319813  1.80049938  0.79146096  0.24767579 -1.5146543  0.20764287
    4:       1  -0.9662717 -0.18522601  1.7929933 -0.8632913 -0.01030888  1.24720317  0.23760894  0.37743587 -1.3870241 -0.05495192
    5:       2  -1.1582331  0.87773675  1.5487178  0.4030339 -0.40719338  0.09592146  0.59294075 -0.27053268  0.8177393  0.75307443
   ---
284803: 172786 -11.8811179 10.07178497 -9.8347835 -2.0666557 -5.36447278 -2.60683733 -4.91821543  7.30533402  1.9144283  4.35617041
284804: 172787  -0.7327887 -0.05508049  2.0350297 -0.7385886  0.86822940  1.05841527  0.02432970  0.29486870  0.5848000 -0.97592606
284805: 172788   1.9195650 -0.30125385 -3.2496398 -0.5578281  2.63051512  3.03126010 -0.29682653  0.70841718  0.4324540 -0.48478176
284806: 172788  -0.2404400  0.53048251  0.7025102  0.6897992 -0.37796113  0.62370772 -0.68617999  0.67914546  0.3920867 -0.39912565
284807: 172792  -0.5334125 -0.18973334  0.7033374 -0.5062712 -0.01254568 -0.64961669  1.57700625 -0.41465041  0.4861795 -0.91542665
              V11         V12         V13         V14         V15         V16         V17         V18         V19         V20
    1: -0.5515995 -0.61780086 -0.9913898 -0.31116935  1.46817697 -0.4704005  0.20797124  0.02579058  0.4039930  0.25141210
    2:  1.6127267  1.06523531  0.4890950 -0.14377230  0.63555809  0.4639170 -0.11480466 -0.18336127 -0.1457830 -0.06908314
    3:  0.6245015  0.06608369  0.7172927 -0.16594592  2.34586495 -2.8900832  1.10996938 -0.12135931 -2.2618571  0.52497973
    4: -0.2264873  0.17822823  0.5077569 -0.28792375 -0.63141812 -1.0596472 -0.68409279  1.96577500 -1.2326220 -0.20803778
    5: -0.8228429  0.53819555  1.3458516 -1.11966983  0.17512113 -0.4514492 -0.23703324 -0.03819479  0.8034869  0.40854236
   ---
284803: -1.5931053  2.71194079 -0.6892556  4.62694203 -0.92445871  1.1076406  1.99169111  0.51063233 -0.6829197  1.47582913
284804: -0.1501888  0.91580191  1.2147558 -0.67514296  1.16493091 -0.7117573 -0.02569286 -1.22117886 -1.5455561  0.05961590
284805:  0.4116137  0.06311886 -0.1836987 -0.51060184  1.32928351  0.1407160  0.31350179  0.39565248 -0.5772518  0.00139597
284806: -1.9338488 -0.96288614 -1.0420817  0.44962444  1.96256312 -0.6085771  0.50992846  1.11398059  2.8978488  0.12743352
284807: -1.0404583 -0.03151305 -0.1880929 -0.08431647  0.04133346 -0.3026201 -0.66037665  0.16742993 -0.2561169  0.38294810
               V21          V22         V23          V24         V25         V26          V27         V28 Amount Class
    1: -0.018306778  0.277837576 -0.11047391  0.066928075  0.1285394 -0.1891148  0.133558377 -0.02105305 149.62     0
    2: -0.225775248 -0.638671953  0.10128802 -0.339846476  0.1671704  0.1258945 -0.008983099  0.01472417   2.69     0
    3:  0.247998153  0.771679402  0.90941226 -0.689280956 -0.3276418 -0.1390966 -0.055352794 -0.05975184 378.66     0
    4: -0.108300452  0.005273597 -0.19032052 -1.175575332  0.6473760 -0.2219288  0.062722849  0.06145763 123.50     0
    5: -0.009430697  0.798278495 -0.13745808  0.141266984 -0.2060096  0.5022922  0.219422230  0.21515315  69.99     0
   ---
284803:  0.213454108  0.111863736  1.01447990 -0.509348453  1.4368069  0.2500343  0.943651172  0.82373096   0.77     0
284804:  0.214205342  0.924383585  0.01246304 -1.016225669 -0.6066240 -0.3952551  0.068472470 -0.05352739  24.79     0
284805:  0.232045036  0.578229010 -0.03750086  0.640133881  0.2657455 -0.0873706  0.004454772 -0.02656083  67.88     0
284806:  0.265244916  0.800048741 -0.16329794  0.123205244 -0.5691589  0.5466685  0.108820735  0.10453282  10.00     0
284807:  0.261057331  0.643078438  0.37677701  0.008797379 -0.4736487 -0.8182671 -0.002415309  0.01364891 217.00     0
>
```

```
> table(data$Class)

     0      1
284315    492
> names(data)
 [1] "Time"   "V1"     "V2"     "V3"     "V4"     "V5"     "V6"     "V7"     "V8"     "V9"     "V10"    "V11"    "V12"    "V13"
[15] "V14"    "V15"    "V16"    "V17"    "V18"    "V19"    "V20"    "V21"    "V22"    "V23"    "V24"    "V25"    "V26"    "V27"
[29] "V28"    "Amount" "Class"
>
> # Summary of Amount
> summary(data$Amount)
    Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
    0.00     5.60    22.00    88.35    77.17 25691.16
> # Standard deviations
> sd(data$Amount)
[1] 250.1201
> # Inter-Quartile Range values
> IQR(data$Amount)
[1] 71.565
> # Variance of the data frame
> var(data$Amount)
[1] 62560.07
>
> # Data Manipulation
> data$Amount <- scale(data$Amount)  #data normalization
> # R, C
> data2 <- data[,-c(1)]  #dropping the useless time column
```

