Illinois Institute of Technology

Credit Card Fraud Detection with Machine Learning in R

Prof. Lulu Kang

Submitted By:
Sachin Janwalkar A20479201
FNU Deepanshu A20449479

1. Abstract

In this project, a technique for `Credit Card Fraud Detection' is developed. As fraudsters are increasing day by day. And fallacious transactions are done by the credit card and there are various types of fraud. So to solve this problem we model data sets using machine learning with credit card fraud detection. The problem includes modelling past credit card transactions with data of the ones that turned out to be fraudulent. This way models are tested individually and whatever suits the best is further proceeded. And the foremost goal is to detect fraud by filtering the techniques to get better results. In this process, we have focused on analyzing and preprocessing data sets as well as deployment of anomaly detection algorithms on the PCA transformed data.

2. Introduction

Credit card is generally referred to as a card which belongs to each customer/ cardholder, which can be used by their owners to purchase various products, goods and opt for different services within their credit card limit. Using a credit card a user can purchase particular products and opt for paying at a later period before the next billing cycle.

Credit card frauds can be performed easily without the owners knowledge and involves significantly less risk. As each fraudulent transaction appears to be a legitimate transaction, this makes detecting more challenging. In 2017, there were 1,579 data breaches and nearly 179 million records among which Credit card frauds were the most common reported form of fraud with 133,015 reports followed by employment or tax-related fraud with 82,051 reports, phone fraud with 55045 cases and bank fraud with 5,517 reports as per report released by FTC[6].

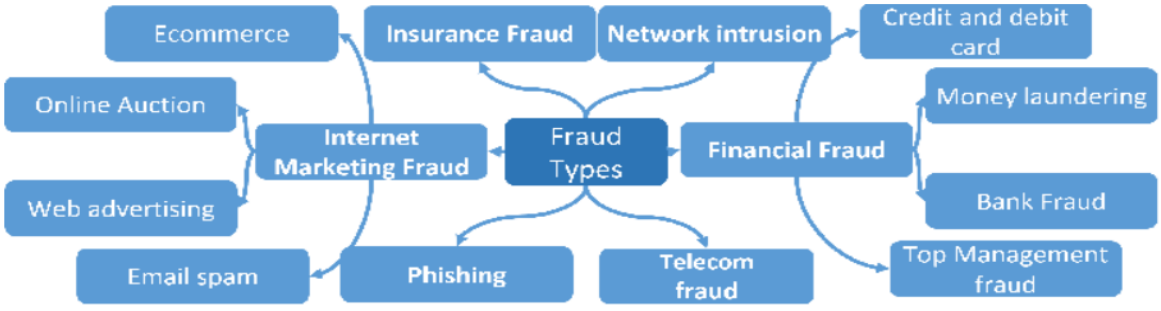

Fig 2.1 Taxonomy of Fraud

According to the US Forum report in 2017, most of the crims on credit cards are related to CNP transactions i.e. Credit cards are not present as the security of chip cards have increased. The below fig 2.2 shows CNP fraud occurred in each Year.

