Fraud Detection using Big Data Technologies

Group 7

Tony Vu
Shubham Maheshwari

November 30th, 2024

TABLE OF CONTENTS

Abstract

This report presents an analysis of credit card fraud detection using big data techniques. The primary objective is to gather a suitable dataset and analyze it to build a predictive model capable of accurately identifying fraudulent transactions. The dataset used includes historical credit card transaction records, with relevant features despite being altered due to confidentiality to detect patterns and anomalies associated with fraud. In this analysis, due to the confidential nature of our data, there is no publicly available data sources to collect. Thus, we skipped the data preprocessing stage the data has already been processed. Our main focuses were on building the predictive model and visualizing any significant insights. Our findings indicate that the use of big data analytics significantly enhances the ability to detect credit card fraud in real-time, reducing false positives and financial losses. The report concludes with recommendations for deploying the model in a production environment and suggests future improvements to further increase detection accuracy.

Introduction

Big data has become a powerful tool in addressing complex challenges across various industries, with credit card fraud detection being a prominent example. By leveraging big data, financial institutions can analyze massive datasets to uncover patterns, trends, and anomalies that signal fraudulent activity. This proactive approach allows for real-time detection of suspicious transactions, enabling companies to respond swiftly and prevent financial losses. Unlike conventional fraud detection methods, which may rely on static rules and outdated information, big data analytics offers a dynamic and adaptable solution. It combines machine learning algorithms with advanced data processing to continuously improve its accuracy in identifying fraudulent behaviors, even as tactics evolve. As a result, integrating big data into credit card fraud detection not only enhances security measures but also helps in reducing the number of false positives, thereby improving customer experience. This report explores the application of big data techniques using a sample dataset in building predictive models for credit card fraud detection, demonstrating how data-driven strategies can effectively combat financial crime.

Data Research and Integration

The dataset we've selected for analysis focuses on identifying transaction among many legitimate ones. Fraud detection is an important issue in the financial sector, as fraud can lead to significance financial losses. The dataset consists of real-world transaction made by European cardholders over a two-day period in September 2013.Since the dataset presents a challenge due to the class imbalance which is why the dataset is particularly useful due to the imbalance in class. The rarity of fraud cases in credit card transactions makes this dataset particularly valuable. Its particularly valuable due to its class imbalance. Fraudulent transactions are rare compared to legitimate one, which reflects real-world scenarios and make this dataset especially useful for building effective fraud detection

The dataset must undergo several steps before integrating this dataset like loading the dataset, understanding the features, dealing with Class Imbalance, Scaling the data and splitting the data to evaluate the performance. Once the data has passed through these steps the dataset is ready for machine learning model what makes this dataset to stand outmost than any other dataset is because of the class imbalance. Fraud cases are exceptionally rare, often representing only 0.1 % of all transaction. Although its rare it makes it challenging for the machine learning models. to address the challenges, we use the metric system like precision, recall and F1 score are crucial for evaluating the dataset. Accuracy provides an overall measure of how often the model predicts correctly, but it can be misleading in imbalanced dataset as it will focus on majority classes while ignoring the minority classes. Precision Evaluates on positive predictions while recall focuses on capturing as many actual positives as possible, minimizing the false negative and ensuring most fraud transaction are identified .the f1 score serves as a balance metric that combines both precision and recalls make it useful in imbalanced dataset where both type of errors carry a significant consequences when choosing a metrics it depends on what question we are answering .Here we are looking more into classification and what are we actually looking into , if the dataset can identify spam or fraudulent transaction.

The data set contains only numerical input variables which were created using PCA otherwise known as Principal Component Analysis .This transformation happens because of the data privacy so other features like the original features and more background features are protected and is not used in the database .The features that is not transferred are time and amount .Here we could take the feature Amount were the money was involved in each transaction and it can be used the model more sensitive to the size of transaction .The class feature is a factorial where the values shows 1 and 0 where 1 means fraud and 0 indicates a legitimate transaction . the time shows the time of transactions.

Normalization is an essential preprocessing steps for this dataset. Sometimes features like amount can have widely difference in price which can have varying scales thus leading to a not looking normal data. we are normalizing the data cause some Machine Learning algorithms are sensitive to feature scales.

Splitting the dataset into training and test data set for machine learning models to run and understand which model is better suitable for running this Fraudulent dataset

Data Collection

Data collection is an ongoing process, constantly evolving as the focus of research sharpens. To get started, we've filtered out several key datasets from Kaggle, including those related to fraud detection in areas like IEEE-CIS, E-Commerce, healthcare, promo code abuse, self-checkout systems, and general fraud detection. Each of these datasets captures a different type of fraud, giving us a broad view of the challenges we're up against. By examining this wide range of data, we'll be able to understand various fraudulent activities and the techniques that are commonly used to catch them. This variety helped us paint a clearer picture of the types of fraud happening across different industries. At the end, we decided to go with credit card fraud dataset as it is a major problem that many have been trying to resolve, hence offer the maximum references and resources.

For this dataset, it has already been cleaned and altered for confidentiality, so we did not need to go through the data cleansing and transformation process. However, we understand that for raw datasets, to make sure we're working with the best possible data, we would need to put effort into data cleansing and maintenance. This means addressing any missing information, inconsistencies, or outliers in the datasets so that we're left with reliable, high-quality data. Clean, accurate data is essential for drawing meaningful conclusions, especially when dealing with something as complex as fraud detection. It ensures that our analysis is aligned with our goals and gives us the foundation to build strong fraud detection models.

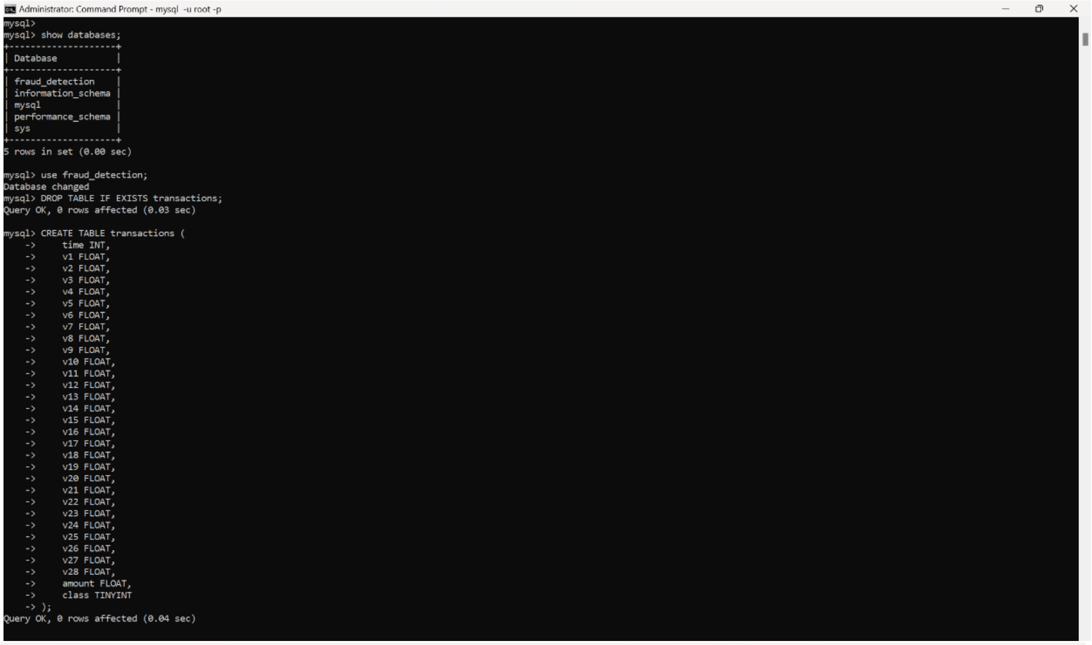
Beyond cleansing, feature engineering and transforming the data are next steps to make it even more useful. Fraud detection is all about finding subtle patterns, and by creating new features—like looking at transaction frequency or spotting time-based trends—we can get a deeper understanding of fraudulent behaviors. Unfortunately, for similar reason of confidentiality mentioned above, we also skipped these steps. However, we deeply understand the importance of data engineering and transforming to help improve the performance of our models. The long-term goal is not just to make accurate models, but to make sure they're flexible and can handle different types of fraud across various industries, making our analysis both reliable and effective.

Data Storage and Maintenance

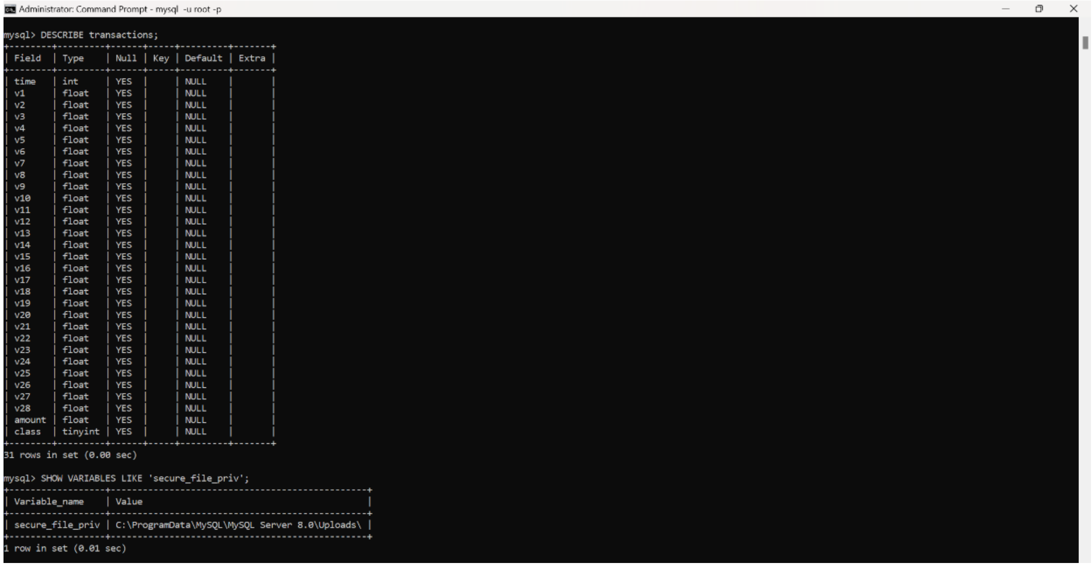
In our fraud detection project, the dataset is stored and managed using **MySQL**, a robust relational database management system. MySQL was chosen for its ability to handle large-scale data efficiently while supporting fast and flexible querying, which is essential for identifying patterns and anomalies in credit card transactions. Here's how we have structured and maintained our data:

1. **Database Design:**

    - A dedicated **fraud_detection** database was created in MySQL to store transaction data. The primary table, transactions, contains key features such as:
        - **time**: Time of the transaction.,
        - amount, and
        - **class**: A binary label indicating fraud (1) or legitimate transaction (0).
        - **v1 to v28**: Numerical features derived using Principal Component Analysis (PCA) for enhanced data privacy

