APPROACHES TO FRAUD DETECTION ON CREDIT CARD TRANSACTIONS USING ARTIFICIAL INTELLIGENCE METHODS

Aryaman Raj Saxena

ABSTRACT

In this paper, we study the problem of detecting fraudulent credit card transactions. We select the most relevant features using a heuristic approach, and fit three different model classes to a simulated dataset: Logistic Regression, Random Forests and Gradient Boosting Machines. We find that hyperparameter tuning has a big impact on the precision and recall of our classifiers. We also find that of the three classes, Gradient Boosting Machines were the best-performing model class, achieving 83% precision and 64% recall on unseen data.

1. INTRODUCTION

The number of cashless transactions is at its peak point since the beginning of the digital era and it is most likely to increase in the future. While that is an advantage and provides ease of use for customers, it also creates opportunities for fraudsters. Only in 2016, 34,260.6 million transactions have been performed, making a total of 66,089 transactions per second. The net loss of the global economy out of fraudulent transactions is $2.17 billion. As the loss is quite major, there is a number of research to decrease the causalities created by credit card fraud. While some of them try to solve this using mathematical rule-based algorithms, recently, machine learning and artificial intelligence techniques are in demand. That is the result of the big data collected from billions of transactions, and this data somehow could be useful in trying to predict whether a next, unknown transaction is actually a fraud or not.

An effective fraud detection system should be able to detect fraudulent transactions with high accuracy and efficiency. While it is necessary to prevent fraudsters from executing fraudulent transactions, it is also very critical to ensure genuine users are not prevented from accessing the payments system. A large number of false positives may translate into bad customer experience and may lead customers to take their business elsewhere.

A major challenge in applying ML to fraud detection is presence of highly imbalanced data sets. In many available datasets, majority of transactions are genuine with an extremely small percentage of fraudulent ones. Designing an accurate and efficient fraud detection system that is low on false positives but detects fraudulent activity effectively is a significant challenge for researchers.

In our paper, we first analyse the variables present in data to understand their predictive power to predict fraud and then apply techniques like Logistic regression, Random Forest and GBM and separate fraud transactions from non-fraud transactions. We compare the effectiveness of these approaches in detecting fraud transactions.

2. DATA ANALYSIS AND EXPLORATION

The data used for analysis is a simulated credit card transaction dataset containing legitimate and fraud transactions from the duration 1st Jan 2019 - 31st Dec 2020. It covers credit cards of 1000 customers doing transactions with a pool of 800 merchants. This was generated using Sparkov Data Generation | Github tool created by Brandon Harris. This simulation was run for the duration - 1 Jan 2019 to 31 Dec 2020. The files were combined and converted into a standard format. The simulator has certain pre-defined list of merchants, customers and transaction categories. And then using a python library called "faker", and with the number of customers, merchants that you mention during simulation, an intermediate list is created. The source of the data is: https://www.kaggle.com/datasets/kartik2112/fraud-detection/metadata

The training dataset contains 23 columns such as the date, time and month of the credit card transaction, the merchant, the spending category, the amount of transaction, and personal information about the credit card holders, including their names, genders, locations and birthdays. It also contains a column called "is_fraud" which marks fraudulent transactions as 1 and non-fraudulent as 0. There is no missing data in the dataset and we also remove any duplicated observations in the data set to make it ready for further analysis. The train data has 1,296,675 rows and test data has 555,719 rows. The dataset is highly skewed with only 7506 (or 0.58%) of the transactions are fraudulent. That means the data is highly unbalanced with respect to the target variable.

We now explore the relation between each of the independent variables with the dependent variable – fraud.

2.1 Gender

There are 54.7% Females and remaining Males in the data. The fraud rate across both categories is similar as seen in table below:

Table 1: Fraud Rate – By Gender

Gender	Fraud Rate
Male	0.64%
Female	0.52%

The distribution of fraud vs non fraud is also similar by gender as seen in histogram below.