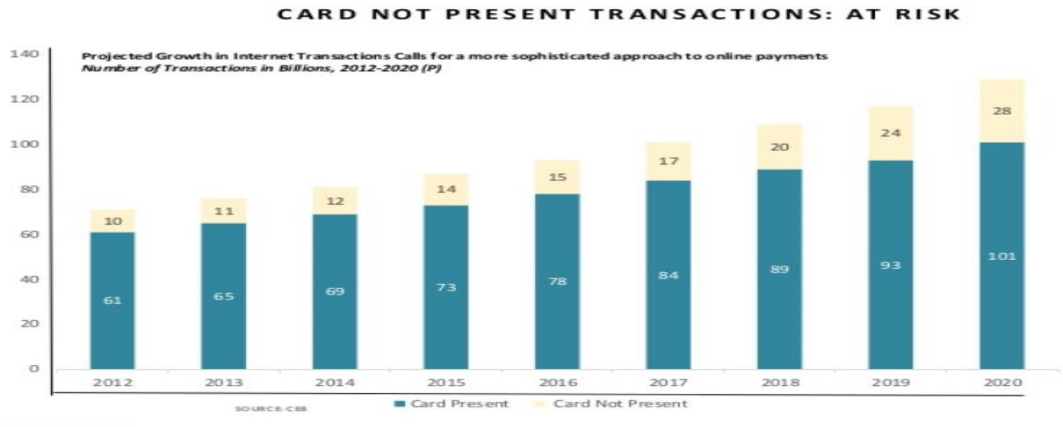

Fig 2.2 CNP Frauds

For implementing this project Multiple Supervised learning techniques are used. Various challenges faced during working with the data set that was obtained through kaggle. These are: 1) highly unbalanced dataset, 2) large numbers of predictor variables are unlabeled thus it becomes difficult to perform different analysis techniques. 3) High variability in the transaction amount makes it difficult for predictability.

Different Supervised machine learning algorithms [3] like Decision Trees, Logistic Regression and LDA, QDA, KNN, Decision tree and Artificial Neural Network are used to detect fraudulent transactions in real-time datasets. Two methods under unbalanced data set perform extraordinarily well which are Isolation forest algorithm and local outlier factor algorithm are also implemented and their results are compared with other models. The reason for these models to work is also explained. The future work will focus on solving the above-mentioned problem. The algorithm of the random forest itself should be improved.

Though supervised learning methods can be used, they may fail at certain cases of detecting the fraud cases. A model of deep Auto-encoder and restricted Boltzmann machine (RBM) [2] that can construct normal transactions to find anomalies from normal patterns. Not only that a hybrid method is developed with a combination of Adaboost and Majority Voting methods

3. Problem Statement

Card transactions are always unfamiliar when compared to previous transactions made by the customer. This unfamiliarity is a very difficult problem in real-world when it is called concept drift problems [1]. Concept drift can be said as a variable which changes over time and in unforeseen ways. These variables cause a high imbalance in data. The main aim of our project is to overcome the problem of Concept drift to implement in the real-world scenario. Table 3.1, [1] shows basic features that are captured when any transaction is made.

Attribute name	Description
Transaction ID	Transaction identification number
Time	Date and time of the transaction
Account number	Identification number of the customer
Card number	Identification of the credit card
Transaction type	ie. Internet, ATM, POS, ...
Entry mode	ie. Chip and pin, magnetic stripe, ...
Amount	Amount of the transaction in Euros
Merchant code	Identification of the merchant type
Merchant group	Merchant group identification
Country	Country of trx
Country 2	Country of residence
Type of card	ie. Visa debit, Mastercard, American Express...
Gender	Gender of the card holder
Age	Card holder age
Bank	Issuer bank of the card

Table 3.1 Features of credit card transaction.

3.1 Data Presentation:

- The dataset[7] 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.

Sr. No	Ip Features	Description
1.	Time	seconds elapsed between each transaction and the first transaction in the dataset
2.	Amount	Exact transaction amount
3.	Class	0- no fraud 1- fraud

Table 3.2 Credit card dataset description

To get more in-depth intuition on the dataset we can refer below correlation heatmap as shown in fig 3.1. As it can be seen from fig 3.1, it clearly shows that variable class is independent on the time and amount but it has dependency on the PCA computed variables.