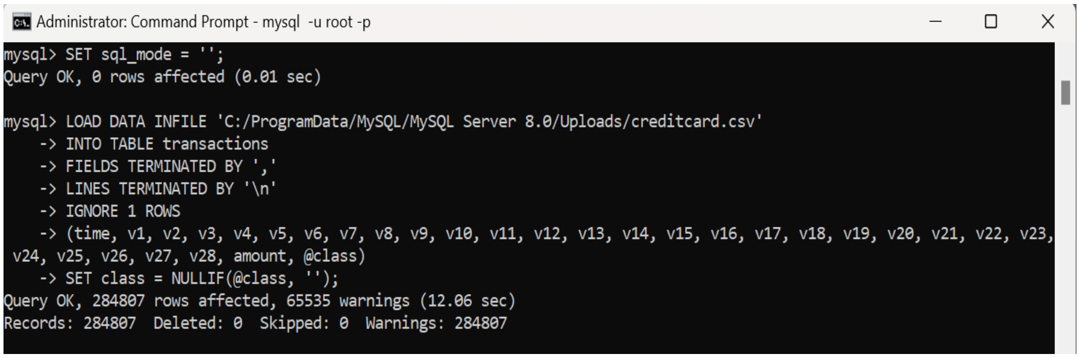


```
mysql>
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| fraud_detection    |
| information_schema |
| mysql              |
| performance_schema |
| sys                |
+--------------------+
5 rows in set (0.00 sec)

mysql> use fraud_detection;
Database changed
mysql> DROP TABLE IF EXISTS transactions;
Query OK, 0 rows affected (0.03 sec)

mysql> CREATE TABLE transactions (
    ->     time INT,
    ->     v1 FLOAT,
    ->     v2 FLOAT,
    ->     v3 FLOAT,
    ->     v4 FLOAT,
    ->     v5 FLOAT,
    ->     v6 FLOAT,
    ->     v7 FLOAT,
    ->     v8 FLOAT,
    ->     v9 FLOAT,
    ->     v10 FLOAT,
    ->     v11 FLOAT,
    ->     v12 FLOAT,
    ->     v13 FLOAT,
    ->     v14 FLOAT,
    ->     v15 FLOAT,
    ->     v16 FLOAT,
    ->     v17 FLOAT,
    ->     v18 FLOAT,
    ->     v19 FLOAT,
    ->     v20 FLOAT,
    ->     v21 FLOAT,
    ->     v22 FLOAT,
    ->     v23 FLOAT,
    ->     v24 FLOAT,
    ->     v25 FLOAT,
    ->     v26 FLOAT,
    ->     v27 FLOAT,
    ->     v28 FLOAT,
    ->     amount FLOAT,
    ->     class TINYINT
    -> );
Query OK, 0 rows affected (0.04 sec)
```



```
mysql> DESCRIBE transactions;
+--------+---------+------+-----+---------+-------+
| Field  | Type    | Null | Key | Default | Extra |
+--------+---------+------+-----+---------+-------+
| time   | int     | YES  |     | NULL    |       |
| v1     | float   | YES  |     | NULL    |       |
| v2     | float   | YES  |     | NULL    |       |
| v3     | float   | YES  |     | NULL    |       |
| v4     | float   | YES  |     | NULL    |       |
| v5     | float   | YES  |     | NULL    |       |
| v6     | float   | YES  |     | NULL    |       |
| v7     | float   | YES  |     | NULL    |       |
| v8     | float   | YES  |     | NULL    |       |
| v9     | float   | YES  |     | NULL    |       |
| v10    | float   | YES  |     | NULL    |       |
| v11    | float   | YES  |     | NULL    |       |
| v12    | float   | YES  |     | NULL    |       |
| v13    | float   | YES  |     | NULL    |       |
| v14    | float   | YES  |     | NULL    |       |
| v15    | float   | YES  |     | NULL    |       |
| v16    | float   | YES  |     | NULL    |       |
| v17    | float   | YES  |     | NULL    |       |
| v18    | float   | YES  |     | NULL    |       |
| v19    | float   | YES  |     | NULL    |       |
| v20    | float   | YES  |     | NULL    |       |
| v21    | float   | YES  |     | NULL    |       |
| v22    | float   | YES  |     | NULL    |       |
| v23    | float   | YES  |     | NULL    |       |
| v24    | float   | YES  |     | NULL    |       |
| v25    | float   | YES  |     | NULL    |       |
| v26    | float   | YES  |     | NULL    |       |
| v27    | float   | YES  |     | NULL    |       |
| v28    | float   | YES  |     | NULL    |       |
| amount | float   | YES  |     | NULL    |       |
| class  | tinyint | YES  |     | NULL    |       |
+--------+---------+------+-----+---------+-------+
31 rows in set (0.00 sec)

mysql> SHOW VARIABLES LIKE 'secure_file_priv';
+------------------+------------------------------------------------+
| Variable_name    | Value                                          |
+------------------+------------------------------------------------+
| secure_file_priv | C:\ProgramData\MySQL\MySQL Server 8.0\Uploads\ |
+------------------+------------------------------------------------+
1 row in set (0.01 sec)
```

2. **Data Loading**:
    - The dataset was loaded into the transactions table using the LOAD DATA INFILE command. This method ensures efficient bulk data insertion while managing over 284,807 rows of transaction data.
    - Special considerations were made to handle potential NULL values in the class column during the data import process.



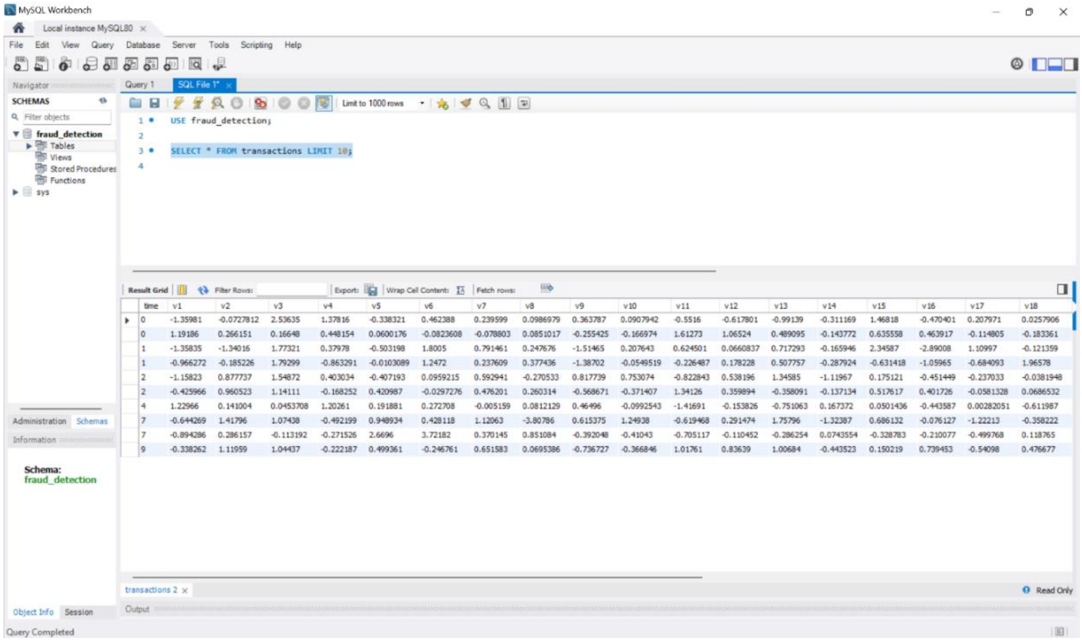
```
mysql> SET sql_mode = '';
Query OK, 0 rows affected (0.01 sec)

mysql> LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/creditcard.csv'
    -> INTO TABLE transactions
    -> FIELDS TERMINATED BY ','
    -> LINES TERMINATED BY '\n'
    -> IGNORE 1 ROWS
    -> (time, v1, v2, v3, v4, v5, v6, v7, v8, v9, v10, v11, v12, v13, v14, v15, v16, v17, v18, v19, v20, v21, v22, v23,
 v24, v25, v26, v27, v28, amount, @class)
    -> SET class = NULLIF(@class, '');
Query OK, 284807 rows affected, 65535 warnings (12.06 sec)
Records: 284807  Deleted: 0  Skipped: 0  Warnings: 284807
```

3. **Data Verification and Backup:**

    a. After loading the data, we verified its accuracy by querying the database and reviewing sample records using MySQL Workbench.

    b. Regular backups are created to prevent data loss. A combination of **local backups** and **Google Drive storage** ensures data safety and accessibility for all team members.

    c. The **secure_file_priv** configuration was utilized to ensure secure import/export operations within MySQL.



```
USE fraud_detection;

SELECT * FROM transactions LIMIT 10;
```

time	v1	v2	v3	v4	v5	v6	v7	v8	v9	v10	v11	v12	v13	v14	v15	v16	v17	v18
0	-1.35981	-0.0727812	2.53635	1.37816	-0.338321	0.462388	0.239599	0.0986979	0.363787	0.0907942	-0.5516	-0.617801	-0.99139	-0.311169	1.46818	-0.470401	0.207971	0.0257906
0	1.19186	0.266151	0.16648	0.448154	0.0600176	-0.0823608	-0.078803	0.0851017	-0.255425	-0.166974	1.61273	1.06524	0.489095	-0.143772	0.635558	0.463917	-0.114805	-0.183361
1	-1.35835	-1.34016	1.77321	0.37978	-0.503198	1.8005	0.791461	0.247676	-1.51465	0.207643	0.624501	0.0660837	0.717293	-0.165946	2.34587	-2.89008	1.10997	-0.121359
1	-0.966272	-0.185226	1.79299	-0.863291	-0.0103089	1.2472	0.237609	0.377436	-1.38702	-0.0549519	-0.226487	0.178228	0.507757	-0.287924	-0.631418	-1.05965	-0.684093	1.96578
2	-1.15823	0.877737	1.54872	0.403034	-0.407193	0.0959215	0.592941	-0.270533	0.817739	0.753074	-0.822843	0.538196	1.34585	-1.11967	0.175121	-0.451449	-0.237033	-0.0381948
2	-0.425966	0.960523	1.14111	-0.168252	0.420987	-0.0297276	0.476201	0.260314	-0.568671	-0.371407	1.34126	0.359894	-0.358091	-0.137134	0.517617	0.401726	-0.0581328	0.0686532
4	1.22966	0.141004	0.0453708	1.20261	0.191881	0.272708	-0.005159	0.0812129	0.46496	-0.0992543	-1.41691	-0.153826	-0.751063	0.167372	0.0501436	-0.443587	0.00282051	-0.611987
7	-0.644269	1.41796	1.07438	-0.492199	0.948934	0.428118	1.12063	-3.80786	0.615375	1.24938	-0.619468	0.291474	1.75796	-1.32387	0.686132	-0.076127	-1.22213	-0.358222
7	-0.894286	0.286157	-0.113192	-0.271526	2.6696	3.72182	0.370145	0.851084	-0.392048	-0.41043	-0.705117	-0.110452	-0.286254	0.0743554	-0.328783	-0.210077	-0.499768	0.118765
9	-0.338262	1.11959	1.04437	-0.222187	0.499361	-0.246761	0.651583	0.0695386	-0.736727	-0.366846	1.01761	0.83639	1.00684	-0.443523	0.150219	0.739453	-0.54098	0.476677

4. **Collaboration and Maintenance:**

    a. **Google Drive** facilitates real-time data sharing and updates among team members. This setup allows synchronized work on data analysis and visualization tasks.

    b. To ensure version control and reproducibility, we plan to integrate **Data Version Control (DVC)** in future iterations. This will help track changes to the dataset and models, ensuring all members use the most up-to-date data.

5. **Future Considerations:**

    a. As the project scales, we aim to implement distributed storage solutions like **Amazon RDS** or **Google Cloud SQL** for enhanced performance and scalability.

    b. Additional measures, such as automated backup schedules and enhanced database indexing, will further improve data accessibility and reliability.

Data Quality

As data has already been processed, there is limitations in our capacity to confirm the quality of the data. However, with highly reviewed and large number of comments on Kaggle for the dataset, we trust the owner of the dataset has gone through intensive considerations and processing to ensure data is ready for training and visualizations.

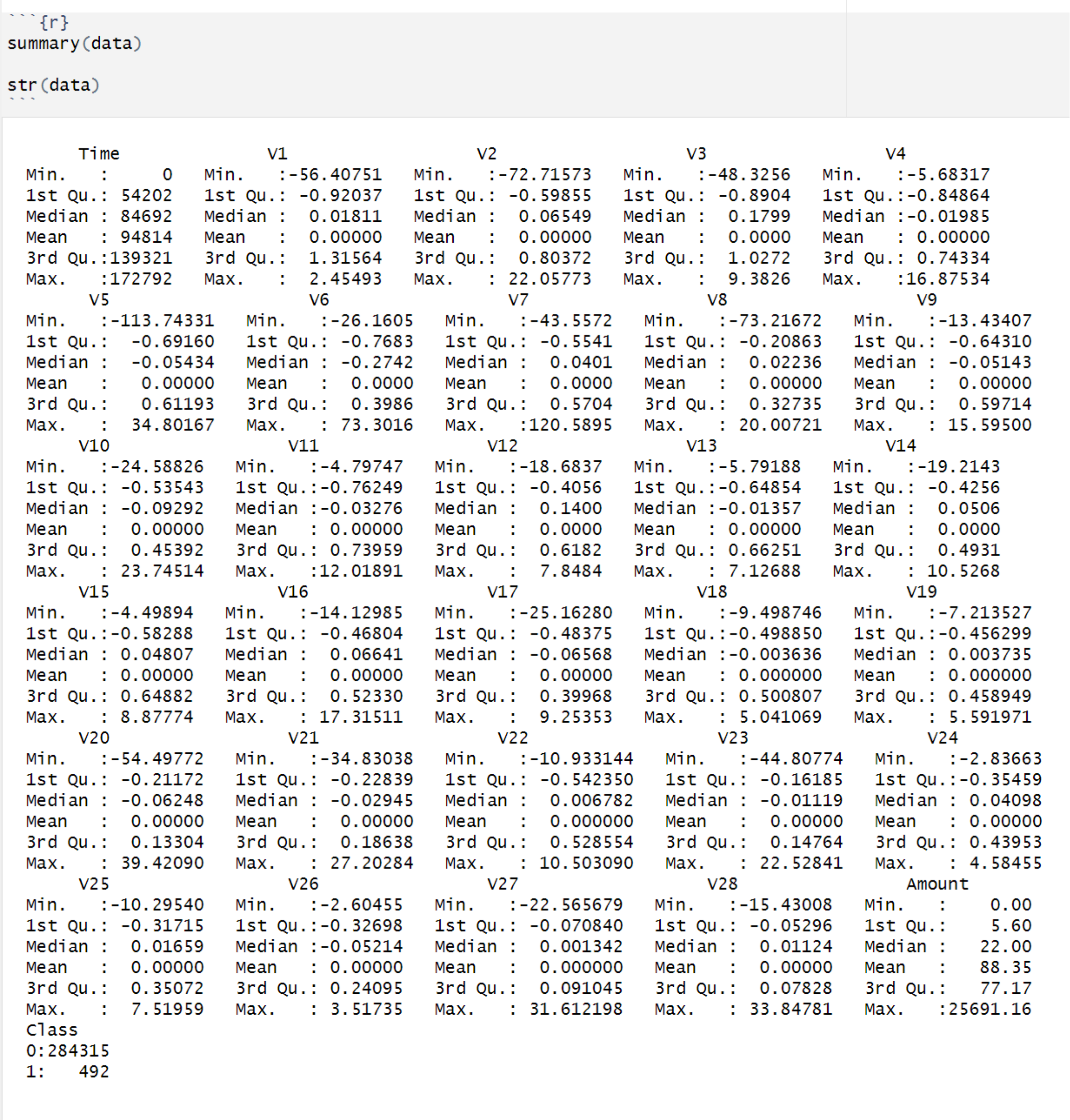
Data Analysis and Visualization

We will use R with visualization libraries to analyze and visualize the data. We will also use Microsoft Excel to create reports containing insightful dashboards to drive informed decision making from potential stakeholders.