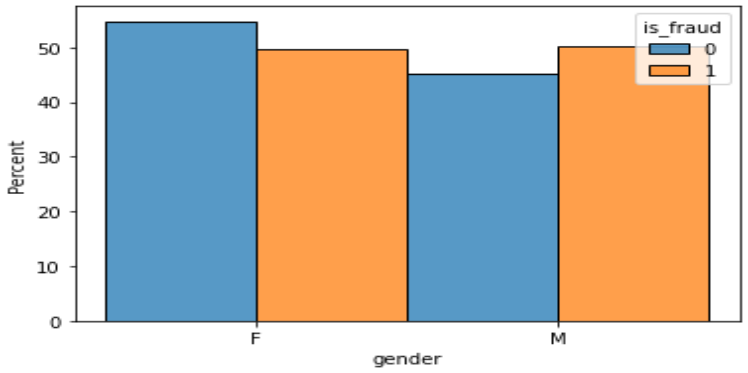

Figure 1: Fraud Distribution – By Gender

Hence gender doesn't seem to be an important variable to predict fraud and is not further explored during modelling exercise.

2.2 Transaction spending category

The spending category is available in the data and analysed to understand its relation with fraud. It is observed that certain categories like grocery_ (1.41%), misc_net (1.44%) and shopping_net (1.75%) have a higher fraud rate and could be significant in predicting fraud. The fraud rate across categories is seen in table below:

Table 2: Fraud Rate – By Spending Category

Spending Category	Fraud Rate
entertainment	0.25%
food_dining	0.17%
gas_transport	0.47%
grocery_net	0.29%
grocery_pos	1.41%
health_fitness	0.15%
home	0.16%
kids_pets	0.21%
misc_net	1.45%
misc_pos	0.31%
personal_care	0.24%
shopping_net	1.76%
shopping_pos	0.72%
travel	0.29%

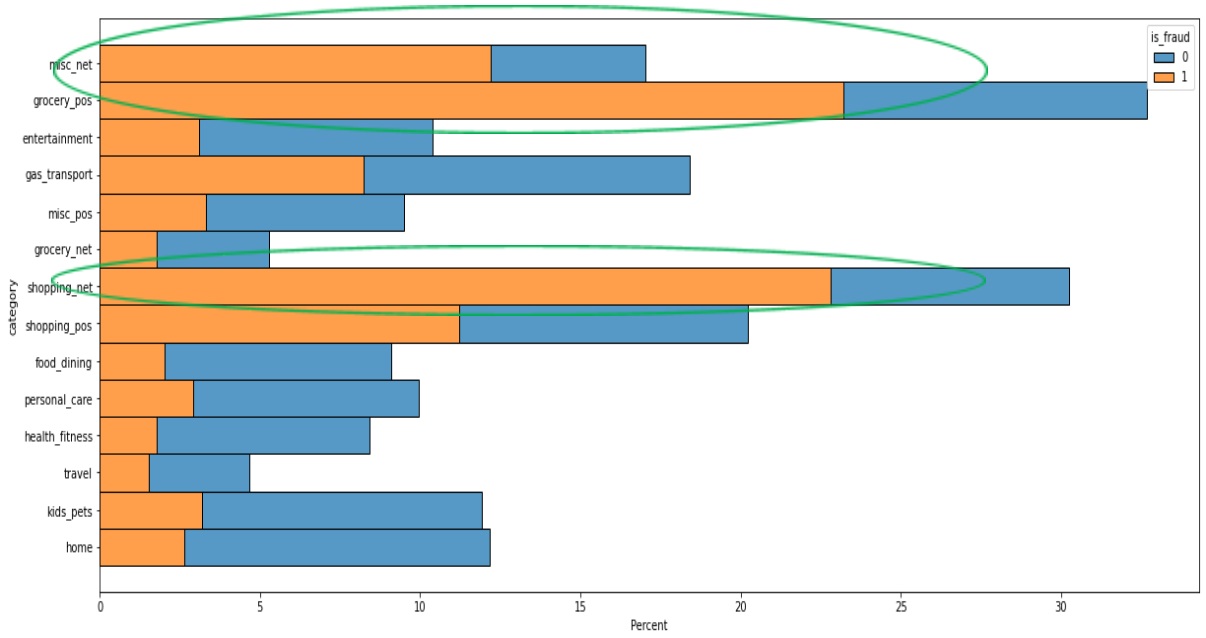

Figure 2: Fraud Distribution – By Spending Category

The distribution of fraud vs non fraud also shows a higher number of frauds in some categories (as highlighted in the histogram above).