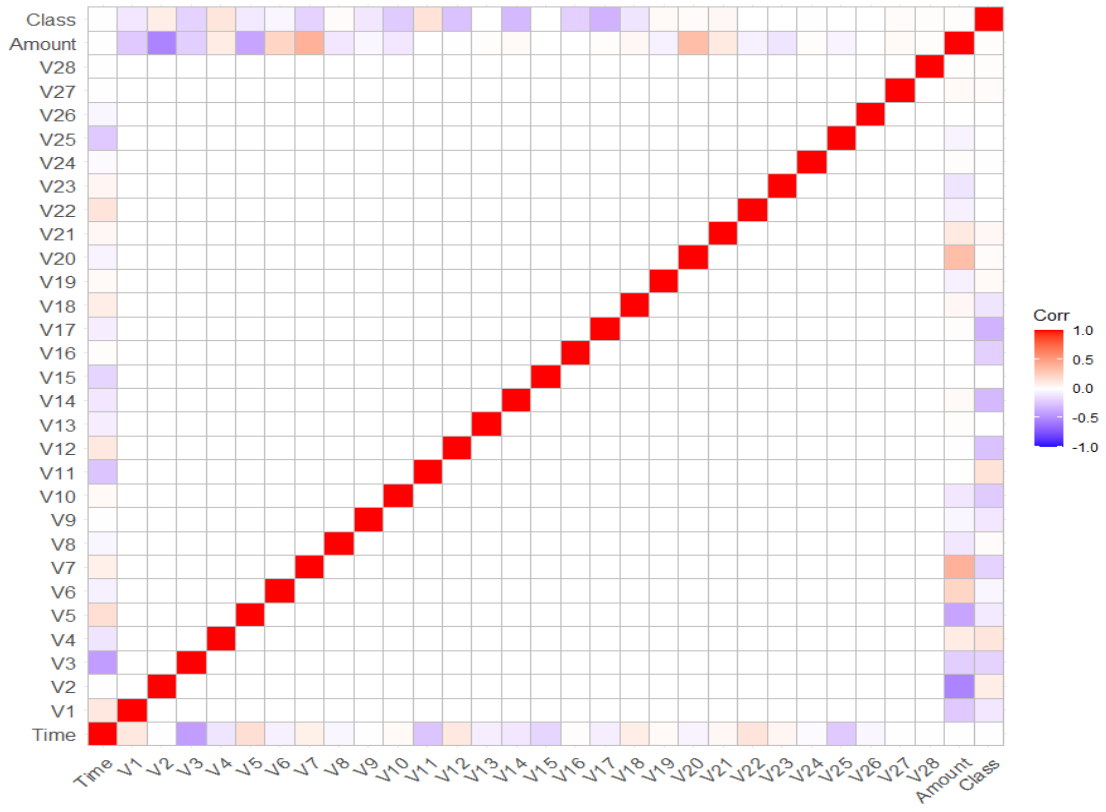

Fig 3.1 Correlation heatmap

4. Methodology

The approach that we implemented, uses the latest machine learning algorithm to detect fraudulent transactions, called outliers. Different data exploratory analysis and feature scaling techniques are used to analyse the dataset. The highly unbalanced data set can produce large bias during model training so the data set is properly balanced and various features that have very high/low correlation with the response variable are analyzed. Later different classifier algorithms are trained on the dataset and the output of all those classifiers is predicted on the test data to measure the accuracy of the model. The classifier with the best score can be chosen as the one that best predicts frauds.

We also conducted a literature survey on a novel method for fraud detection[5], where customers are grouped based on their transactions and behavioral pattern with an objective to analyse the past transaction details of customers and extract behavior patterns. Where different features of a fraud and non-fraudulent transaction are stored using sliding window strategy.
Method that is used for feature extraction of fraudulent and non-fraudulent transactions:
Our own modified Algorithm to derive aggregated transaction details and to extract card holder features using sliding window technique[1].
Algorithm:

```
L = nrow(data_set)
Genuine =null
Fraud =null
For i in range (0 to l-w+1):
        T =,null
        #For sliding window feature
        For j in range (i, i+w-1):
                a= data_set[i,]
                T = rbind(t, a)
        Endfor
        # extracting features in the window 'w'
        a1= max(data_set$amount)
        a2= min(data_set$amount)
        a3= avg(data_set$amount)
        For j in range(i+w-1):
                xi = data_set$time(tj)- data_set$time(tj-1)
```

```
        Endfor
        X = append(a1,a2,a3,a4)
        Y= LABEL(Ti)
        # Classifying transactions into fraud or not
        If Yi =0 then
                Genuine= rbind(genuine, X)
        Else:
                Fraud = rbind(fraud,X)
Endfor
```

Input : Sequence of transactions T, window size w and id of a customer holding a card
Output: Extracted features of genuine or fraud transactions.