Analysis

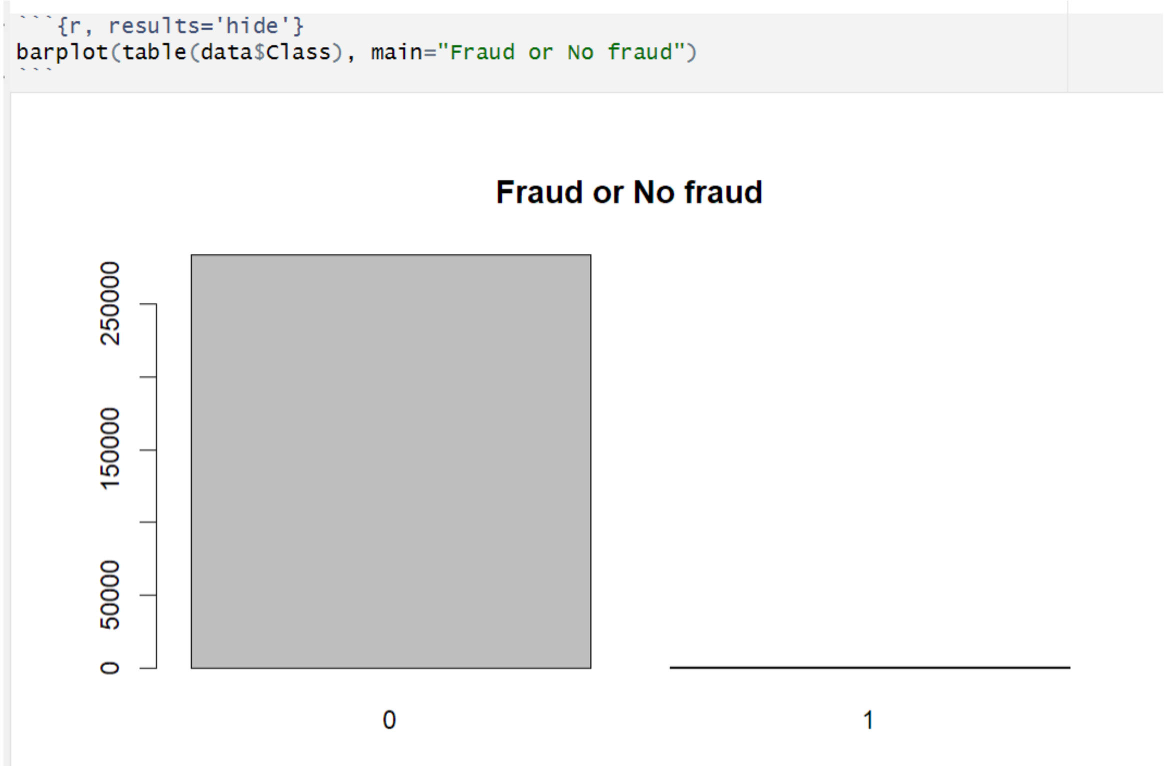
1. Data exploratory:

As we already know from our data research that the dataset is imbalanced and normalized. Out of 31 variables, there are 28 variables have been normalized for easier training. There are 284315 records of non-fraud transactions and 492 fraud transactions, representing a disproportionate distribution within the data set (99.83% VS 0.17%).



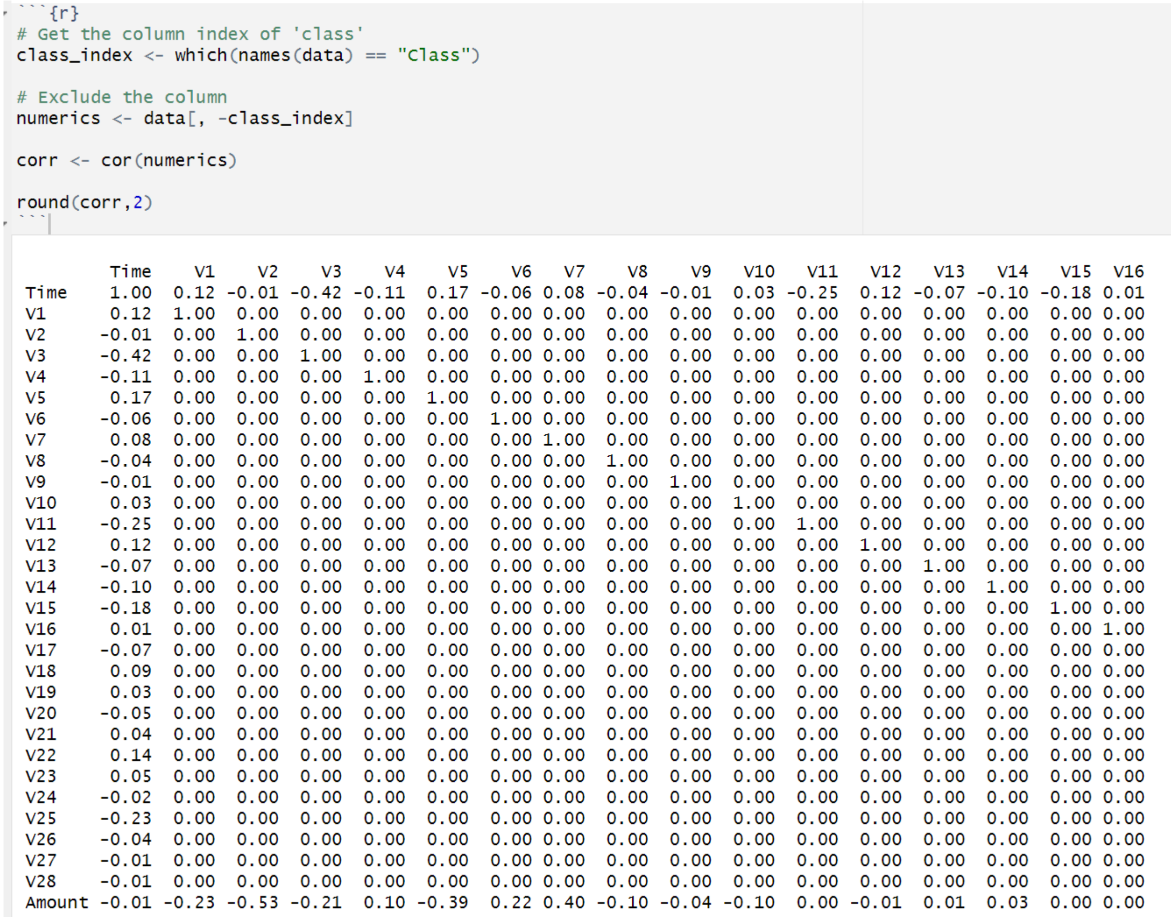
~~~
```{r}
summary(data)

str(data)
```
~~~

```
      Time              V1                  V2                  V3                 V4          
 Min.   :     0   Min.   :-56.40751   Min.   :-72.71573   Min.   :-48.3256   Min.   :-5.68317  
 1st Qu.: 54202   1st Qu.: -0.92037   1st Qu.: -0.59855   1st Qu.: -0.8904   1st Qu.:-0.84864  
 Median : 84692   Median :  0.01811   Median :  0.06549   Median :  0.1799   Median :-0.01985  
 Mean   : 94814   Mean   :  0.00000   Mean   :  0.00000   Mean   :  0.0000   Mean   : 0.00000  
 3rd Qu.:139321   3rd Qu.:  1.31564   3rd Qu.:  0.80372   3rd Qu.:  1.0272   3rd Qu.: 0.74334  
 Max.   :172792   Max.   :  2.45493   Max.   : 22.05773   Max.   :  9.3826   Max.   :16.87534  
       V5                   V6                  V7                  V8                  V9          
 Min.   :-113.74331   Min.   :-26.1605   Min.   :-43.5572   Min.   :-73.21672   Min.   :-13.43407  
 1st Qu.:  -0.69160   1st Qu.: -0.7683   1st Qu.: -0.5541   1st Qu.: -0.20863   1st Qu.: -0.64310  
 Median :  -0.05434   Median : -0.2742   Median :  0.0401   Median :  0.02236   Median : -0.05143  
 Mean   :   0.00000   Mean   :  0.0000   Mean   :  0.0000   Mean   :  0.00000   Mean   :  0.00000  
 3rd Qu.:   0.61193   3rd Qu.:  0.3986   3rd Qu.:  0.5704   3rd Qu.:  0.32735   3rd Qu.:  0.59714  
 Max.   :  34.80167   Max.   : 73.3016   Max.   :120.5895   Max.   : 20.00721   Max.   : 15.59500  
      V10                 V11                V12                V13                V14          
 Min.   :-24.58826   Min.   :-4.79747   Min.   :-18.6837   Min.   :-5.79188   Min.   :-19.2143  
 1st Qu.: -0.53543   1st Qu.:-0.76249   1st Qu.: -0.4056   1st Qu.:-0.64854   1st Qu.: -0.4256  
 Median : -0.09292   Median :-0.03276   Median :  0.1400   Median :-0.01357   Median :  0.0506  
 Mean   :  0.00000   Mean   : 0.00000   Mean   :  0.0000   Mean   : 0.00000   Mean   :  0.0000  
 3rd Qu.:  0.45392   3rd Qu.: 0.73959   3rd Qu.:  0.6182   3rd Qu.: 0.66251   3rd Qu.:  0.4931  
 Max.   : 23.74514   Max.   :12.01891   Max.   :  7.8484   Max.   : 7.12688   Max.   : 10.5268  
      V15                V16                 V17                 V18                 V19           
 Min.   :-4.49894   Min.   :-14.12985   Min.   :-25.16280   Min.   :-9.498746   Min.   :-7.213527  
 1st Qu.:-0.58288   1st Qu.: -0.46804   1st Qu.: -0.48375   1st Qu.:-0.498850   1st Qu.:-0.456299  
 Median : 0.04807   Median :  0.06641   Median : -0.06568   Median :-0.003636   Median : 0.003735  
 Mean   : 0.00000   Mean   :  0.00000   Mean   :  0.00000   Mean   : 0.000000   Mean   : 0.000000  
 3rd Qu.: 0.64882   3rd Qu.:  0.52330   3rd Qu.:  0.39968   3rd Qu.: 0.500807   3rd Qu.: 0.458949  
 Max.   : 8.87774   Max.   : 17.31511   Max.   :  9.25353   Max.   : 5.041069   Max.   : 5.591971  
      V20                 V21                 V22                  V23                 V24          
 Min.   :-54.49772   Min.   :-34.83038   Min.   :-10.933144   Min.   :-44.80774   Min.   :-2.83663  
 1st Qu.: -0.21172   1st Qu.: -0.22839   1st Qu.: -0.542350   1st Qu.: -0.16185   1st Qu.:-0.35459  
 Median : -0.06248   Median : -0.02945   Median :  0.006782   Median : -0.01119   Median : 0.04098  
 Mean   :  0.00000   Mean   :  0.00000   Mean   :  0.000000   Mean   :  0.00000   Mean   : 0.00000  
 3rd Qu.:  0.13304   3rd Qu.:  0.18638   3rd Qu.:  0.528554   3rd Qu.:  0.14764   3rd Qu.: 0.43953  
 Max.   : 39.42090   Max.   : 27.20284   Max.   : 10.503090   Max.   : 22.52841   Max.   : 4.58455  
      V25                 V26                V27                  V28                Amount        
 Min.   :-10.29540   Min.   :-2.60455   Min.   :-22.565679   Min.   :-15.43008   Min.   :    0.00  
 1st Qu.: -0.31715   1st Qu.:-0.32698   1st Qu.: -0.070840   1st Qu.: -0.05296   1st Qu.:    5.60  
 Median :  0.01659   Median :-0.05214   Median :  0.001342   Median :  0.01124   Median :   22.00  
 Mean   :  0.00000   Mean   : 0.00000   Mean   :  0.000000   Mean   :  0.00000   Mean   :   88.35  
 3rd Qu.:  0.35072   3rd Qu.: 0.24095   3rd Qu.:  0.091045   3rd Qu.:  0.07828   3rd Qu.:   77.17  
 Max.   :  7.51959   Max.   : 3.51735   Max.   : 31.612198   Max.   : 33.84781   Max.   :25691.16  
 Class     
 0:284315  
 1:   492  
```