The variable will be further explored during the modelling exercise.

2.3 Age

Age is explored as the next variable as a predictor of fraud. The fraud rate across age bands is seen in the table below. The table shows no significant difference in fraud rate across different age bands.

Table 3: Fraud Rate – By Category

Age Band	Fraud
20-40	0.51%
40-60	0.54%
60-80	0.69%
< 20	0.45%
> 80	0.94%

The density plot also shows that fraud is normally distributed across age bands and hence doesn't seem to be impacting fraud.

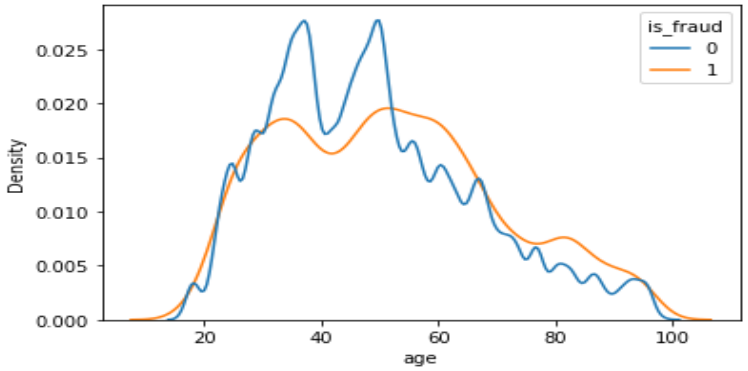

Figure 3: Fraud Distribution – By Age

2.4 Transaction time

Transaction time is explored as the next variable to analyse its relation with fraud. The density plot clearly shows more fraud during early and late hours of the day.

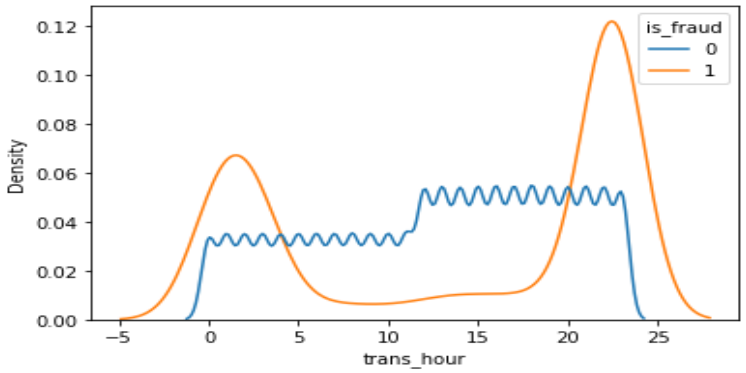

Figure 4: Fraud Distribution – By Transaction Time

The same information can be seen by analysing the fraud rate by hour of transaction. Clearly 10 PM to 3AM is the peak time for fraudulent transactions.

Table 4: Fraud Rate – By Category

Hour of Transaction	Fraud Rate
0	1.49%
1	1.53%
2	1.47%
3	1.42%
4	0.11%
5	0.14%
6	0.09%
7	0.13%
8	0.12%
9	0.11%
10	0.09%
11	0.10%
12	0.10%
13	0.12%
14	0.13%
15	0.12%
16	0.12%
17	0.12%
18	0.12%
19	0.12%
20	0.10%
21	0.11%
22	2.88%
23	2.84%

2.5 Transaction day

Transaction Day is explored as the next variable. The histogram shows that day of transaction doesn't impact fraud.