In the sliding window method during each loop iteration features are extracted for each transaction in the window size 'w' and stored based on the behavioural pattern of each group. These extracted features can be further utilized to improve the accuracy of the model by comparing the predicted output and the features of that prediction with the group it belongs to. This technique can be highly effective in increasing the precision of prediction.
Even when we apply classifiers on the dataset, due to imbalance (shown in fig 4.3) in the dataset, the classifiers do not work well on the dataset.
Before balancing the dataset we perform exploratory data analysis in which we check density plot of amount and time features.

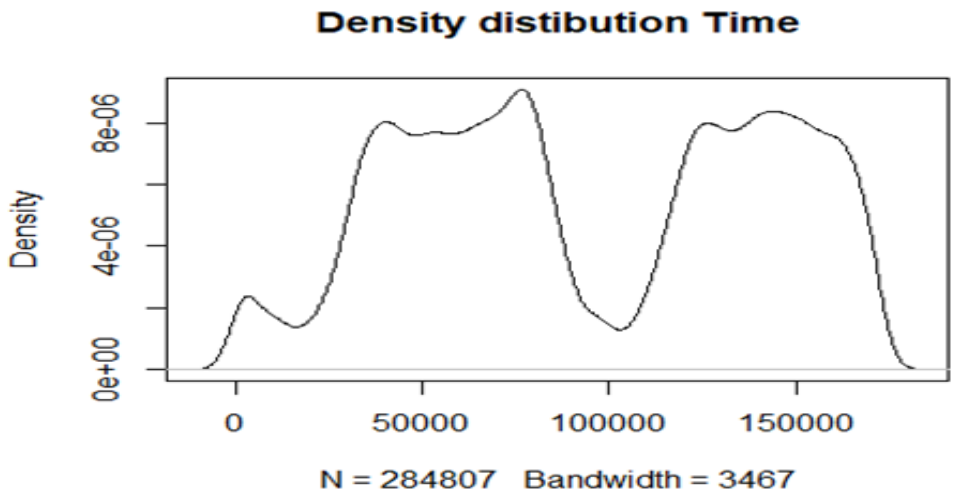

Fig 4.1 Density plot of Time

It can be seen from fig 4.1 that less transactions were made during night and more during day.

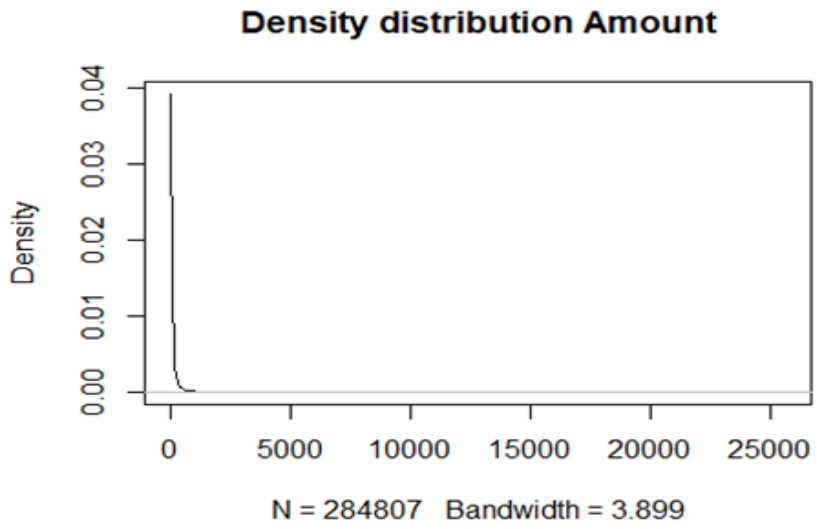

Fig 4.2 Density plot of amount

It can be seen from fig 4.2 that the majority of transactions are relatively small and only a handful of them come close to actual amounts. Then we scale parameter such amount and time and proceed to balancing the unbalanced dataset as shown in the below fig 4.3 and 4.4.