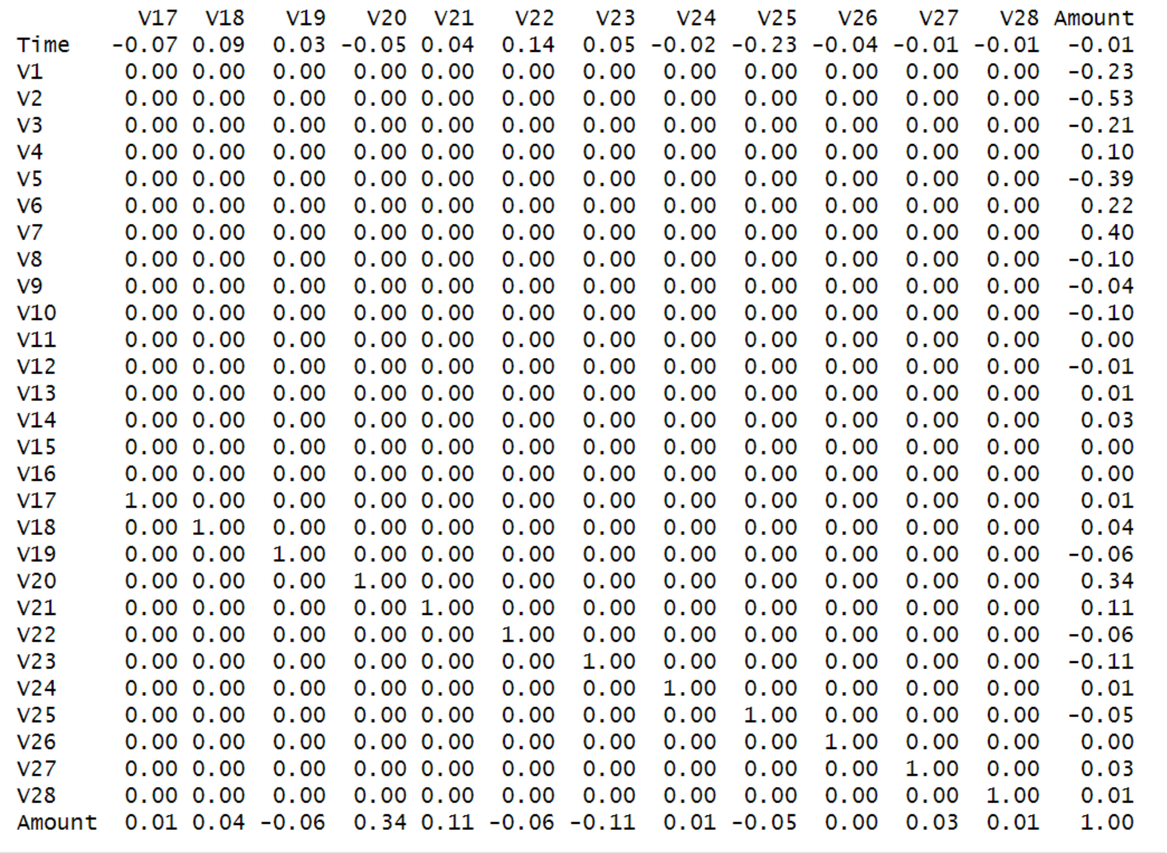
~~~
```{r, results='hide'}
barplot(table(data$Class), main="Fraud or No fraud")
```
~~~

Because data was normalized, we don’t have the context or know what each data represents. However, we can still go through the data and try understand the relationships between variables. Let’s look at the correlations between variables to see if we can get some useful information.



~~~
```{r}
# Get the column index of 'class'
class_index <- which(names(data) == "Class")

# Exclude the column
numerics <- data[, -class_index]

corr <- cor(numerics)

round(corr,2)
```
~~~

```
        Time    V1    V2    V3    V4    V5    V6   V7    V8    V9   V10   V11   V12   V13   V14   V15  V16
Time    1.00  0.12 -0.01 -0.42 -0.11  0.17 -0.06 0.08 -0.04 -0.01  0.03 -0.25  0.12 -0.07 -0.10 -0.18 0.01
V1      0.12  1.00  0.00  0.00  0.00  0.00  0.00 0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00 0.00
V2     -0.01  0.00  1.00  0.00  0.00  0.00  0.00 0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00 0.00
V3     -0.42  0.00  0.00  1.00  0.00  0.00  0.00 0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00 0.00
V4     -0.11  0.00  0.00  0.00  1.00  0.00  0.00 0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00 0.00
V5      0.17  0.00  0.00  0.00  0.00  1.00  0.00 0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00 0.00
V6     -0.06  0.00  0.00  0.00  0.00  0.00  1.00 0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00 0.00
V7      0.08  0.00  0.00  0.00  0.00  0.00  0.00 1.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00 0.00
V8     -0.04  0.00  0.00  0.00  0.00  0.00  0.00 0.00  1.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00 0.00
V9     -0.01  0.00  0.00  0.00  0.00  0.00  0.00 0.00  0.00  1.00  0.00  0.00  0.00  0.00  0.00  0.00 0.00
V10     0.03  0.00  0.00  0.00  0.00  0.00  0.00 0.00  0.00  0.00  1.00  0.00  0.00  0.00  0.00  0.00 0.00
V11    -0.25  0.00  0.00  0.00  0.00  0.00  0.00 0.00  0.00  0.00  0.00  1.00  0.00  0.00  0.00  0.00 0.00
V12     0.12  0.00  0.00  0.00  0.00  0.00  0.00 0.00  0.00  0.00  0.00  0.00  1.00  0.00  0.00  0.00 0.00
V13    -0.07  0.00  0.00  0.00  0.00  0.00  0.00 0.00  0.00  0.00  0.00  0.00  0.00  1.00  0.00  0.00 0.00
V14    -0.10  0.00  0.00  0.00  0.00  0.00  0.00 0.00  0.00  0.00  0.00  0.00  0.00  0.00  1.00  0.00 0.00
V15    -0.18  0.00  0.00  0.00  0.00  0.00  0.00 0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  1.00 0.00
V16     0.01  0.00  0.00  0.00  0.00  0.00  0.00 0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00 1.00
V17    -0.07  0.00  0.00  0.00  0.00  0.00  0.00 0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00 0.00
V18     0.09  0.00  0.00  0.00  0.00  0.00  0.00 0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00 0.00
V19     0.03  0.00  0.00  0.00  0.00  0.00  0.00 0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00 0.00
V20    -0.05  0.00  0.00  0.00  0.00  0.00  0.00 0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00 0.00
V21     0.04  0.00  0.00  0.00  0.00  0.00  0.00 0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00 0.00
V22     0.14  0.00  0.00  0.00  0.00  0.00  0.00 0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00 0.00
V23     0.05  0.00  0.00  0.00  0.00  0.00  0.00 0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00 0.00
V24    -0.02  0.00  0.00  0.00  0.00  0.00  0.00 0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00 0.00
V25    -0.23  0.00  0.00  0.00  0.00  0.00  0.00 0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00 0.00
V26    -0.04  0.00  0.00  0.00  0.00  0.00  0.00 0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00 0.00
V27    -0.01  0.00  0.00  0.00  0.00  0.00  0.00 0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00 0.00
V28    -0.01  0.00  0.00  0.00  0.00  0.00  0.00 0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00 0.00
Amount -0.01 -0.23 -0.53 -0.21  0.10 -0.39  0.22 0.40 -0.10 -0.04 -0.10  0.00 -0.01  0.01  0.03  0.00 0.00
```

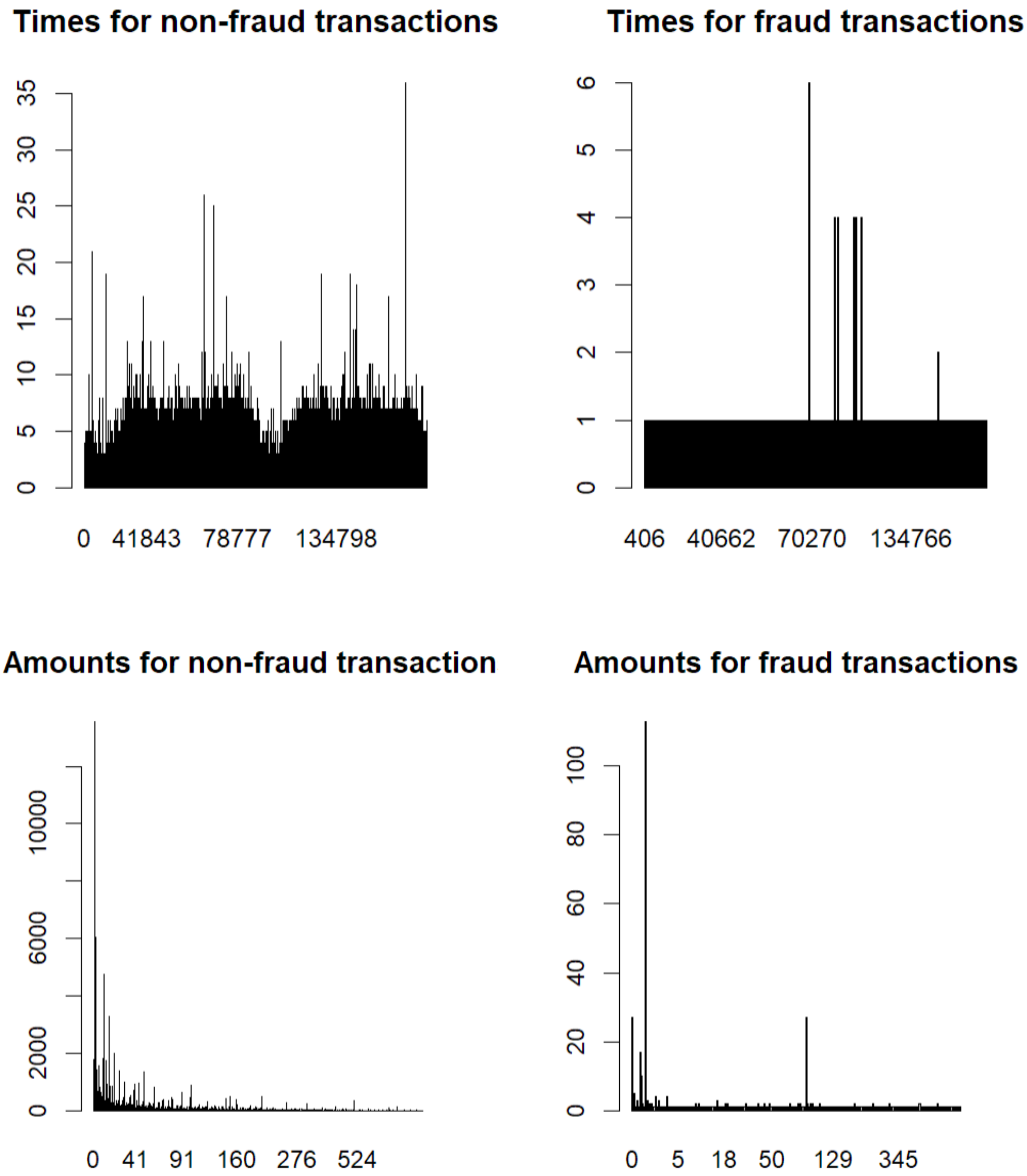


```
          V17  V18   V19   V20  V21   V22   V23   V24   V25   V26   V27   V28 Amount
Time    -0.07 0.09  0.03 -0.05 0.04  0.14  0.05 -0.02 -0.23 -0.04 -0.01 -0.01  -0.01
V1       0.00 0.00  0.00  0.00 0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  -0.23
V2       0.00 0.00  0.00  0.00 0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  -0.53
V3       0.00 0.00  0.00  0.00 0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  -0.21
V4       0.00 0.00  0.00  0.00 0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00   0.10
V5       0.00 0.00  0.00  0.00 0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  -0.39
V6       0.00 0.00  0.00  0.00 0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00   0.22
V7       0.00 0.00  0.00  0.00 0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00   0.40
V8       0.00 0.00  0.00  0.00 0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  -0.10
V9       0.00 0.00  0.00  0.00 0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  -0.04
V10      0.00 0.00  0.00  0.00 0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  -0.10
V11      0.00 0.00  0.00  0.00 0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00   0.00
V12      0.00 0.00  0.00  0.00 0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  -0.01
V13      0.00 0.00  0.00  0.00 0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00   0.01
V14      0.00 0.00  0.00  0.00 0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00   0.03
V15      0.00 0.00  0.00  0.00 0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00   0.00
V16      0.00 0.00  0.00  0.00 0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00   0.00
V17      1.00 0.00  0.00  0.00 0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00   0.01
V18      0.00 1.00  0.00  0.00 0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00   0.04
V19      0.00 0.00  1.00  0.00 0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00  -0.06
V20      0.00 0.00  0.00  1.00 0.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00   0.34
V21      0.00 0.00  0.00  0.00 1.00  0.00  0.00  0.00  0.00  0.00  0.00  0.00   0.11
V22      0.00 0.00  0.00  0.00 0.00  1.00  0.00  0.00  0.00  0.00  0.00  0.00  -0.06
V23      0.00 0.00  0.00  0.00 0.00  0.00  1.00  0.00  0.00  0.00  0.00  0.00  -0.11
V24      0.00 0.00  0.00  0.00 0.00  0.00  0.00  1.00  0.00  0.00  0.00  0.00   0.01
V25      0.00 0.00  0.00  0.00 0.00  0.00  0.00  0.00  1.00  0.00  0.00  0.00  -0.05
V26      0.00 0.00  0.00  0.00 0.00  0.00  0.00  0.00  0.00  1.00  0.00  0.00   0.00
V27      0.00 0.00  0.00  0.00 0.00  0.00  0.00  0.00  0.00  0.00  1.00  0.00   0.03
V28      0.00 0.00  0.00  0.00 0.00  0.00  0.00  0.00  0.00  0.00  0.00  1.00   0.01
Amount   0.01 0.04 -0.06  0.34 0.11 -0.06 -0.11  0.01 -0.05  0.00  0.03  0.01   1.00
```

We noticed that there are no relationships between the Vs variables (V1-V28), but there are some noticeable correlations between some Vs variables and Time or Amount. Some considerate ones are below:

- Moderate negative relation between V3 and Time
- Moderate negative relation between V2 and Amount
- Moderate negative relation between V5 and Amount
- Moderate positive relation between V7 and Amount
- Weak positive relation between V20 and Amount

Now we want to see if there is any difference in the distributions of Times and Amounts between non-fraud and fraud transactions.