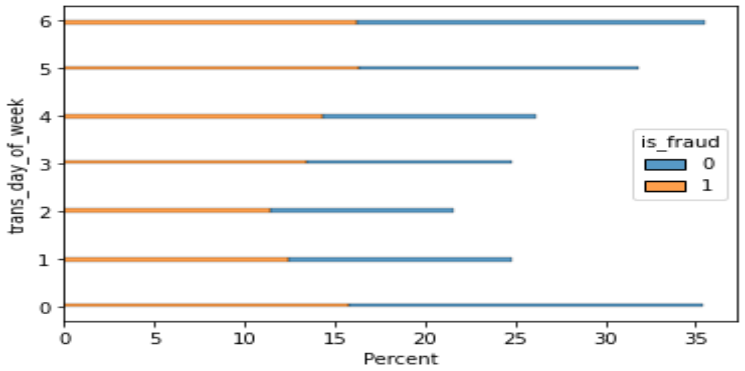

Figure 5: Fraud Distribution – By Transaction Day

The same information can be seen by analysing the fraud rate by day of transaction and it shows no significant relation.

Table 5: Fraud Rate – By Transaction Day

Transaction Day	Fraud Rate
0	0.46%
1	0.58%
2	0.66%
3	0.68%
4	0.71%
5	0.61%
6	0.49%

2.6 Transaction month

Fraud rate by month of transaction also shows no significant relation.

Table 6: Fraud Rate – By Transaction Month

Transaction Month	Fraud Rate
1	0.81%
2	0.87%
3	0.65%
4	0.50%
5	0.64%
6	0.48%
7	0.38%
8	0.44%
9	0.59%
10	0.66%
11	0.55%
12	0.42%

2.7 Transaction amount

Transaction amount is explored as the next variable to analyse its relation with fraud. The distribution of amount shows presence of extremely outliers in the data. As seen in the table below, while 95% of the transactions are less than $83, the max is $28,948.

Table 7: Distribution of Amount

Stat	Amount
Mean	70.35
std	160.31
min	1
25%	9.6
50%	47.52
75%	83.14
max	28,948

The remove the impact outlier, the histogram is plotted for transactions less than $100. The plot clearly shows higher fraud as transaction amount increases.

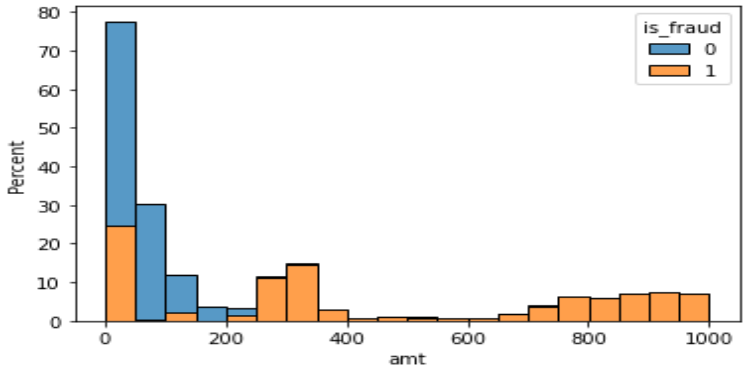

Figure 6: Fraud & Transaction amount

In the next part of the document, variables such as state, city, zip code, merchant, job are explored. These variables have many categories, making it complex to analyse each category. Hence, for the purposes of analysis, only the top 20 categories (based on occurrence in data) are analysed for each variable respectively.

2.8 State & city:

Top 20 states and cities are respectively analysed to understand impact on fraud rate. They capture ~66% and 6.7% of data. State data is statistically significant but 6.7% data is not statistically representative of full data. The fraud rate is not very different across states and is bit higher for Houston city. Only state is further explored in modelling exercise.