```
> # Statistical Analysis of the data
> summary(data)
      Time              V1                    V2                   V3                  V4                   V5
 Min.   :     0   Min.   :-56.40751   Min.   :-72.71573   Min.   :-48.3256   Min.   :-5.68317   Min.   :-113.74331
 1st Qu.: 54202   1st Qu.: -0.92037   1st Qu.: -0.59855   1st Qu.: -0.8904   1st Qu.:-0.84864   1st Qu.:  -0.69160
 Median : 84692   Median :  0.01811   Median :  0.06549   Median :  0.1799   Median :-0.01985   Median :  -0.05434
 Mean   : 94814   Mean   :  0.00000   Mean   :  0.00000   Mean   :  0.0000   Mean   : 0.00000   Mean   :   0.00000
 3rd Qu.:139321   3rd Qu.:  1.31564   3rd Qu.:  0.80372   3rd Qu.:  1.0272   3rd Qu.: 0.74334   3rd Qu.:   0.61193
 Max.   :172792   Max.   :  2.45493   Max.   : 22.05773   Max.   :  9.3826   Max.   :16.87534   Max.   :  34.80167
       V6                 V7                   V8                   V9                  V10                  V11
 Min.   :-26.1605   Min.   :-43.5572   Min.   :-73.21672   Min.   :-13.43407   Min.   :-24.58826   Min.   :-4.79747
 1st Qu.: -0.7683   1st Qu.: -0.5541   1st Qu.: -0.20863   1st Qu.: -0.64310   1st Qu.: -0.53543   1st Qu.:-0.76249
 Median : -0.2742   Median :  0.0401   Median :  0.02236   Median : -0.05143   Median : -0.09292   Median :-0.03276
 Mean   :  0.0000   Mean   :  0.0000   Mean   :  0.00000   Mean   :  0.00000   Mean   :  0.00000   Mean   : 0.00000
 3rd Qu.:  0.3986   3rd Qu.:  0.5704   3rd Qu.:  0.32735   3rd Qu.:  0.59714   3rd Qu.:  0.45392   3rd Qu.: 0.73959
 Max.   : 73.3016   Max.   :120.5895   Max.   : 20.00721   Max.   : 15.59500   Max.   : 23.74514   Max.   :12.01891
      V12                V13                  V14                  V15                  V16                  V17
 Min.   :-18.6837   Min.   :-5.79188   Min.   :-19.2143   Min.   :-4.49894   Min.   :-14.12985   Min.   :-25.16280
 1st Qu.: -0.4056   1st Qu.:-0.64854   1st Qu.: -0.4256   1st Qu.:-0.58288   1st Qu.: -0.46804   1st Qu.: -0.48375
 Median :  0.1400   Median :-0.01357   Median :  0.0506   Median : 0.04807   Median :  0.06641   Median : -0.06568
 Mean   :  0.0000   Mean   : 0.00000   Mean   :  0.0000   Mean   : 0.00000   Mean   :  0.00000   Mean   :  0.00000
 3rd Qu.:  0.6182   3rd Qu.: 0.66251   3rd Qu.:  0.4931   3rd Qu.: 0.64882   3rd Qu.:  0.52330   3rd Qu.:  0.39968
 Max.   :  7.8484   Max.   : 7.12688   Max.   : 10.5268   Max.   : 8.87774   Max.   : 17.31511   Max.   :  9.25353
      V18                V19                  V20                  V21                  V22                  V23
 Min.   :-9.498746   Min.   :-7.213527   Min.   :-54.49772   Min.   :-34.83038   Min.   :-10.933144   Min.   :-44.80774
 1st Qu.:-0.498850   1st Qu.:-0.456299   1st Qu.: -0.21172   1st Qu.: -0.22839   1st Qu.: -0.542350   1st Qu.: -0.16185
 Median :-0.003636   Median : 0.003735   Median : -0.06248   Median : -0.02945   Median :  0.006782   Median : -0.01119
 Mean   : 0.000000   Mean   : 0.000000   Mean   :  0.00000   Mean   :  0.00000   Mean   :  0.000000   Mean   :  0.00000
 3rd Qu.: 0.500807   3rd Qu.: 0.458949   3rd Qu.:  0.13304   3rd Qu.:  0.18638   3rd Qu.:  0.528554   3rd Qu.:  0.14764
 Max.   : 5.041069   Max.   : 5.591971   Max.   : 39.42090   Max.   : 27.20284   Max.   : 10.503090   Max.   : 22.52841
      V24                V25                  V26                  V27                  V28                Amount
 Min.   :-2.83663   Min.   :-10.29540   Min.   :-2.60455   Min.   :-22.565679   Min.   :-15.43008   Min.   :    0.00
 1st Qu.:-0.35459   1st Qu.: -0.31715   1st Qu.:-0.32698   1st Qu.: -0.070840   1st Qu.: -0.05296   1st Qu.:    5.60
 Median : 0.04098   Median :  0.01659   Median :-0.05214   Median :  0.001342   Median :  0.01124   Median :   22.00
 Mean   : 0.00000   Mean   :  0.00000   Mean   : 0.00000   Mean   :  0.000000   Mean   :  0.00000   Mean   :   88.35
 3rd Qu.: 0.43953   3rd Qu.:  0.35072   3rd Qu.: 0.24095   3rd Qu.:  0.091045   3rd Qu.:  0.07828   3rd Qu.:   77.17
 Max.   : 4.58455   Max.   :  7.51959   Max.   : 3.51735   Max.   : 31.612198   Max.   : 33.84781   Max.   :25691.16
     Class
 Min.   :0.000000
 1st Qu.:0.000000
 Median :0.000000
 Mean   :0.001728
 3rd Qu.:0.000000
 Max.   :1.000000
```

2. DATA CLEANING AND PREPROCESSING:

Following are done in this step:-

- Normalised the data.
- Setting a random number of seeds.
- We removed the Time columns since we don't need it anymore.
- Then we will divide our data into 2 parts Training data and Testing data, in the ratio 1:8.