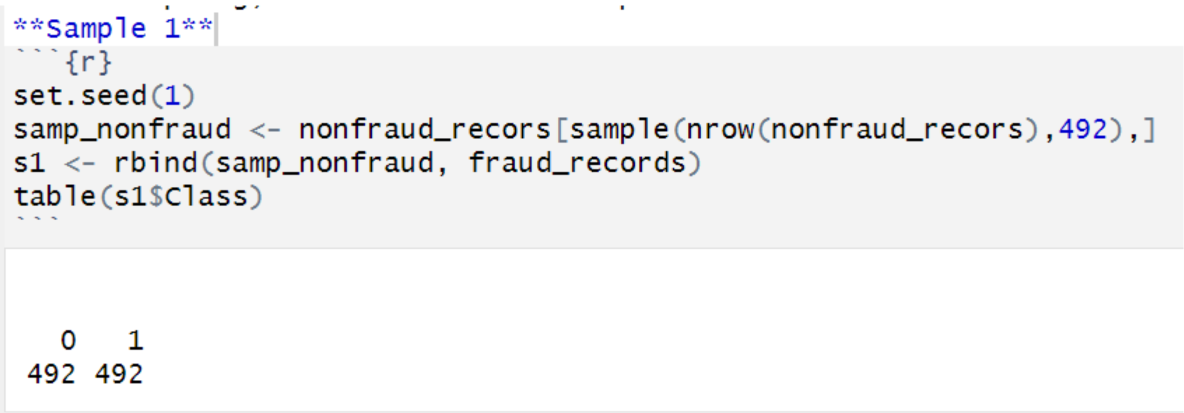


We noticed that for certain period, there tends to have more fraud transactions than other periods. Hence, Time should be a significant factor in our model. Also, we notice majority of fraud transactions has amount of less than $5. This should be a critical information to keep in mind when developing our model.

2. Data modeling:

   As data is imbalanced and biased toward non-fraud transactions. We believe that we should sample the dataset and build our model based on the sample dataset. We also understand that if we only take 1 sample set, we will miss out on a lot of information that might be critical to the confidence level of our model. Our solution to this problem is to sample 10 different sample set with different seed value, so none of them overlapping each other. Then we will decide which sample set best represent and predict the original dataset implementing logistics regression algorithm.

   First, we will do the sampling. It will be 10 different one with different seed value ranging from 1 to 10. Below is an example code chunk of how we sample our dataset.

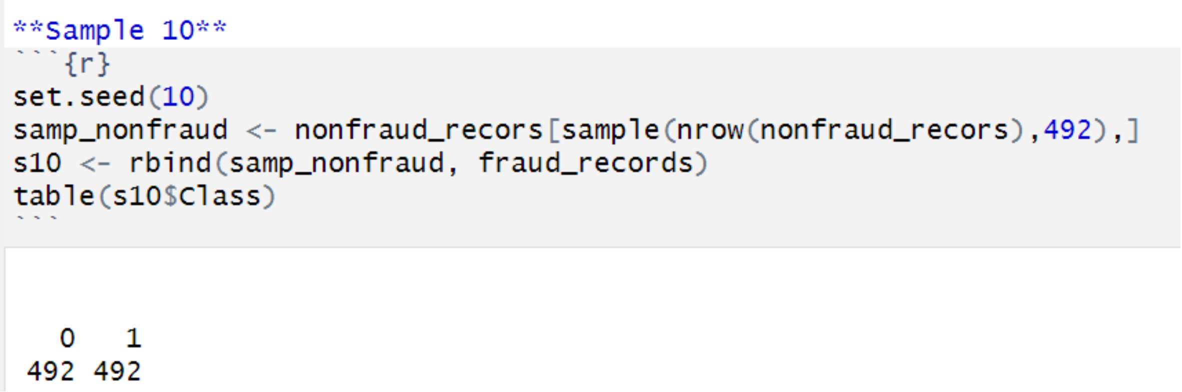


~~~
**Sample 1**
```{r}
set.seed(1)
samp_nonfraud <- nonfraud_recors[sample(nrow(nonfraud_recors),492),]
s1 <- rbind(samp_nonfraud, fraud_records)
table(s1$Class)
```
~~~

```

   0   1
 492 492
```

…

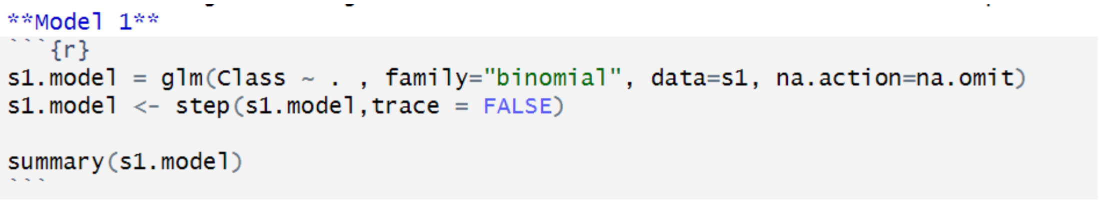
…



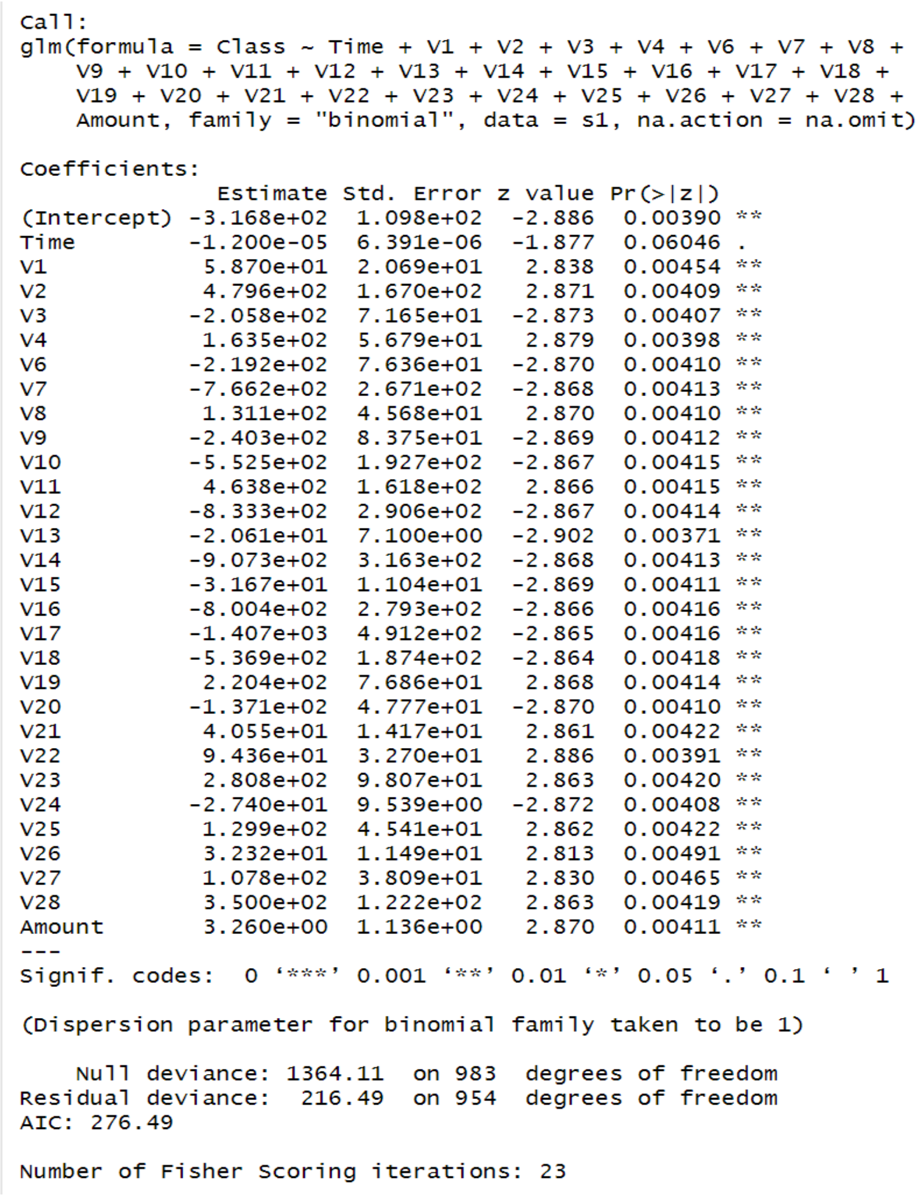
~~~
**Sample 10**
```{r}
set.seed(10)
samp_nonfraud <- nonfraud_recors[sample(nrow(nonfraud_recors),492),]
s10 <- rbind(samp_nonfraud, fraud_records)
table(s10$Class)
```
~~~

```

   0   1
 492 492
```

After running the sampling codes, we have 10 samples (s1 to s10). Next step is to use logistics regression to build the model for each dataset and then calculate metrics like accuracy rate, precision rate, recall rate and f1-score for later evaluations. Below is the sample code chunk of s1 sample set and its model using logistics regression algorithm.

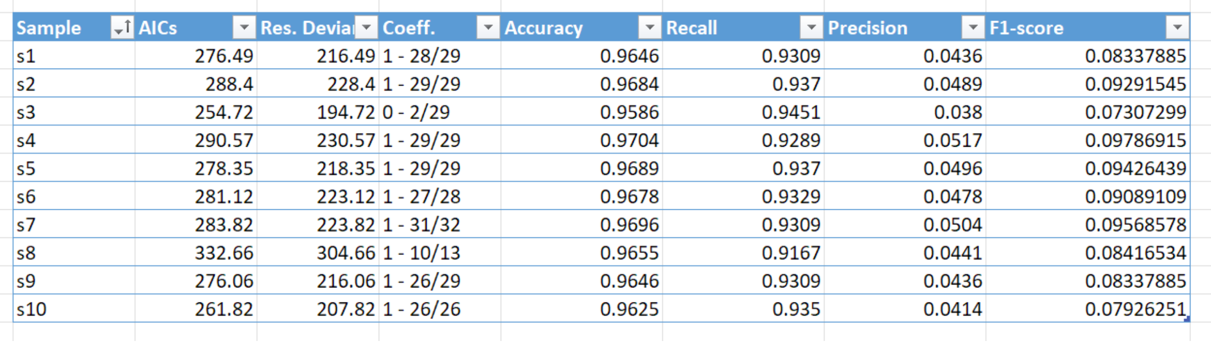


~~~
**Model 1**
```{r}
s1.model = glm(Class ~ . , family="binomial", data=s1, na.action=na.omit)
s1.model <- step(s1.model,trace = FALSE)