Table 8: Fraud Rate – By State and City

State	% Distribution	Fraud Rate	City	% Distribution	Fraud Rate
AL	3.1%	0.52%	Allentown	0.28%	0.58%
AR	2.4%	0.52%	Arcadia	0.31%	0.44%
CA	4.3%	0.58%	Birmingham	0.43%	0.20%
FL	3.3%	0.66%	Brandon	0.31%	0.52%
IA	2.1%	0.53%	Burbank	0.32%	0.43%
IL	3.3%	0.57%	Cleveland	0.35%	0.39%
IN	2.1%	0.51%	Conway	0.35%	0.37%
KY	2.2%	0.54%	Dallas	0.28%	0.74%
MI	3.5%	0.52%	Fulton	0.31%	0.28%
MN	2.4%	0.65%	Houston	0.32%	**0.94%**
MO	2.9%	0.50%	Indianapolis	0.31%	0.42%
NC	2.3%	0.49%	Lahoma	0.28%	0.44%
NY	6.4%	0.66%	Lakeland	0.28%	0.58%
OH	3.6%	0.69%	Meridian	0.39%	0.26%
OK	2.0%	0.54%	Naples	0.32%	0.70%
PA	6.1%	0.57%	Phoenix	0.39%	0.04%
SC	2.2%	0.66%	San Antonio	0.39%	0.49%
TX	7.3%	0.50%	Thomas	0.36%	0.30%
VA	2.2%	0.68%	Utica	0.39%	0.49%
WI	2.3%	0.56%	Warren	0.35%	0.72%

2.9 Zip code and merchant

Top 20 zips and merchants are respectively analysed to understand impact on fraud rate and capture ~5% of data and not statistically significant. The information is not further explored.

Table 9: Fraud Rate – By Zip Code and Merchants

Zip Code	% Distribution	Fraud Rate	Merchant	% Distribution	Fraud Rate
4287	0.24%	0.52%	Bartoletti-Wunsch	0.21%	0.56%
5461	0.24%	0.48%	Berge LLC	0.21%	0.34%
6024	0.24%	0.00%	Boyer PLC	0.27%	1.15%
12419	0.24%	0.00%	Connelly, Reichert and Fritsch	0.21%	0.45%
15484	0.24%	0.00%	Cormier LLC	0.28%	1.32%
16858	0.24%	0.35%	Cummerata-Jones	0.21%	0.44%
26292	0.24%	0.45%	Dickinson Ltd	0.26%	0.23%
28405	0.24%	0.06%	Erdman-Kertzmann	0.21%	0.52%
29819	0.24%	0.00%	Friesen-Stamm	0.21%	0.52%
34112	0.28%	0.50%	Huels-Hahn	0.21%	0.37%
38761	0.24%	0.00%	Jenkins, Hauck and Friesen	0.21%	0.55%
44233	0.24%	0.00%	Kilback LLC	0.34%	1.07%
48088	0.28%	0.64%	Kling Inc	0.21%	0.37%
49628	0.24%	0.32%	Kuhn LLC	0.27%	1.25%
72042	0.24%	0.45%	Kutch LLC	0.21%	0.55%
73754	0.28%	0.44%	Olson, Becker & Koch	0.21%	0.40%
80120	0.24%	0.00%	Prohaska-Murray	0.20%	0.82%
82514	0.27%	0.60%	Rodriguez Group	0.21%	0.41%
85173	0.24%	0.22%	Schumm PLC	0.28%	0.85%
98238	0.24%	0.29%	Stroman, Hudson and Erdman	0.21%	0.37%

2.10 Occupation and street

Top 20 occupation and streets respectively capture 11% and 5% of the data. There is no significant fraud rate differentiation across categories. The information is not further explored.