```
> head(data2)
         V1          V2        V3         V4          V5          V6          V7          V8         V9         V10        V11
1 -1.3598071 -0.07278117 2.5363467  1.3781552 -0.33832077  0.46238778  0.23959855  0.09869790  0.3637870  0.09079417 -0.5515995
2  1.1918571  0.26615071 0.1664801  0.4481541  0.06001765 -0.08236081 -0.07880298  0.08510165 -0.2554251 -0.16697441  1.6127267
3 -1.3583541 -1.34016307 1.7732093  0.3797796 -0.50319813  1.80049938  0.79146096  0.24767579 -1.5146543  0.20764287  0.6245015
4 -0.9662717 -0.18522601 1.7929933 -0.8632913 -0.01030888  1.24720317  0.23760894  0.37743587 -1.3870241 -0.05495192 -0.2264873
5 -1.1582331  0.87773675 1.5487178  0.4030339 -0.40719338  0.09592146  0.59294075 -0.27053268  0.8177393  0.75307443 -0.8228429
6 -0.4259659  0.96052304 1.1411093 -0.1682521  0.42098688 -0.02972755  0.47620095  0.26031433 -0.5686714 -0.37140720  1.3412620
         V12        V13        V14        V15        V16         V17         V18         V19         V20          V21          V22
1 -0.61780086 -0.9913898 -0.3111694  1.4681770 -0.4704005  0.20797124  0.02579058  0.40399296  0.25141210 -0.018306778  0.277837576
2  1.06523531  0.4890950 -0.1437723  0.6355581  0.4639170 -0.11480466 -0.18336127 -0.14578304 -0.06908314 -0.225775248 -0.638671953
3  0.06608369  0.7172927 -0.1659459  2.3458649 -2.8900832  1.10996938 -0.12135931 -2.26185710  0.52497973  0.247998153  0.771679402
4  0.17822823  0.5077569 -0.2879237 -0.6314181 -1.0596472 -0.68409279  1.96577500 -1.23262197 -0.20803778 -0.108300452  0.005273597
5  0.53819555  1.3458516 -1.1196698  0.1751211 -0.4514492 -0.23703324 -0.03819479  0.80348692  0.40854236 -0.009430697  0.798278495
6  0.35989384 -0.3580907 -0.1371337  0.5176168  0.4017259 -0.05813282  0.06865315 -0.03319379  0.08496767 -0.208253515 -0.559824796
         V23         V24        V25        V26          V27         V28      Amount Class
1 -0.11047391  0.06692807  0.1285394 -0.1891148  0.133558377 -0.02105305  0.24496383     0
2  0.10128802 -0.33984648  0.1671704  0.1258945 -0.008983099  0.01472417 -0.34247394     0
3  0.90941226 -0.68928096 -0.3276418 -0.1390966 -0.055352794 -0.05975184  1.16068389     0
4 -0.19032052 -1.17557533  0.6473760 -0.2219288  0.062722849  0.06145763  0.14053401     0