summary(s1.model)
```
~~~



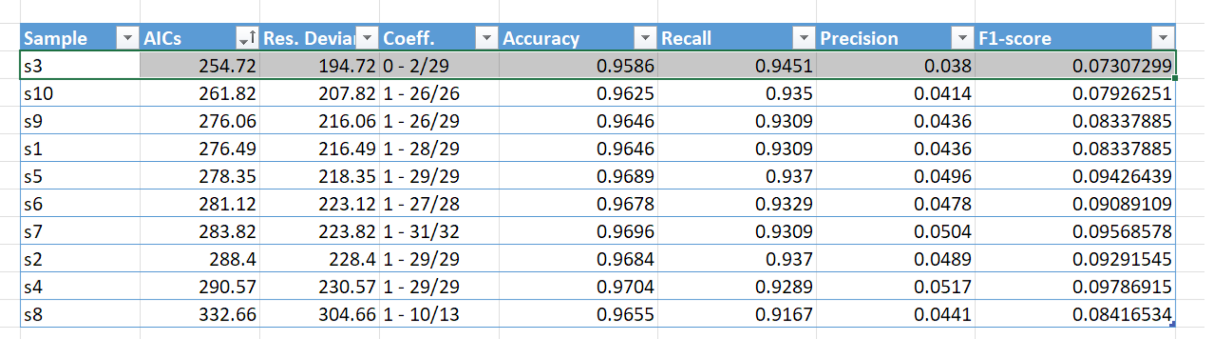
```
Call:
glm(formula = Class ~ Time + V1 + V2 + V3 + V4 + V6 + V7 + V8 +
    V9 + V10 + V11 + V12 + V13 + V14 + V15 + V16 + V17 + V18 +
    V19 + V20 + V21 + V22 + V23 + V24 + V25 + V26 + V27 + V28 +
    Amount, family = "binomial", data = s1, na.action = na.omit)

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept) -3.168e+02  1.098e+02  -2.886  0.00390 **
Time        -1.200e-05  6.391e-06  -1.877  0.06046 .
V1           5.870e+01  2.069e+01   2.838  0.00454 **
V2           4.796e+02  1.670e+02   2.871  0.00409 **
V3          -2.058e+02  7.165e+01  -2.873  0.00407 **
V4           1.635e+02  5.679e+01   2.879  0.00398 **
V6          -2.192e+02  7.636e+01  -2.870  0.00410 **
V7          -7.662e+02  2.671e+02  -2.868  0.00413 **
V8           1.311e+02  4.568e+01   2.870  0.00410 **
V9          -2.403e+02  8.375e+01  -2.869  0.00412 **
V10         -5.525e+02  1.927e+02  -2.867  0.00415 **
V11          4.638e+02  1.618e+02   2.866  0.00415 **
V12         -8.333e+02  2.906e+02  -2.867  0.00414 **
V13         -2.061e+01  7.100e+00  -2.902  0.00371 **
V14         -9.073e+02  3.163e+02  -2.868  0.00413 **
V15         -3.167e+01  1.104e+01  -2.869  0.00411 **
V16         -8.004e+02  2.793e+02  -2.866  0.00416 **
V17         -1.407e+03  4.912e+02  -2.865  0.00416 **
V18         -5.369e+02  1.874e+02  -2.864  0.00418 **
V19          2.204e+02  7.686e+01   2.868  0.00414 **
V20         -1.371e+02  4.777e+01  -2.870  0.00410 **
V21          4.055e+01  1.417e+01   2.861  0.00422 **
V22          9.436e+01  3.270e+01   2.886  0.00391 **
V23          2.808e+02  9.807e+01   2.863  0.00420 **
V24         -2.740e+01  9.539e+00  -2.872  0.00408 **
V25          1.299e+02  4.541e+01   2.862  0.00422 **
V26          3.232e+01  1.149e+01   2.813  0.00491 **
V27          1.078e+02  3.809e+01   2.830  0.00465 **
V28          3.500e+02  1.222e+02   2.863  0.00419 **
Amount       3.260e+00  1.136e+00   2.870  0.00411 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1364.11  on 983  degrees of freedom
Residual deviance:  216.49  on 954  degrees of freedom
AIC: 276.49

Number of Fisher Scoring iterations: 23
```

For each of the 10 models we create, we will look at important factors like whether the model converged and if all of its coefficients pass the z-test, so that they are all different from 0. We made a summary table of all 10 models in excel as below. Note that the table also includes the earlier mentioned metrics (accuracy, precision, recall and f1-score).



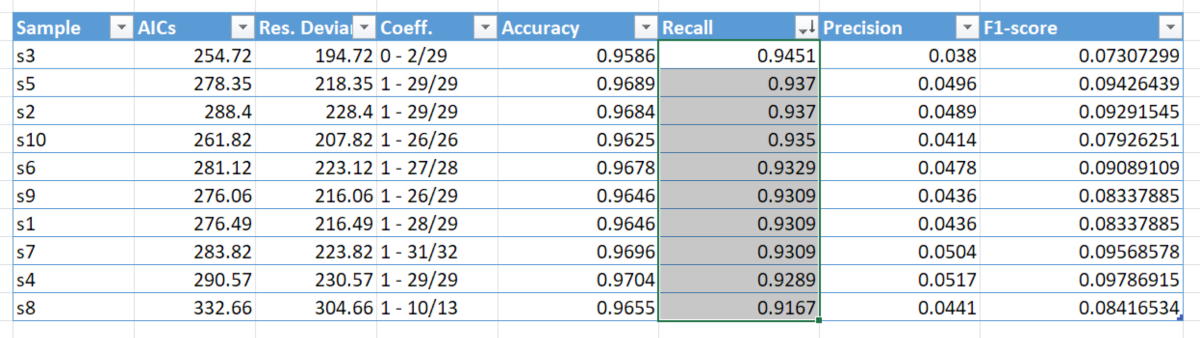
Sample	AICs	Res. Devia	Coeff.	Accuracy	Recall	Precision	F1-score
s1	276.49	216.49	1 - 28/29	0.9646	0.9309	0.0436	0.08337885
s2	288.4	228.4	1 - 29/29	0.9684	0.937	0.0489	0.09291545
s3	254.72	194.72	0 - 2/29	0.9586	0.9451	0.038	0.07307299
s4	290.57	230.57	1 - 29/29	0.9704	0.9289	0.0517	0.09786915
s5	278.35	218.35	1 - 29/29	0.9689	0.937	0.0496	0.09426439
s6	281.12	223.12	1 - 27/28	0.9678	0.9329	0.0478	0.09089109
s7	283.82	223.82	1 - 31/32	0.9696	0.9309	0.0504	0.09568578
s8	332.66	304.66	1 - 10/13	0.9655	0.9167	0.0441	0.08416534
s9	276.06	216.06	1 - 26/29	0.9646	0.9309	0.0436	0.08337885
s10	261.82	207.82	1 - 26/26	0.9625	0.935	0.0414	0.07926251

First thing we want to see which sample has the lowest AIC which measure the level of error of the model. The sample that has the lowest AIC is s3, but it is not a good candidate for selection because only 2 coefficients passed the z-test. And despite having the highest recall rate, which is what we want to focus for fraud detection, its f1-score is the lowest. In short, AIC might not be the best determinant to help with sample selection. We want to take the opportunity to discuss about recall rate and f1-score as the main determinant for fraud detection problem. For problem like fraud detection, we would rather have recall rate, which measures how many actual fraud cases are correctly identified as fraudulent, as high as possible. We don’t might to have a decent number of precision rate as it could be acceptable if the model identifies a non-fraud transaction as fraudulent and then has us humans to make the final call. So, f1-score could be a bit higher.

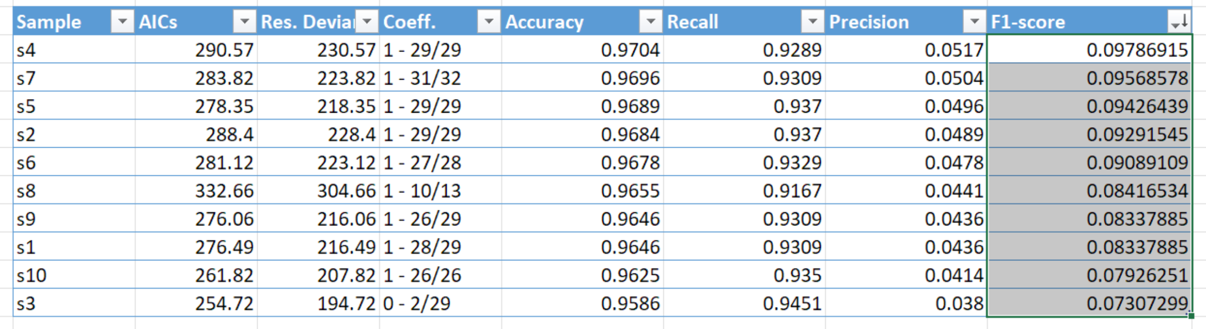


Sample	AICs	Res. Devia	Coeff.	Accuracy	Recall	Precision	F1-score
s3	254.72	194.72	0 - 2/29	0.9586	0.9451	0.038	0.07307299
s10	261.82	207.82	1 - 26/26	0.9625	0.935	0.0414	0.07926251
s9	276.06	216.06	1 - 26/29	0.9646	0.9309	0.0436	0.08337885
s1	276.49	216.49	1 - 28/29	0.9646	0.9309	0.0436	0.08337885
s5	278.35	218.35	1 - 29/29	0.9689	0.937	0.0496	0.09426439
s6	281.12	223.12	1 - 27/28	0.9678	0.9329	0.0478	0.09089109
s7	283.82	223.82	1 - 31/32	0.9696	0.9309	0.0504	0.09568578
s2	288.4	228.4	1 - 29/29	0.9684	0.937	0.0489	0.09291545
s4	290.57	230.57	1 - 29/29	0.9704	0.9289	0.0517	0.09786915
s8	332.66	304.66	1 - 10/13	0.9655	0.9167	0.0441	0.08416534

We sort the table based on recall rate and then based on f1-score. We have a few decent candidate sample sets to select which are s5, s2, and s7.



Sample	AICs	Res. Devia	Coeff.	Accuracy	Recall	Precision	F1-score
s3	254.72	194.72	0 - 2/29	0.9586	0.9451	0.038	0.07307299
s5	278.35	218.35	1 - 29/29	0.9689	0.937	0.0496	0.09426439
s2	288.4	228.4	1 - 29/29	0.9684	0.937	0.0489	0.09291545
s10	261.82	207.82	1 - 26/26	0.9625	0.935	0.0414	0.07926251
s6	281.12	223.12	1 - 27/28	0.9678	0.9329	0.0478	0.09089109
s9	276.06	216.06	1 - 26/29	0.9646	0.9309	0.0436	0.08337885
s1	276.49	216.49	1 - 28/29	0.9646	0.9309	0.0436	0.08337885
s7	283.82	223.82	1 - 31/32	0.9696	0.9309	0.0504	0.09568578
s4	290.57	230.57	1 - 29/29	0.9704	0.9289	0.0517	0.09786915
s8	332.66	304.66	1 - 10/13	0.9655	0.9167	0.0441	0.08416534



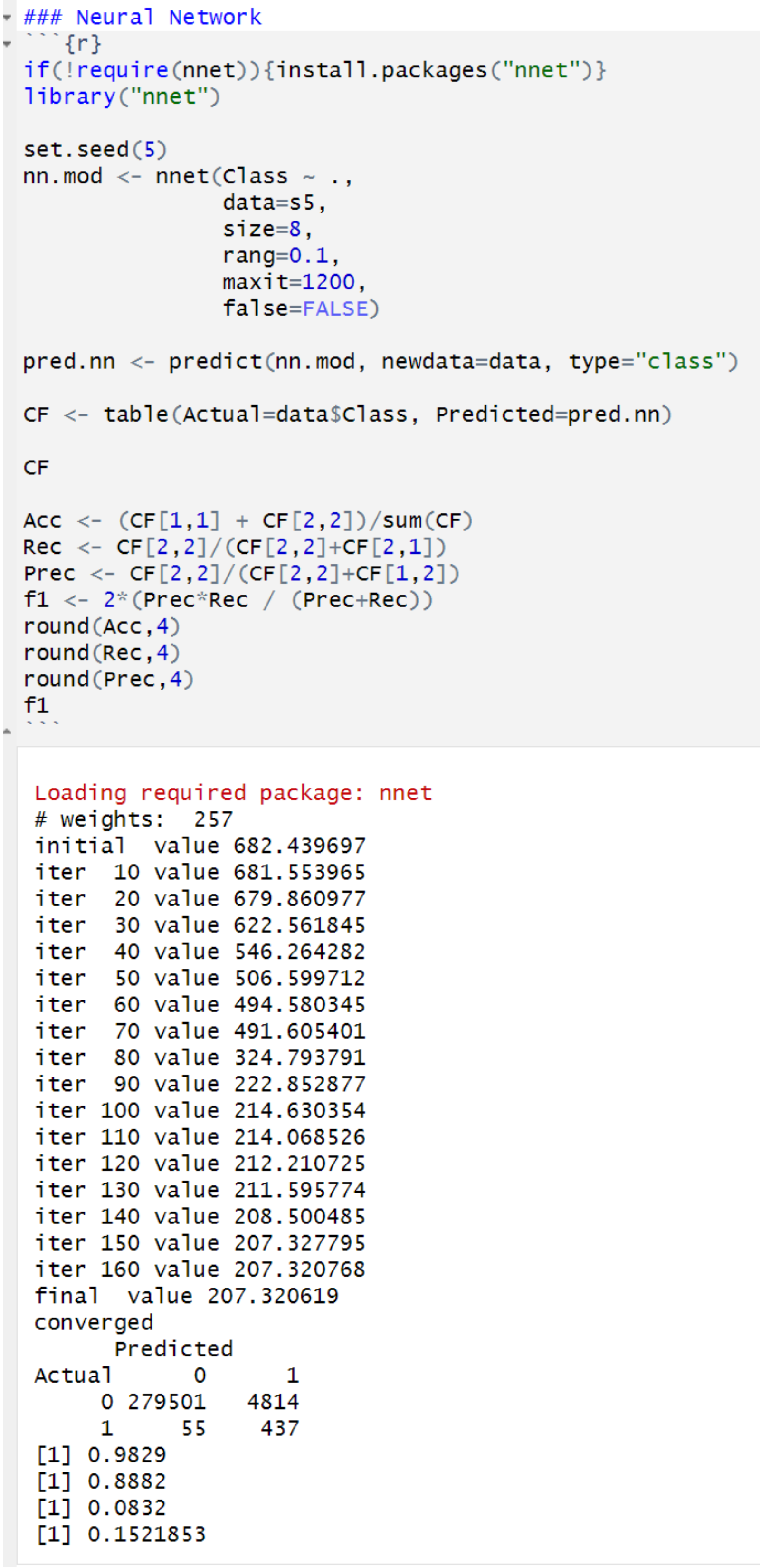
Sample	AICs	Res. Devia	Coeff.	Accuracy	Recall	Precision	F1-score
s4	290.57	230.57	1 - 29/29	0.9704	0.9289	0.0517	0.09786915
s7	283.82	223.82	1 - 31/32	0.9696	0.9309	0.0504	0.09568578
s5	278.35	218.35	1 - 29/29	0.9689	0.937	0.0496	0.09426439
s2	288.4	228.4	1 - 29/29	0.9684	0.937	0.0489	0.09291545
s6	281.12	223.12	1 - 27/28	0.9678	0.9329	0.0478	0.09089109
s8	332.66	304.66	1 - 10/13	0.9655	0.9167	0.0441	0.08416534
s9	276.06	216.06	1 - 26/29	0.9646	0.9309	0.0436	0.08337885
s1	276.49	216.49	1 - 28/29	0.9646	0.9309	0.0436	0.08337885
s10	261.82	207.82	1 - 26/26	0.9625	0.935	0.0414	0.07926251
s3	254.72	194.72	0 - 2/29	0.9586	0.9451	0.038	0.07307299

Eventually, we decided to go with sample set s5 as it has the second highest recall rate, the third highest f1-score, and all coefficients passing z-test.