Table 10: Fraud Rate – By Occupation and Street

Occupation	% Distribution	Fraud Rate	Street	% Distribution	Fraud Rate
Agricultural consultant	0.51%	0.39%	0069 Robin Brooks Apt. 695	0.24%	0.32%
Chartered public finance accountant	0.55%	0.29%	1652 James Mews	0.24%	0.00%
Chief Executive Officer	0.55%	0.42%	2481 Mills Lock	0.24%	0.45%
Comptroller	0.52%	0.33%	2870 Bean Terrace Apt. 756	0.24%	0.45%
Copywriter, advertising	0.55%	0.63%	29606 Martinez Views Suite 653	0.24%	0.48%
Designer, ceramics/pottery	0.63%	0.15%	3379 Williams Common	0.24%	0.00%
Environmental consultant	0.58%	0.29%	4038 Smith Avenue	0.24%	0.23%
Exhibition designer	0.71%	0.55%	40624 Rebecca Spurs	0.24%	0.45%
Film/video editor	0.75%	0.45%	4664 Sanchez Common Suite 930	0.24%	0.00%
Financial adviser	0.59%	0.30%	574 David Locks Suite 207	0.24%	0.00%
IT trainer	0.59%	0.38%	594 Berry Lights Apt. 392	0.24%	0.06%
Magazine features editor	0.51%	0.61%	6033 Young Track Suite 804	0.24%	0.00%
Materials engineer	0.63%	0.75%	7202 Jeffrey Mills	0.24%	0.29%
Naval architect	0.67%	0.61%	72966 Shannon Pass Apt. 391	0.24%	0.26%
Paramedic	0.51%	0.41%	7618 Gonzales Mission	0.24%	0.36%
Podiatrist	0.51%	0.62%	7952 Karen Pike	0.24%	0.16%
Scientist, audiological	0.55%	0.50%	8030 Beck Motorway	0.24%	0.00%
Sub	0.52%	0.21%	8172 Robertson Parkways Suite 072	0.24%	0.22%
Surveyor, land/geomatics	0.67%	0.58%	854 Walker Dale Suite 488	0.24%	0.52%
Systems developer	0.59%	0.17%	864 Reynolds Plains	0.24%	0.00%

2.11 Latitude and longitude

Some non-intuitive variables such as card holder and merchant latitude and longitude are also present. The histogram shows no significant relation with fraud rate. The data is not further explored.