5 -0.13745808  0.14126698 -0.2060096  0.5022922  0.219422230  0.21515315 -0.07340321     0
6 -0.02639767 -0.37142658 -0.2327938  0.1059148  0.253844225  0.08108026 -0.33855582     0
>
> set.seed(12)  # setting the random number seed
> library(caTools)
>
> sample_data <- sample.split(data2$Class,SplitRatio = 0.80)
>
> train_data <- subset(data2,sample_data==TRUE)
>
> test_data <- subset(data2,sample_data==FALSE)
>
> dim(train_data)  # Dimesions of the dataset
[1] 227846     30
> dim(test_data)
[1] 56961    30
>
> # Logistic Regression
> Logistic_Model <- glm(Class~., test_data, family= binomial())
```

3. DATA VISUALIZATION:

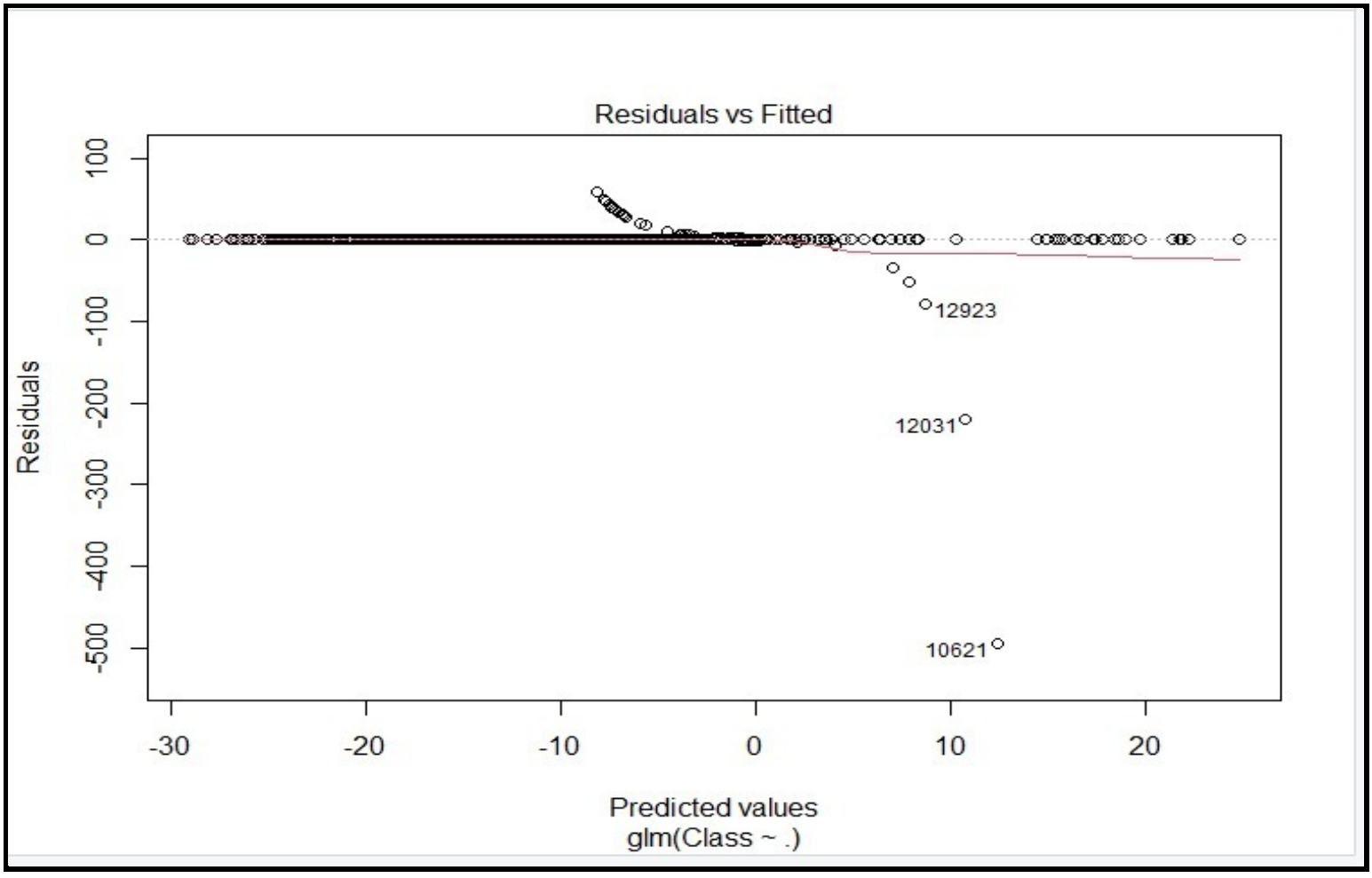
Data Visualization is processed by the help of Logistic regression:

About the Graphs:

Residuals vs Fitted:

Residuals here are the data which are not contributing to the cause of the dataset It is a scatter plot of residuals on the y axis and fitted values (estimated responses) on the x axis. The plot is used to detect non-linearity, unequal error variances, and outliers

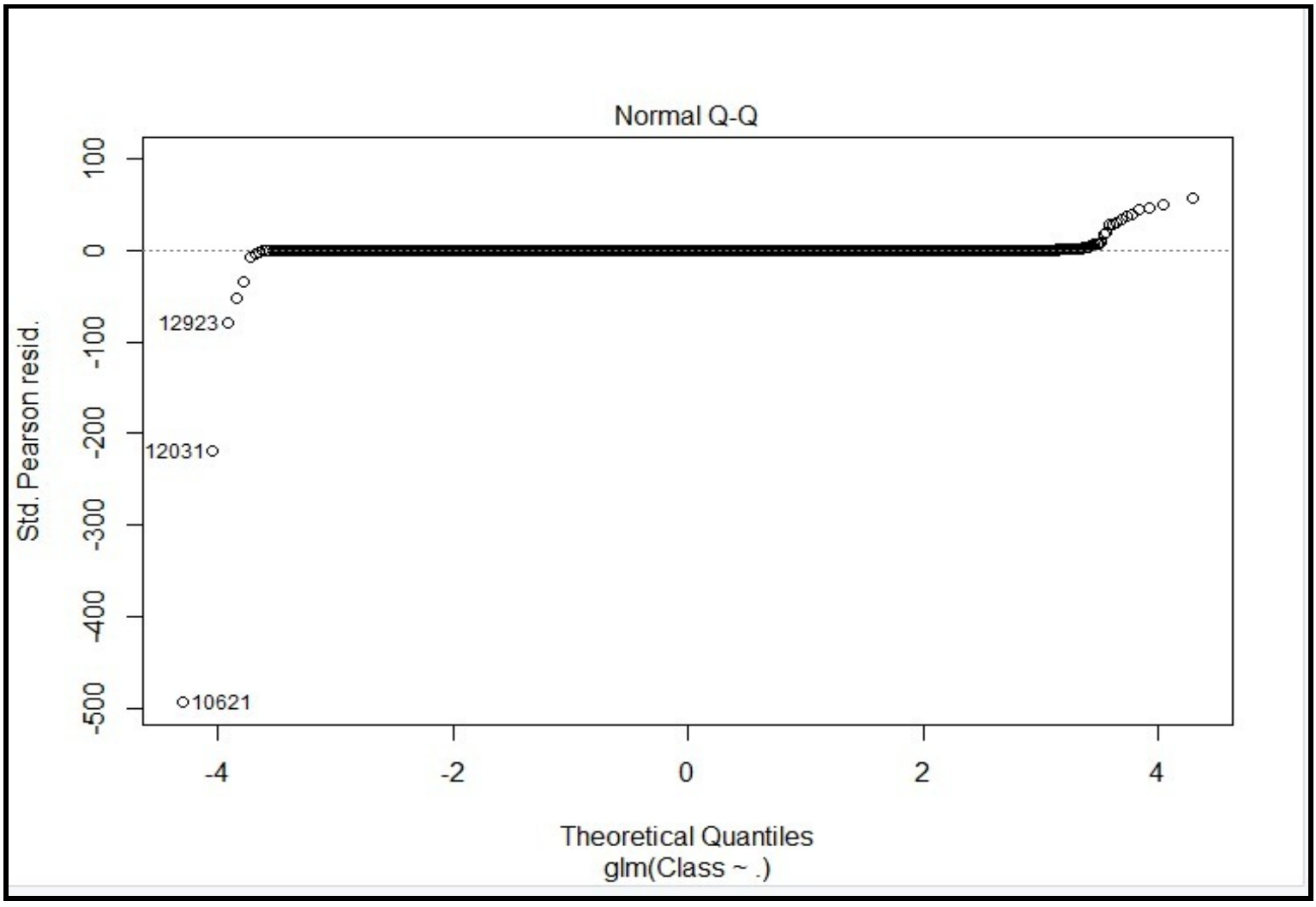
here the black dots denote residuals and red lines denote fitted.



Normal Q-Q:

A Q-Q plot is a scatterplot created by plotting two sets of quantiles against one another. If both sets of quantiles came from the same distribution, we should see the points forming a line that's roughly straight. Here's an example of a Normal Q-Q plot when both sets of quantiles truly come from Normal distributions.
Used to find the type of distribution for a random variable whether it be a Gaussian Distribution, Uniform Distribution, Exponential Distribution or even Pareto Distribution, etc. You can tell the type of distribution using the power of the Q-Q plot just by looking at the plot.

The straight line tells us that our project is not part of a time series.



Scale-Location plot:

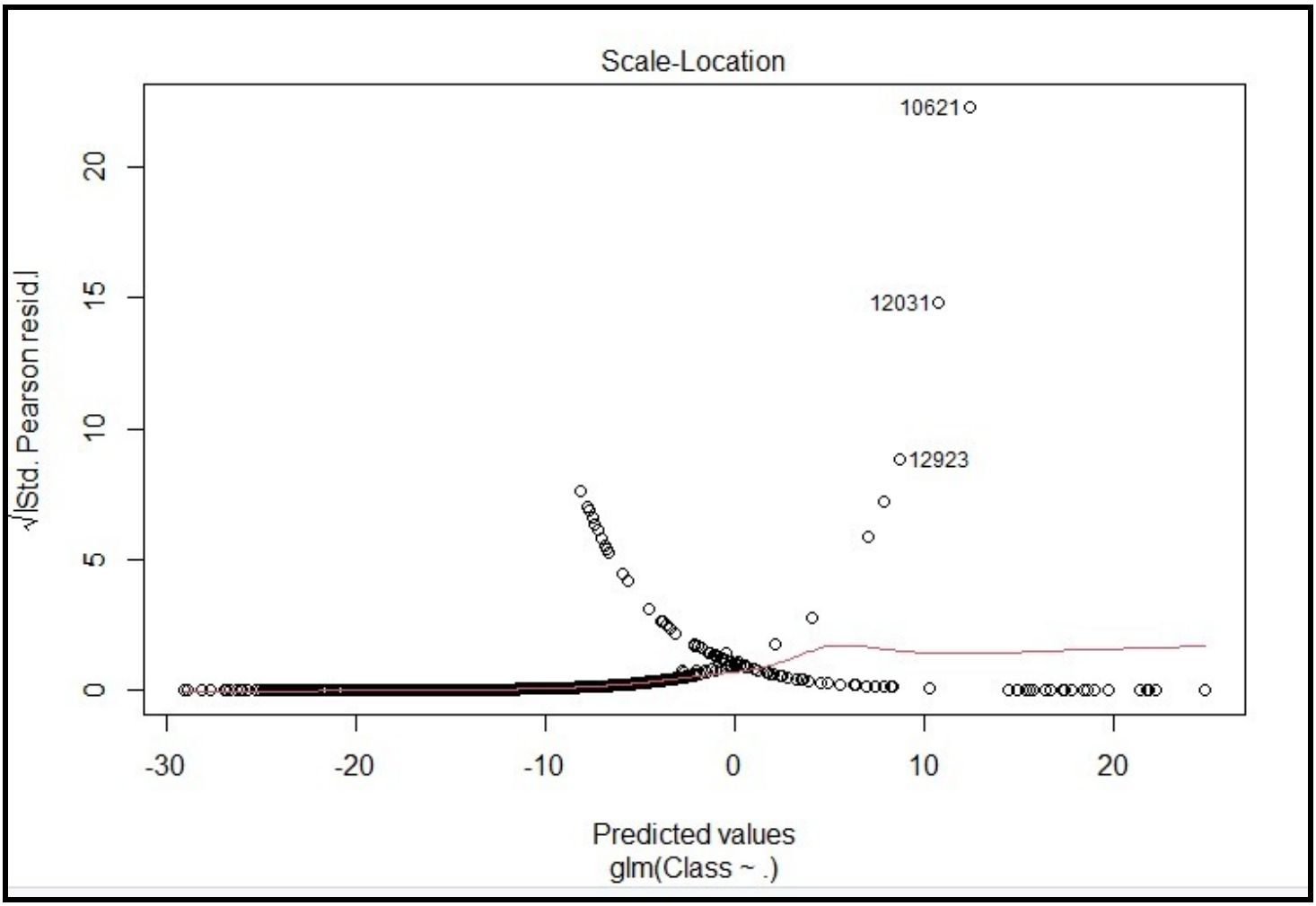
Scale-Location plot shows whether residuals are spread equally along the ranges of input variables (predictor). The assumption of equal variance (homoscedasticity) could also be checked with this plot. If we see a horizontal line with randomly spread points, it means that the model is good.