3. Predictive evaluations:

   After successfully selecting a good sample set, we want to see if our logistics regression modeling is the best choice for our training. We would want to benchmark the model we have with models from other algorithms like neural network, Naïve Bayes and Decision Tree. For each algorithm, we create a model based on sample set s5 and calculate the mentioned metrics for each of the model.

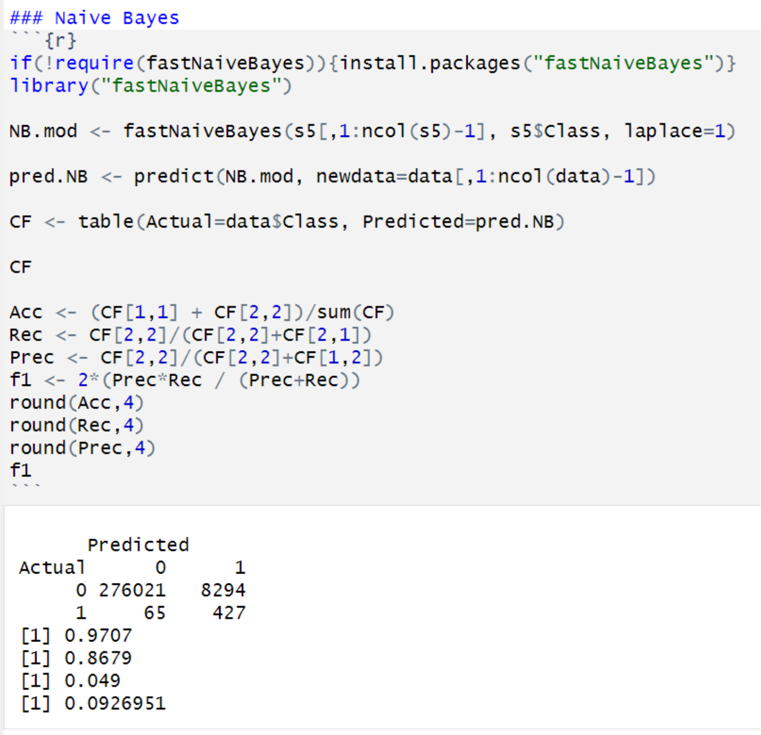
- Neural networks

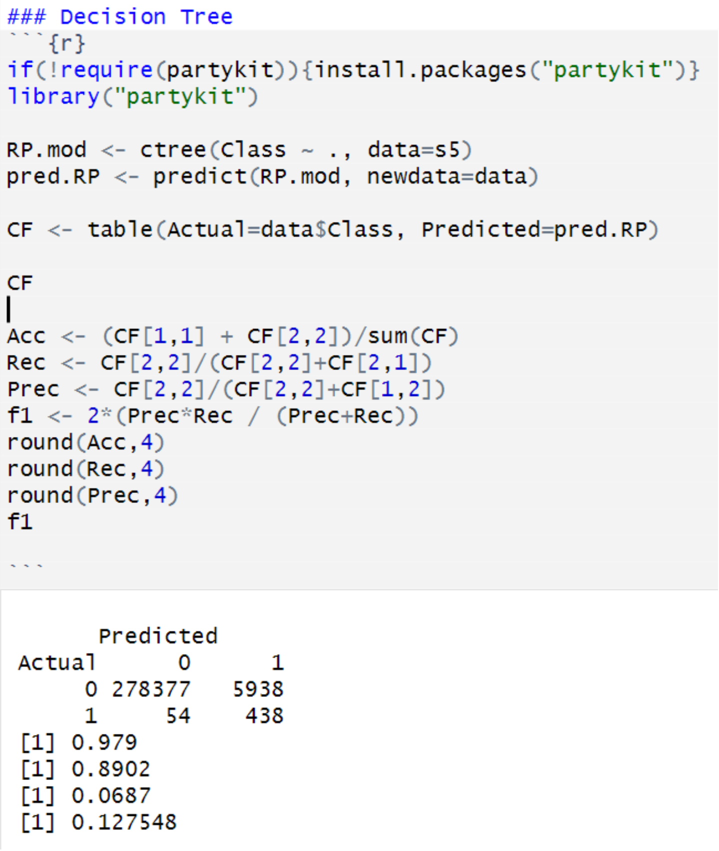


~~~
### Neural Network
```{r}
if(!require(nnet)){install.packages("nnet")}
library("nnet")

set.seed(5)
nn.mod <- nnet(Class ~ .,
               data=s5,
               size=8,
               rang=0.1,
               maxit=1200,
               false=FALSE)

pred.nn <- predict(nn.mod, newdata=data, type="class")

CF <- table(Actual=data$Class, Predicted=pred.nn)

CF

Acc <- (CF[1,1] + CF[2,2])/sum(CF)
Rec <- CF[2,2]/(CF[2,2]+CF[2,1])
Prec <- CF[2,2]/(CF[2,2]+CF[1,2])
f1 <- 2*(Prec*Rec / (Prec+Rec))
round(Acc,4)
round(Rec,4)
round(Prec,4)
f1
```
~~~

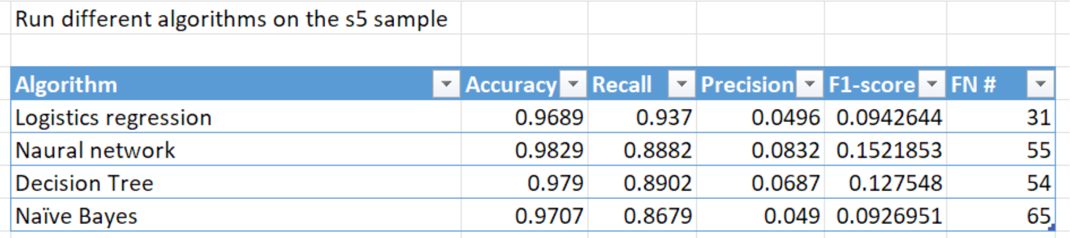
```
Loading required package: nnet
# weights:  257
initial  value 682.439697
iter  10 value 681.553965
iter  20 value 679.860977
iter  30 value 622.561845
iter  40 value 546.264282
iter  50 value 506.599712
iter  60 value 494.580345
iter  70 value 491.605401
iter  80 value 324.793791
iter  90 value 222.852877
iter 100 value 214.630354
iter 110 value 214.068526
iter 120 value 212.210725
iter 130 value 211.595774
iter 140 value 208.500485
iter 150 value 207.327795
iter 160 value 207.320768
final  value 207.320619
converged
      Predicted
Actual      0      1
     0 279501   4814
     1     55    437
[1] 0.9829
[1] 0.8882
[1] 0.0832
[1] 0.1521853
```

- Naïve Bayes



~~~
### Naive Bayes
```{r}
if(!require(fastNaiveBayes)){install.packages("fastNaiveBayes")}
library("fastNaiveBayes")

NB.mod <- fastNaiveBayes(s5[,1:ncol(s5)-1], s5$Class, laplace=1)

pred.NB <- predict(NB.mod, newdata=data[,1:ncol(data)-1])

CF <- table(Actual=data$Class, Predicted=pred.NB)

CF

Acc <- (CF[1,1] + CF[2,2])/sum(CF)
Rec <- CF[2,2]/(CF[2,2]+CF[2,1])
Prec <- CF[2,2]/(CF[2,2]+CF[1,2])
f1 <- 2*(Prec*Rec / (Prec+Rec))
round(Acc,4)
round(Rec,4)
round(Prec,4)
f1
~~~
```

```
      Predicted
Actual      0      1
     0 276021   8294
     1     65    427
[1] 0.9707
[1] 0.8679
[1] 0.049
[1] 0.0926951
```

- Decision Tree



```
### Decision Tree
```{r}
if(!require(partykit)){install.packages("partykit")}
library("partykit")

RP.mod <- ctree(Class ~ ., data=s5)
pred.RP <- predict(RP.mod, newdata=data)

CF <- table(Actual=data$Class, Predicted=pred.RP)