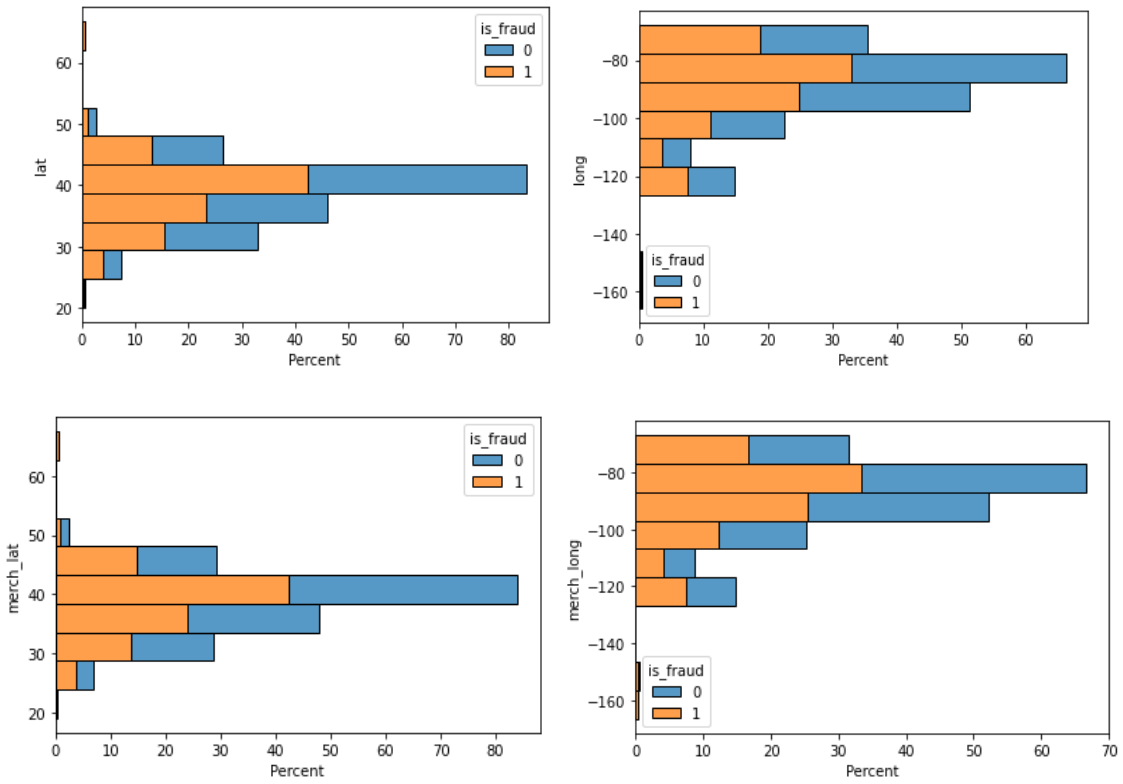

Figure 7: Fraud & Latitude/Longitude

2.11 Selected features

The key significant variables based on the above analysis are: transaction amount, transaction time and spending category. Fraud month and state are also kept in further steps. Although higher levels of data analysis and more features can be used for more professional model developments, for the sake of simplicity we will use these selected features.

This approach to feature selection is limited, because we ignore correlations between variables that could help us predict fraudulent transactions. We use this heuristic method for now due to computational limitations, but other techniques that would be better-placed for selecting appropriate features include principal component analysis and shrinkage regression.

3. MODELLING APPROACH

Post variable selection, the modelling process is initiated. The models are built on 'train' data and validated on 'test' data using 3 ML approaches – Logistic Regression, Random Forest and GBM. The models are tested based on 2 parameters – AROC value and Recall & Precision value obtained from confusion matrix.

3.1 Model Assessment Metrics

The Area Under the Curve (AUC) is the measure of the ability of a classifier to distinguish between classes and is used as a summary of the ROC curve. The higher the AUC, the better the performance of the model at distinguishing between the positive and negative classes.

A Confusion matrix is an N x N matrix used for evaluating the performance of a classification model, where N is the number of target classes. The matrix compares the actual target values with those predicted by the machine learning model. This gives us a holistic view of how well our classification model is performing and what kinds of errors it is making. For a binary classification problem, we would have a 2 x 2 matrix as shown below with 4 values:

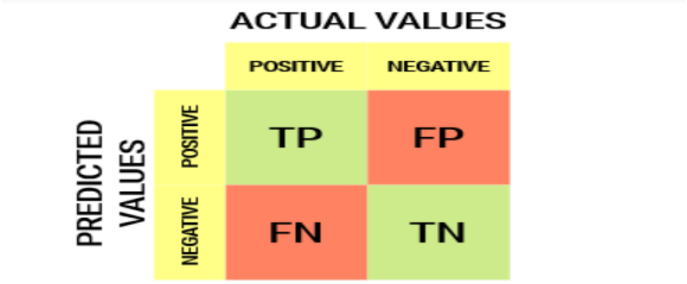

Figure 8: Confusion Matrix

True Positive (TP) - The predicted value matches the actual value. The actual and predicted value both are positive.

True Negative (TN) - The predicted value matches the actual value. The actual and predicted value is negative.

False Positive (FP) or Type 1 error - The predicted value was falsely predicted i.e. The actual value was negative but the model predicted a positive value

False Negative (FN) or Type 2 error - The predicted value was falsely predicted i.e. The actual value was positive but the model predicted a negative value

Precision = TP / (TP+FP) - This would determine whether our model is reliable or not. Precision is a useful metric in cases where False Positive is a higher concern than False Negatives.

Recall = TP/(TP + FN) - Recall tells us how many of the actual positive cases we were able to predict correctly with our model. Recall is important in medical cases where it doesn't matter whether we raise a false alarm but the actual positive cases should not go undetected!