The scale-location plot is very similar to residuals vs fitted, but simplifies analysis of the homoskedasticity assumption. It takes the square root of the absolute value of standardized residuals instead of plotting the residuals themselves. Recall that homoscedasticity means constant variance in linear regression. More formally, in linear regression you have:

$$y = X\beta + \epsilon$$

In the x-scale we have predicted values and at the y-axis we have Absolute standard deviation.

From this diagram we will know that during the prediction of 0 and 1 how many times does it deviate from its S.D. or standard mean.



Residuals vs Leverage:

A residuals vs. leverage plot is a type of diagnostic plot that allows us to identify influential observations in a regression model.

Each observation from the dataset is shown as a single point within the plot. The x-axis shows the leverage of each point and the y-axis shows the standardized residual of each point.

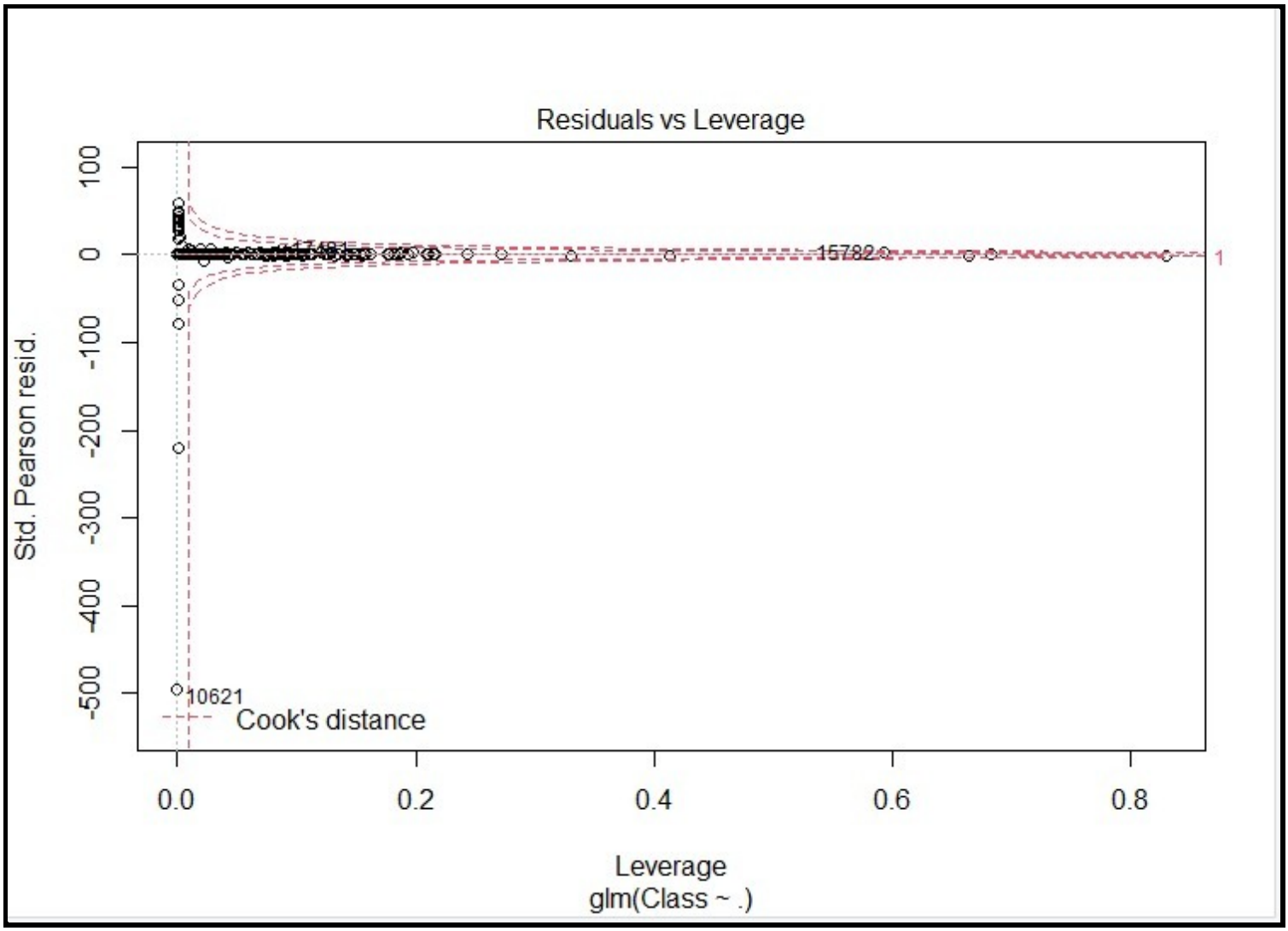
Leverage, refers to the extent to which the coefficients in the regression model would change if a particular observation was removed from the dataset.