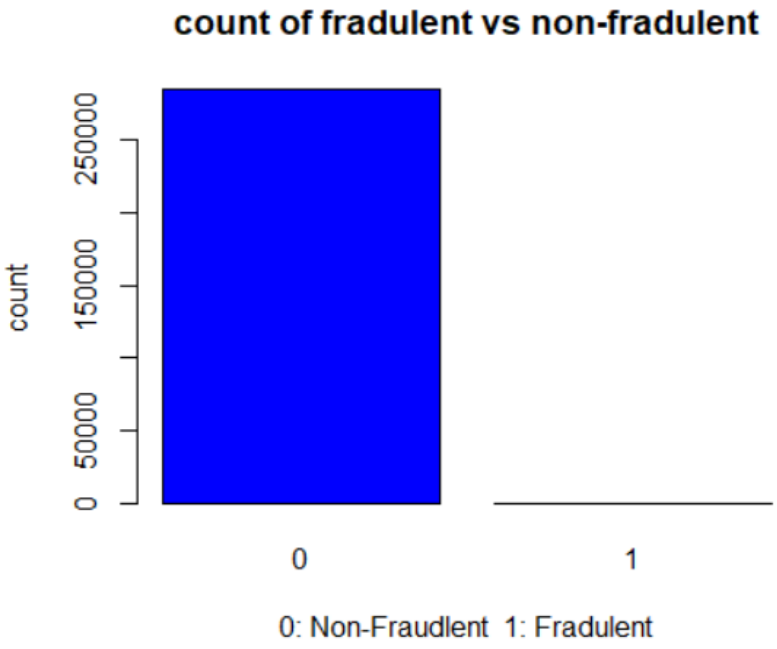

Fig 4.3 Unbalanced data set

For solving this problem, we balance the data set with equal count fraud and non-fraud transactions as shown in fig 4.2.

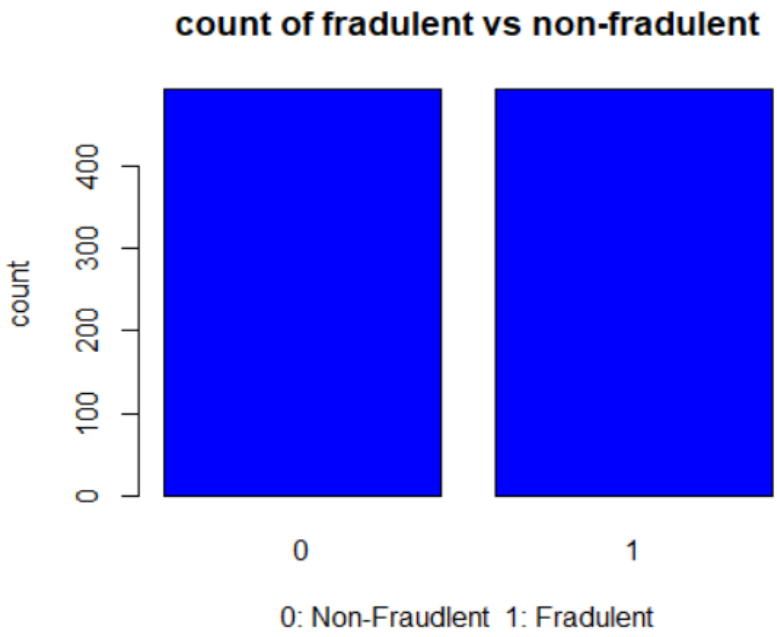

Fig 4.4 Balanced data set

On this balanced dataset we perform various Machine learning algorithms such as Logistic regression, Discriminant analysis, KNN, Neural network, Decision tree, Gradient boosting and compare their performance, which can be seen in table 4.1 below:

Ml Algorithm	Accuracy
Logistic Regression	0.883248730964467
LDA	0.883248730964467
QDA	0.903553299492386
KNN	0.928934010152284
Neural Network	0.923857868020305
Decision Tree	0.908629441624365
Gradient Boost	1
Isolation Forest	0.995084269662921

Table 4.1 Accuracy Table

Implementation of the above mentioned algorithms can be found in credit_card_fraud_project.Rmd File where each step has been implemented and executed from scratch to output the result in table 4.1.