CF

Acc <- (CF[1,1] + CF[2,2])/sum(CF)
Rec <- CF[2,2]/(CF[2,2]+CF[2,1])
Prec <- CF[2,2]/(CF[2,2]+CF[1,2])
f1 <- 2*(Prec*Rec / (Prec+Rec))
round(Acc,4)
round(Rec,4)
round(Prec,4)
f1

```
```

```
      Predicted
Actual      0      1
     0 278377   5938
     1     54    438
[1] 0.979
[1] 0.8902
[1] 0.0687
[1] 0.127548
```
```

We put the results into another excel table for easier evaluation



Run different algorithms on the s5 sample

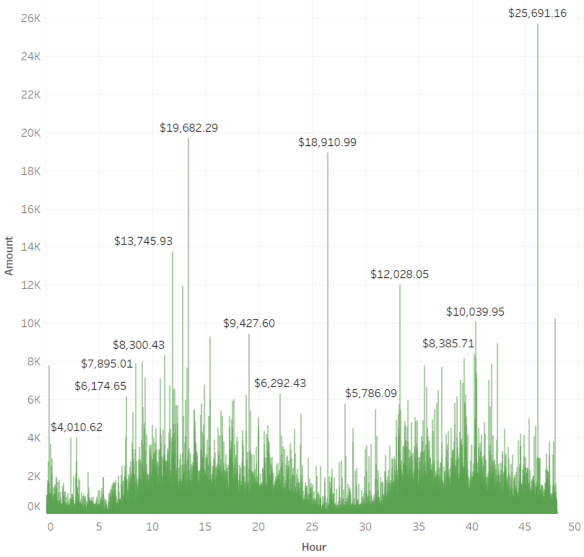
| Algorithm | Accuracy | Recall | Precision | F1-score | FN # |
|---|---|---|---|---|---|
| Logistics regression | 0.9689 | 0.937 | 0.0496 | 0.0942644 | 31 |
| Naural network | 0.9829 | 0.8882 | 0.0832 | 0.1521853 | 55 |
| Decision Tree | 0.979 | 0.8902 | 0.0687 | 0.127548 | 54 |
| Naïve Bayes | 0.9707 | 0.8679 | 0.049 | 0.0926951 | 65 |

According to this summary table, logistics regression algorithm still gives the best model with highest recall rate and lowest number of false negatives.

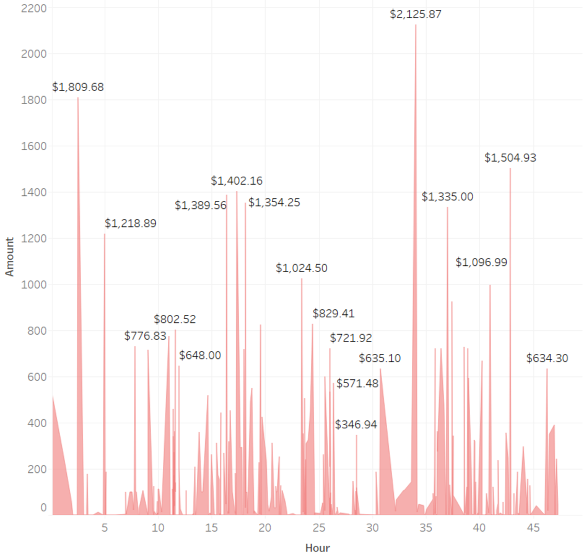
## Visualizations

1. Comparison between legit and fraudulent transactions on an hourly basis

Legit transactions by hour

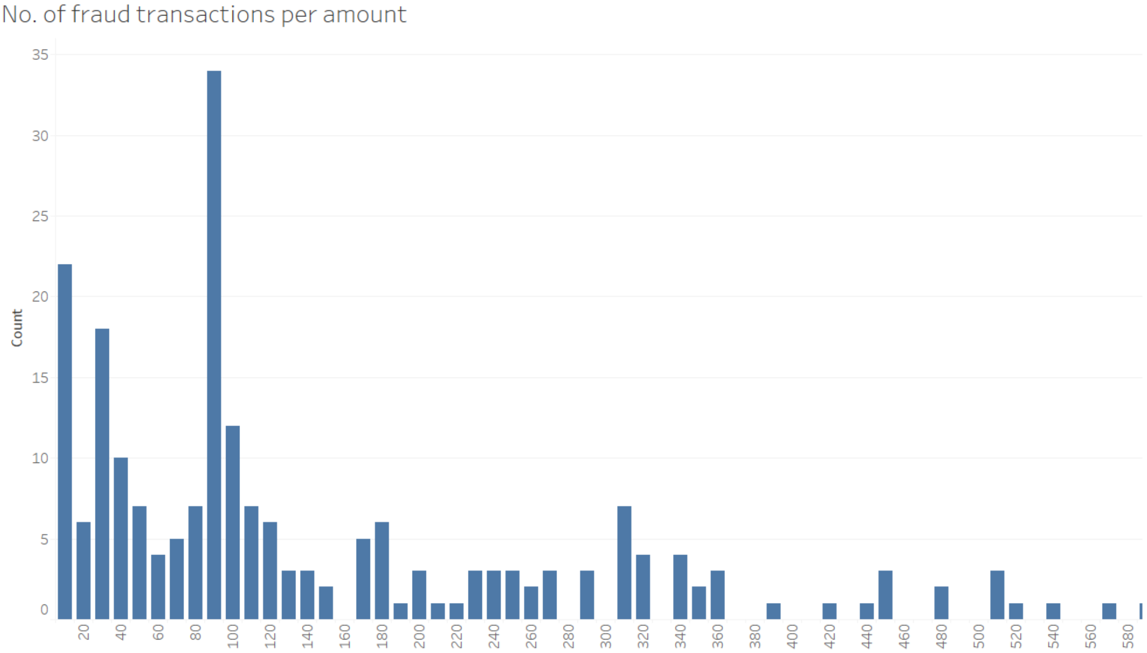


Fraud transactions by hour



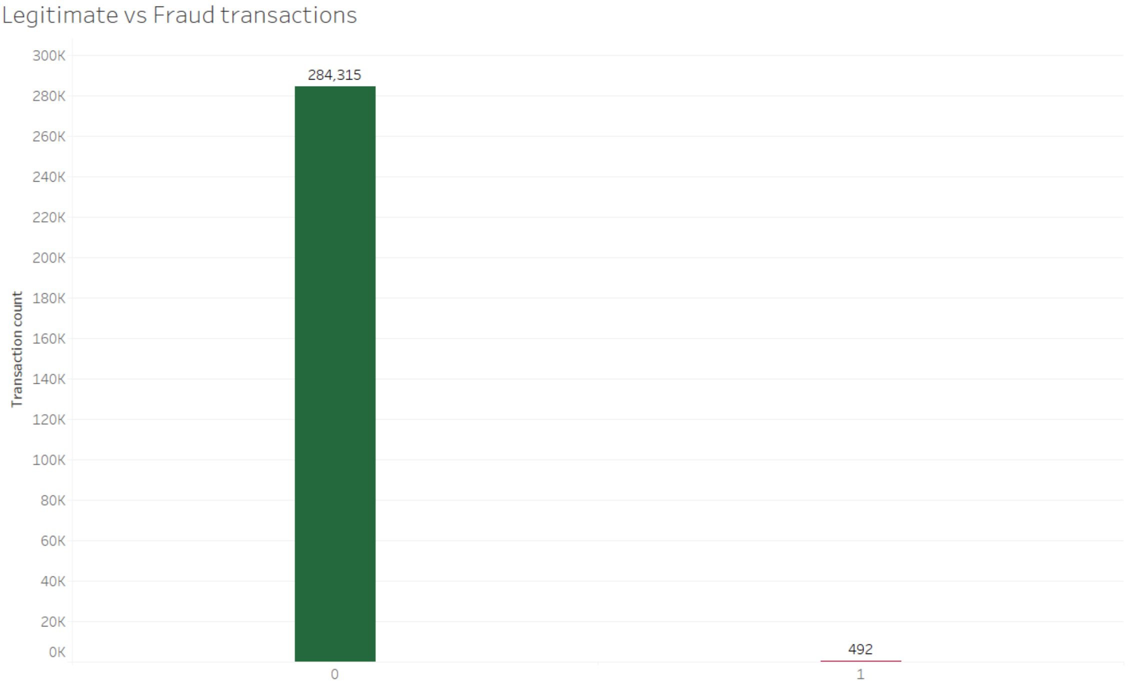
In the above diagram we can see the overlap of volume of legit transactions (green) and fraudulent transactions (red). Most fraudulent transactions were found in the first 5 hours and around 34 hours from when the data was gathered.

2. Distinct count of fraud transactions for every amount



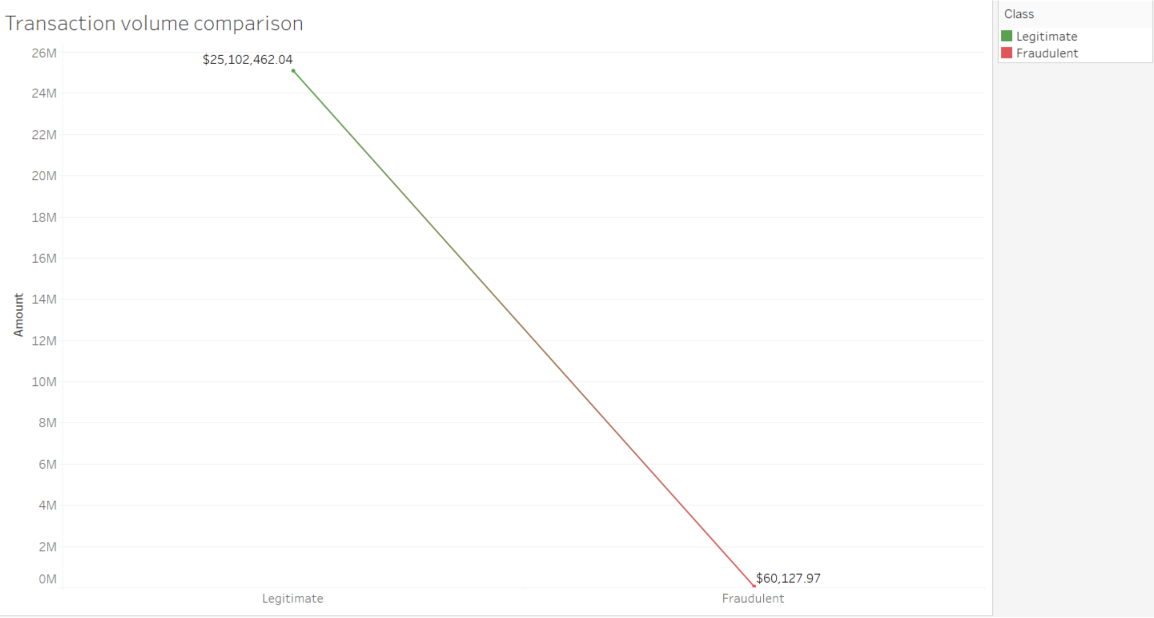
From the above graph we can see that most fraud transactions range from $80 - $100 suggesting minor transactions, such as grocery or cab expenses.

3. Comparison between legit and fraudulent transactions from total transactions



From the above we can see that from the volume of over 300,000 transactions, 284,315 were legitimate and 492 were fraudulent. We can say that we have less than 0.05% chance of fraud on a credit card transaction following this dataset.

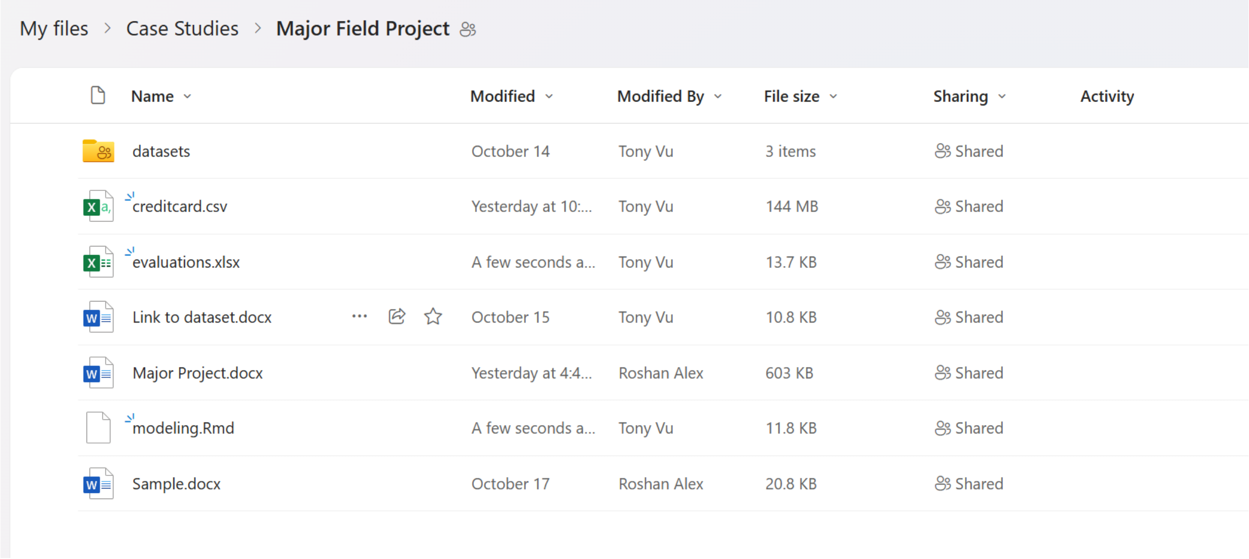
4. Comparing the total valuation of transactions by fraud and legitimate



From the above comparison we can note that transactions amounting to $60,000 were fraudulent whereas $25 Million were legitimate

## Documentation and Style

We use Google Drive to store our documents and record our steps and lesson learned.



My files > Case Studies > **Major Field Project**

| Name | Modified | Modified By | File size | Sharing | Activity |
|---|---|---|---|---|---|
| datasets | October 14 | Tony Vu | 3 items | Shared | |
| creditcard.csv | Yesterday at 10:... | Tony Vu | 144 MB | Shared | |
| evaluations.xlsx | A few seconds a... | Tony Vu | 13.7 KB | Shared | |
| Link to dataset.docx | October 15 | Tony Vu | 10.8 KB | Shared | |
| Major Project.docx | Yesterday at 4:4... | Roshan Alex | 603 KB | Shared | |
| modeling.Rmd | A few seconds a... | Tony Vu | 11.8 KB | Shared | |
| Sample.docx | October 17 | Roshan Alex | 20.8 KB | Shared | |

## Tools and Libraries

The following tools and libraries will be used for this assignment:

1. R: The primary programming language for data analysis and modeling.
2. nnet: library used for neural networks.
3. fastNaiveBayes: library used for Naïve Bayes modeling.
4. Partykit: library used for Decision Tree algorithm.
5. R markdown: An interactive environment for data analysis and code execution.
6. SQL (Structured Query Language): If needed for working with databases.
7. Tableau: Business Intelligence Tool to create visualization report.

## Proposed Allocation Project Team Roles

1. Data Research and Integration: All
2. Data Storage and Maintenance: Shubham
3. Data Collection: Tony
4. Data Quality:  Tony

5. Data Analysis and Visualization: Shubham

## Project timeline

| Date | Deliverable | Responsible |
|---|---|---|
| Nov 5 | Data collection Loading data into Db, Visualization, Quality assurance and database Schemas | All Members |
| Nov 11 | Drafting and finding the data Quality and sources for data collection | All Members |
| Nov 11 | Presentation drafting | All Members |
| Nov 17 | Quality Assurance finding outliers/Inconsistencies | All Members |
| Nov 29 | Final Edits | All Members |
|  | Report submission |  |

## References

Bachmann, J. (2019). Credit Fraud || Dealing with Imbalanced Datasets .Kaggle. https://www.kaggle.com/code/janiobachmann/credit-fraud-dealing-with-imbalanced-datasets

Rutecki, M. (2022). Best techniques and metrics for Imbalanced Dataset. https://www.kaggle.com/code/marcinrutecki/best-techniques-and-metrics-for-imbalanced-dataset

Erickson, J. (August 29, 2024). MySQL: Understanding What It Is and How It’s Used. Oracle. https://www.oracle.com/mysql/what-is-mysql/

IBM. (n.d.). What is a neural network? https://www.ibm.com/topics/neural-networks

IBM. (n.d.). What are Naïve Bayes classifiers? https://www.ibm.com/topics/naive-bayes

IBM. (n.d.). What is a decision tree? https://www.ibm.com/topics/decision-trees

Rdocumentation. (n.d.). nnet: Fit Neural Networks. https://www.rdocumentation.org/packages/nnet/versions/7.3-19/topics/nnet

Rdocumentation. (n.d.). fastNaiveBayes. https://www.rdocumentation.org/packages/fastNaiveBayes/versions/2.2.1

Rdocumentation. (n.d.). partykit - A Toolkit for Recursive Partytioning. https://www.rdocumentation.org/packages/partykit/versions/1.2-20

Geeks-for-Geeks. (September 13, 2024). What is Imbalanced Dataset. https://www.geeksforgeeks.org/what-is-imbalanced-dataset/

Geeks-for-Geeks. (January 2, 2024). Handling Imbalanced Data for Classification. https://www.geeksforgeeks.org/handling-imbalanced-data-for-classification/