Observations with high leverage have a strong influence on the coefficients in the regression model. If we remove these observations, the coefficients of the model would change noticeably.

Standardized residuals, refer to the standardized difference between a predicted value for an observation and the actual value of the observation.

It’s worth noting that an observation can have a high absolute value for a standardized residual, yet have a low value for leverage.

In this Plot if the values are greater than 0.5 then that is fraud also if the value exceeds the cook's distance in the y-axis it is called a fraud value.



4. LOGICAL REGRESSION:

We used a generalized logistic model function using binomial distribution family, to find the deviance residuals, and coefficients of every column. The generalized logistic function or curve, also known as Richards' curve, originally developed for growth modelling, is an extension of the logistic or sigmoid functions, allowing for more flexible S-shaped curves.

Here we are predicting model efficiency using probability efficiency.

```
> summary(Logistic_Model1)

Call:
glm(formula = Class ~ ., family = binomial(), data = train_data)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-4.8309  -0.0292  -0.0198  -0.0126   4.5591

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -8.650018   0.163791 -52.811  < 2e-16 ***
V1           0.077467   0.047334   1.637 0.101716
V2           0.028875   0.073701   0.392 0.695219
V3           0.046287   0.052957   0.874 0.382086
V4           0.698134   0.084232   8.288  < 2e-16 ***
V5           0.106152   0.077383   1.372 0.170130
V6          -0.145069   0.087943  -1.650 0.099027 .
V7          -0.101990   0.079155  -1.288 0.197579
V8          -0.177153   0.036961  -4.793 1.64e-06 ***
V9          -0.291020   0.125318  -2.322 0.020219 *
V10         -0.822565   0.109974  -7.480 7.45e-14 ***
V11         -0.001569   0.085970  -0.018 0.985442
V12          0.119036   0.100727   1.182 0.237300
V13         -0.334341   0.094144  -3.551 0.000383 ***
V14         -0.550566   0.070983  -7.756 8.75e-15 ***
V15         -0.054315   0.096127  -0.565 0.572046
V16         -0.126830   0.143468  -0.884 0.376679
V17         -0.045890   0.078951  -0.581 0.561070
V18         -0.044342   0.145346  -0.305 0.760308
V19          0.093877   0.108981   0.861 0.389014
V20         -0.410040   0.094754  -4.327 1.51e-05 ***
V21          0.339517   0.067448   5.034 4.81e-07 ***
V22          0.553776   0.144118   3.843 0.000122 ***
V23         -0.096101   0.073332  -1.311 0.190025
V24          0.145238   0.169197   0.858 0.390677
V25          0.014571   0.148022   0.098 0.921584
V26          0.230722   0.206031   1.120 0.262782
V27         -0.795025   0.140545  -5.657 1.54e-08 ***
V28         -0.285374   0.098342  -2.902 0.003710 **
Amount       0.232208   0.129465   1.794 0.072877 .
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 5799.1  on 227845  degrees of freedom
Residual deviance: 1766.0  on 227816  degrees of freedom
AIC: 1826

Number of Fisher Scoring iterations: 12

> # ROC Curve
> library(pROC)
```

5. CALCULATING EFFICIENCY:

Gaussian Statistics:

when we train the model, we get these parameters, learning graph and expected outcome and the statistics formed b/2 these two is called gaussian statistics

Confusion matrix:

ROC curve:

An ROC curve (receiver operating characteristic curve) is a graph showing the performance of a classification model at all classification thresholds. This curve plots two parameters:

- True Positive Rate
- False Positive Rate

True Positive Rate (TPR) is a synonym for recall and is therefore defined as follows:

TPR=TPTP+FN

False Positive Rate (FPR) is defined as follows:

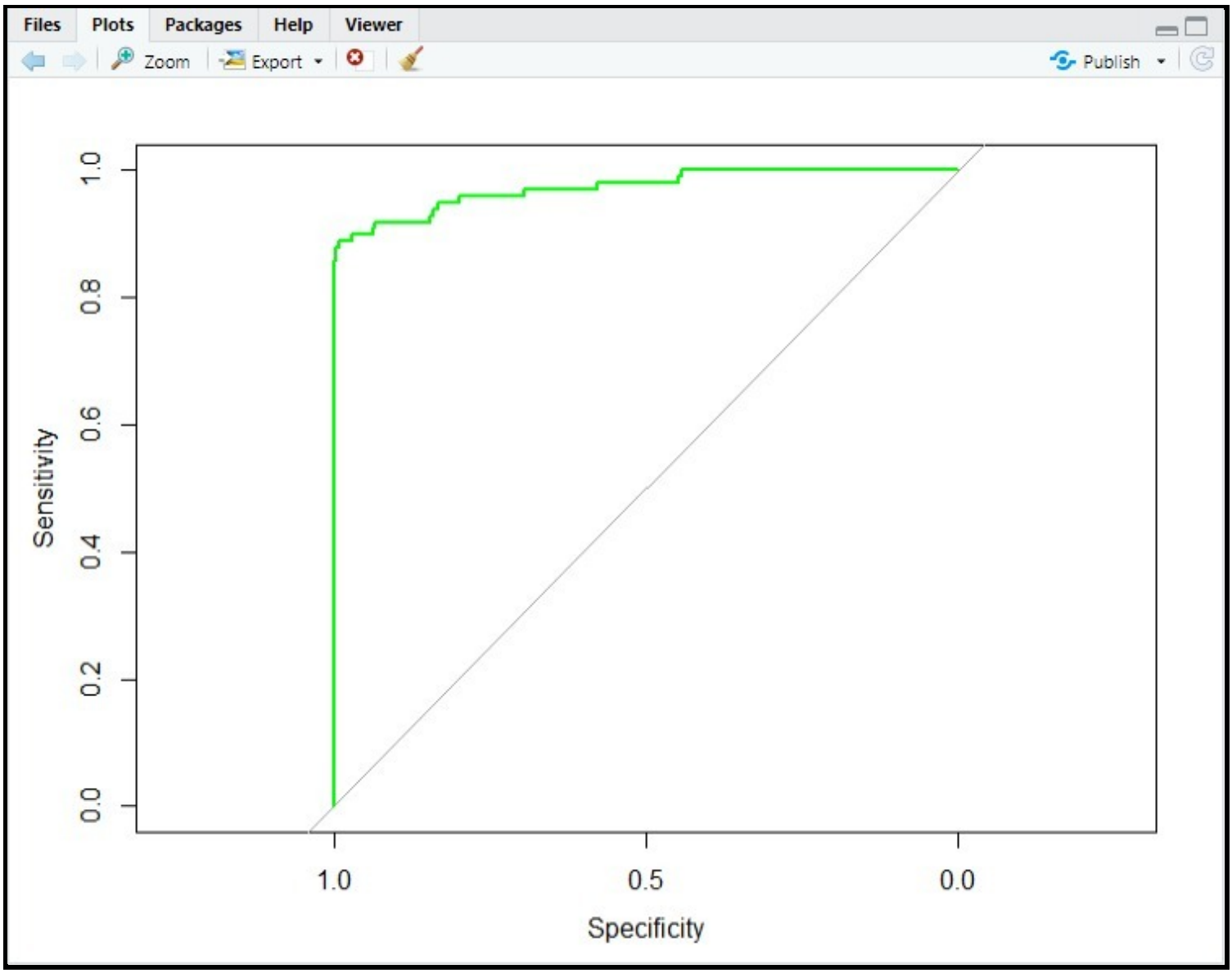
FPR=FPFP+TN

An ROC curve plots TPR vs. FPR at different classification thresholds. Lowering the classification threshold classifies more items as positive, thus increasing both False Positives and True Positives. The following figure shows a typical ROC curve.