After comparing various algorithms, we finally perform One of the newest techniques to detect anomalies is called Isolation Forests. The algorithm is based on the fact that anomalies are data points that are few and different. As a result of these properties, anomalies are susceptible to a mechanism called isolation.

This method is highly useful and is fundamentally different from all existing methods. It introduces the use of isolation as a more effective and efficient means to detect anomalies than the commonly used basic distance and density measures. Moreover, this method is an algorithm with a low linear time complexity and a small memory requirement. It builds a good performing model with a small number of trees using small sub-samples of fixed size, regardless of the size of a data set.

Typical machine learning methods tend to work better when the patterns they try to learn are balanced, meaning the same amount of good and bad behaviors are present in the dataset.

How Isolation Forests Work The Isolation Forest algorithm isolates observations by randomly selecting a feature and then randomly selecting a split value between the maximum and minimum values of the selected feature. The logic argument goes: isolating anomaly observations is easier because only a few conditions are needed to separate those cases from the normal observations. On the other hand, isolating normal observations requires more conditions. Therefore, an anomaly score can be calculated as the number of conditions required to separate a given observation.

The way that the algorithm constructs the separation is by first creating isolation trees, or random decision trees. Then, the score is calculated as the path length to isolate the observation. As it can be seen from table 3 Isolation forest gives us a very high accuracy of 99.5 % even for highly unbalanced dataset.

Another algorithm that can give high accuracy is the local outlier factor, it is an unsupervised outlier detection method which computes the local density deviation of a given data point with respect to its neighbors. It is considered as outlier samples that have a substantially lower density than their neighbors.

Both Isolation tree and LOC are performed on an unbalanced data set, but because of vector memory limit constraints LOC cannot be executed.