We use ROC curve to see rate of the speed and accuracy of our model.

Also in this diagram we see that frequency and sensitivity of the graph is getting disturbed when coming to 1. And from 0-0.9 it's predicting correctly.

The efficiency percentage from this graph is nearly 90-95%.



6. DECISION TREE:

Decision tree is the most powerful and popular tool for classification and prediction. A Decision tree is a flowchart like tree structure, where each internal node denotes a test on an attribute, each branch represents an outcome of the test, and each leaf node (terminal node) holds a class label.

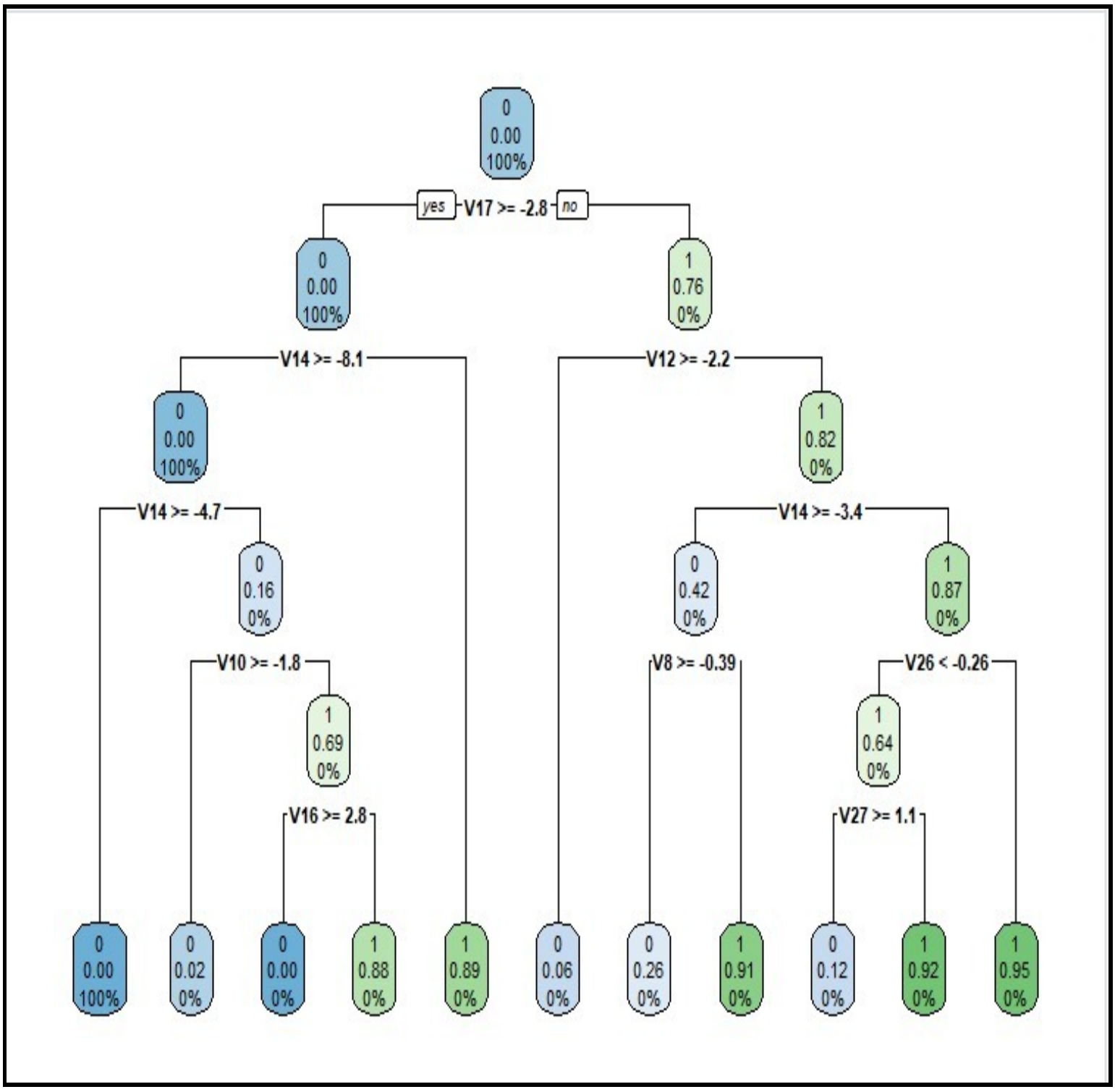
The goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features. A tree can be seen as a piecewise constant approximation.

For instance, in the example below, decision trees learn from data to approximate a sine curve with a set of if-then-else decision rules. The deeper the tree, the more complex the decision rules and the fitter the model.

The feature importance is clear and relations can be viewed easily. This methodology is more commonly known as learning decision tree from data and the above tree is called Classification tree as the target is to classify passengers as survived or died. Regression trees are represented in the same manner, just they predict continuous values like the price of a house. In general, Decision Tree algorithms are referred to as CART or Classification and Regression Trees.

Decision trees are used to solve both classification and regression problems in the form of trees that can be incrementally updated by splitting the dataset into smaller datasets (numerical and categorical), where the results are represented in the leaf nodes.

From the decision tree we can clearly see the efficiency level at every state and we see that for most of them the efficiency of the model is either 100% or 0%.


0
0.00
100%
yes
V17 >= -2.8
no
0
0.00
100%
1
0.76
0%
V14 >= -8.1
V12 >= -2.2
0
0.00
100%
1
0.82
0%
V14 >= -4.7
V14 >= -3.4
0
0.16
0%
0
0.42
0%
1
0.87
0%
V10 >= -1.8
V8 >= -0.39
V26 < -0.26
1
0.69
0%
1
0.64
0%
V16 >= 2.8
V27 >= 1.1
0
0.00
100%
0
0.02
0%
0
0.00
0%
1
0.88
0%
1
0.89
0%
0
0.06
0%
0
0.26
0%
1
0.91
0%
0
0.12
0%
1
0.92
0%
1
0.95
0%

Testing

Here we tested our data and found that the number of fisher scoring iterations are = 12, null deviance and residual deviance are also correct.

So we can say that our model is working correctly and effectively.

```
> summary(Logistic_Model)

Call:
glm(formula = Class ~ ., family = binomial(), data = test_data)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-4.9812  -0.0284  -0.0166  -0.0103   4.0297

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -8.86705    0.34845 -25.447  < 2e-16 ***
V1           0.10890    0.09160   1.189 0.234517
V2          -0.02350    0.11529  -0.204 0.838467
V3           0.02247    0.09916   0.227 0.820719
V4           0.73881    0.15944   4.634 3.59e-06 ***
V5           0.09265    0.13466   0.688 0.491444
V6          -0.02855    0.15510  -0.184 0.853960
V7          -0.15122    0.13488  -1.121 0.262225
V8          -0.14962    0.07383  -2.027 0.042707 *
V9          -0.17260    0.23371  -0.739 0.460191
V10         -0.75396    0.21055  -3.581 0.000343 ***
V11         -0.04688    0.16612  -0.282 0.777781
V12         -0.03301    0.17939  -0.184 0.853991
V13         -0.35109    0.17244  -2.036 0.041747 *
V14         -0.59346    0.13016  -4.559 5.13e-06 ***
V15         -0.13729    0.17922  -0.766 0.443651
V16         -0.36020    0.26083  -1.381 0.167288
V17          0.12062    0.14343   0.841 0.400385
V18         -0.02233    0.26796  -0.083 0.933582
V19          0.02754    0.20450   0.135 0.892873
V20         -0.46862    0.18892  -2.481 0.013118 *
V21          0.45233    0.13790   3.280 0.001038 **
V22          0.68192    0.28715   2.375 0.017557 *
V23         -0.10105    0.13714  -0.737 0.461223
V24          0.05003    0.32682   0.153 0.878332
V25         -0.16713    0.27840  -0.600 0.548299
V26         -0.87998    0.48640  -1.809 0.070423 .
V27         -0.85537    0.26037  -3.285 0.001019 **
V28         -0.38252    0.24096  -1.587 0.112403
Amount       0.25138    0.18051   1.393 0.163749
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1443.4  on 56960  degrees of freedom
Residual deviance:  450.5  on 56931  degrees of freedom
AIC: 510.5

Number of Fisher Scoring iterations: 12
```

Result

The code prints out the number of false positives it detected and compares it with the actual values. This is used to calculate the accuracy score and precision of the algorithms. The fraction of data we used for faster testing is 10% of the entire dataset. The complete dataset is also used at the end and both the results are printed.

These results along with the classification report for each algorithm is given in the output as follows, where class 0 means the transaction was determined to be valid and 1 means it was determined as a fraud transaction. This result matched against the class values to check for false positives.

And from the ROC curve we can clearly see that the efficiency of our model is 90%.

Conclusion

Credit card fraud is without a doubt an act of criminal dishonesty. This article has listed out the most common methods of fraud along with their detection methods and reviewed recent findings in this field. This paper has also explained in detail, how machine learning can be applied to get better results in fraud detection along with the algorithm, pseudocode, explanation its implementation and experimentation results. While the algorithm does reach over 99.6% accuracy, its precision remains only at 28% when a tenth of the data set is taken into consideration. However, when the entire dataset is fed into the algorithm, the precision rises to 33%. This high percentage of accuracy is to be expected due to the huge imbalance between the number of valid and number of genuine transactions.

Since the entire dataset consists of only two days' transaction records, its only a fraction of data that can be made available if this project were to be used on a commercial scale. Being based on machine learning algorithms, the program will only increase its efficiency over time as more data is put into it.

Since the entire dataset consists of only two days' transaction records, its only a fraction of data that can be made available if this project were to be used on a commercial scale. Being based on machine learning algorithms, the program will only increase its efficiency over time as more data is put into it.

References

"Credit Card Fraud Detection Based on Transaction Behaviour -by John Richard D. Kho, Larry A. Vea" published by Proc. of the 2017 IEEE Region 10 Conference (TENCON), Malaysia, November 5-8, 2017

CLIFTON PHUA1, VINCENT LEE1, KATE SMITH1 & ROSS GAYLER2 " A Comprehensive Survey of Data Mining-based Fraud Detection Research" published by School of Business Systems, Faculty of Information Technology, Monash University, Wellington Road, Clayton, Victoria 3800, Australia

"Survey Paper on Credit Card Fraud Detection by Suman" , Research Scholar, GJUS&T Hisar HCE, Sonepat published by International Journal of Advanced Research in Computer Engineering & Technology (IJARCET) Volume 3 Issue 3, March 2014

"Research on Credit Card Fraud Detection Model Based on Distance Sum – by Wen-Fang YU and Na Wang" published by 2009 International Joint Conference on Artificial Intelligence

"Credit Card Fraud Detection through Parenclitic Network AnalysisBy Massimiliano Zanin, Miguel Romance, Regino Criado, and SantiagoMoral" published by Hindawi Complexity Volume 2018, Article ID 5764370, 9 pages

"Credit Card Fraud Detection: A Realistic Modeling and a Novel Learning Strategy" published by IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS, VOL. 29, NO. 8, AUGUST 2018

"Credit Card Fraud Detection-by Ishu Trivedi, Monika, Mrigya, Mridushi" published by International Journal of Advanced Research in Computer and Communication Engineering Vol. 5, Issue 1, January 2016

David J.Wetson,David J.Hand,M Adams,Whitrow and Piotr Jusczak "Plastic Card Fraud Detection using Peer Group Analysis" Springer, Issue 2008.

Credit Card Fraud Detection - Kaggle
(https://www.kaggle.com/mlg-ulb/creditcardfraud)

Data Analysis with R Programming By Google - Coursera
(https://www.coursera.org/learn/data-analysis-r)

R Programming By John Hopkins - Coursera
(https://www.coursera.org/learn/r-programming)