5. Results and Analysis

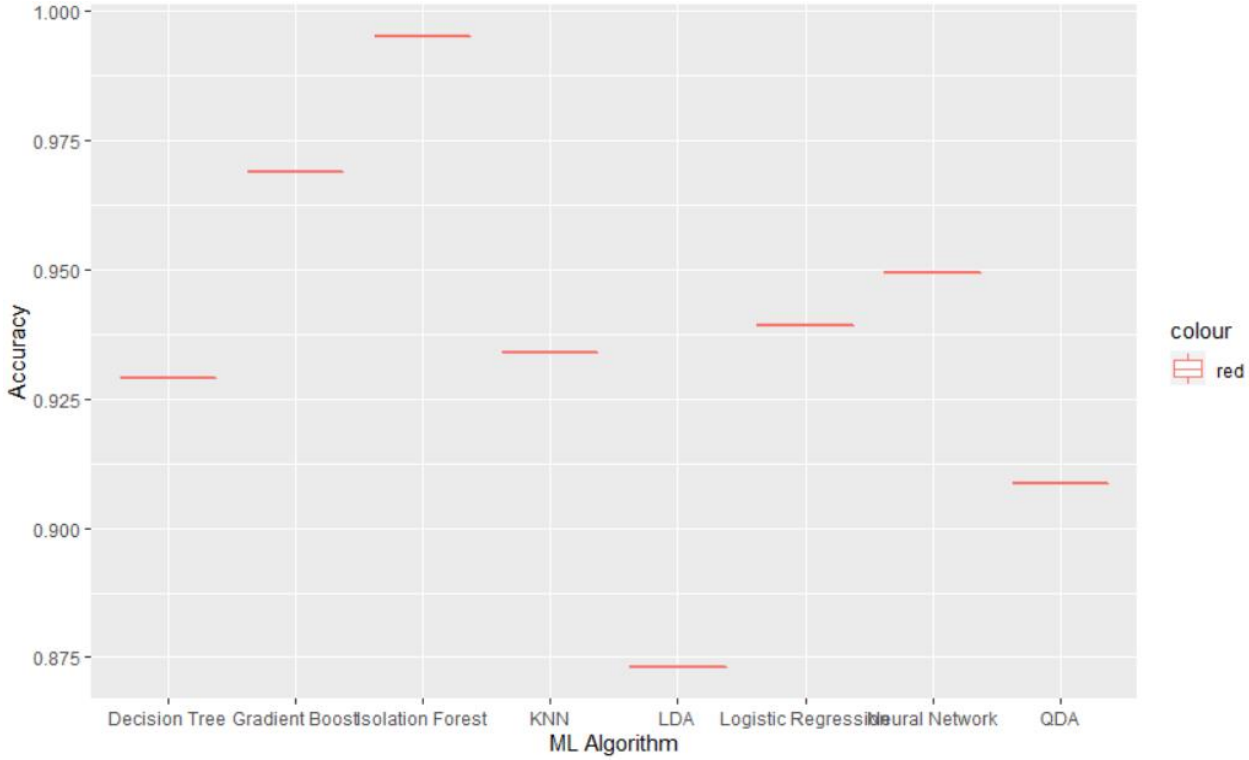

Fig 5.1 Barplot comparison

Fig 5.1 shows the comparison between different ML Algorithms on the balanced as well as unbalanced dataset. It can be seen that Isolation forest gives high accuracy predictions even on highly unbalanced datasets. Decision tree performs really well with accuracy close to 97 percent, which can be further improved by ensemble learning techniques such as use of multiple trees to predict the outcome, Random forests in general perform better than a single decision tree as they manage to reduce both bias and variance.

	ML.Algorithm	Accuracy
1	Logistic Regression	0.9390863
2	LDA	0.8730964
3	QDA	0.9086294
4	KNN	0.9340102
5	Neural Network	0.9492386
6	Decision Tree	0.9289340
7	Gradient Boost	0.9687000
8	Isolation Forest	0.9950843

Table 5.1 Accuracy Table

Accuracy of Neural networks can be further improved by tuning its hyper parameters such as adding more hidden layers, neurons, changing activation function and deep learning for auto feature selection which can be explored further as it is one of the fastest growing techniques out there. When we use deep architecture then features are created automatically and every layer refines the feature [4].

6. Conclusion

Credit card fraud is without a doubt an act of criminal dishonesty and Finding out the Fraud transactions is not an easy problem. It needs very organized planning before implementing the machine learning algorithms directly. It is also an application of business analytics, data science and machine learning which makes sure that the money spent by the customer is secure and not easily tampered with.

This project explains in detail, how machine learning can be applied to get better results in fraud detection along with the algorithm, pseudocode, explanation of its implementation and experimentation results.

Since the entire dataset consists of only two days transaction records, it's 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 the time as more data is put into it.

7. Bibliography

[1]Jiang, Changjun et al. “Credit Card Fraud Detection: A Novel Approach Using Aggregation Strategy and Feedback Mechanism.” IEEE Internet of Things Journal 5 (2018): 3637-3647.

[2]Pumsirirat, A. and Yan, L. (2018). Credit Card Fraud Detection using Deep Learning based on Auto-Encoder and Restricted Boltzmann Machine. International Journal of Advanced Computer Science and Applications, 9(1).

[3]Mohammed, Emad, and Behrouz Far. “Supervised Machine Learning Algorithms for Credit Card Fraudulent Transaction Detection: A Comparative Study.” IEEE Annals of the History of Computing, IEEE, 1 July 2018, doi.ieeecomputersociety.org/10.1109/IRI.2018.00025.

[4]https://papers.nips.cc/paper/2020/file/1959eb9d5a0f7ebc58ebde81d5df400d-Paper.pdf

[5]https://www.sciencedirect.com/science/article/pii/S187705092030065X

[6]https://www.ftc.gov/news-events/press-releases/2019/02/imposter-scams-top-complaints-made-ftc-2018

[7]https://www.kaggle.com/mlg-ulb/creditcardfraud