3.2 Model Constraints

Data Imbalance: As discussed above the data is highly imbalanced. Imbalanced classification involves developing predictive models on classification datasets that have a severe class imbalance. The challenge with imbalanced datasets is that most ML techniques will ignore & have poor performance. One approach to addressing imbalanced datasets is to oversample the minority class. The simplest approach involves duplicating examples in the minority class, although these examples don't add any new information to the model. Instead, new examples can be synthesized from the existing examples. This is a type of data augmentation for the minority class and is referred to as the Synthetic Minority Oversampling Technique, or SMOTE for short.

Customer friction: The most likely outcome if a model predicts a current transaction as fraud is to decline the transaction outright to prevent any financial loss. However, we will soon see that it sometimes proves to be a bone of contention with genuine customers, who might get declined if the model has too many false positives or Type 1 errors. Though we may never achieve 100% accuracy in a real-world scenario, it is desirable for the model to be as accurate as possible to minimize any real customer friction.

Real-Time Detection: For most of the fraud detection models in practice they have to work under very stringent timing conditions. We can take an example of a transaction-level fraud detection model. This model has to run and give the decision as to whether the current transaction is fraud or not within a fraction of a second. If we employ a time-consuming but highly accurate model, we might irritate the customer who is waiting to do the transaction, and if we process too fast, we may improve on customer experience, but it might lose out on accuracy. So, it is a very thin line that we have to tread on while developing such fraud detection models.

3.3 Hyperparameter Tuning

Hyperparameter tuning for Random Forest has been summarised by the table below.

Table 11: Hyperparameter tuning – Random Forest

	Champion	Iteration 1	Iteration 2	Iteration 3	Iteration 4	Iteration 5	Iteration 6	Iteration 7	Iteration 8	Iteration 9	Iteration 10	Iteration 11	Iteration 12	Iteration 13	Iteration 14	Iteration 15
max_depth	4	5	5	4	5		4	7	15	25	50	50	50	7	10	10
random_state	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
n_estimators	200	200	200	300	300							200	100	100	100	300
max_features	50	50	75	50	n_features											
AUC Score - Train	92%	94%	93%	91%	Crash	100%	86%	87%	92%	95%	100%	100%	100%	87%	90%	90%
Precsion - Train	18%	13%	15%	19%		100%	13%	23%	31%	49%	93%	94%	93%	23%	30%	26%
Recall - Train	86%	91%	89%	84%		100%	75%	75%	85%	91%	100%	100%	100%	75%	80%	81%
AUC Score - Test	91%	93%	92%	90%		81%	86%	86%	88%	88%	83%	83%	83%	86%	87%	87%
Precsion - Test	13%	9%	10%	14%		81%	10%	18%	31%	50%	76%	76%	76%	17%	25%	23%
Recall - Test	84%	89%	88%	81%		63%	74%	73%	76%	76%	67%	66%	67%	73%	74%	74%
~Run time (mins)		33	55	55		55	10	10	15	18	30	48	22	6	50	50
AUC Score - diff	-1%	-1%	-1%	-1%		-18%	-1%	-1%	-4%	-8%	-16%	-17%	-17%	-1%	-3%	-3%
Precsion - diff	-5%	-4%	-5%	-5%		-19%	-3%	-5%	0%	1%	-17%	-18%	-17%	-6%	-5%	-3%
Recall - diff	-2%	-2%	-1%	-3%		-37%	-1%	-2%	-9%	-15%	-33%	-34%	-33%	-2%	-6%	-7%
Comments	Baseline	Precision worsens	Precision worsens	Minor improvement		Over fitted	Precision worsens	Max Precision and with balanced Test data metrics	Higher Precison but Over fitted	Higher Precison but Over fitted	Higher Precison but Over fitted	Higher Precison but Over fitted	Higher Precison but Over fitted	Similar to Iteration 7	Best Results	

The table depicts how the model performs across different parameter values – mainly altering 'max_depth', 'n_estimators' and 'max_features'.

In the first few iterations, we vary the value of max_features between 50 and 75. Due to the slight increase in performance with its value as 75, we stick with it for the rest of the iterations.

On increasing max_depth, we observe that precision and recall values both increase however our model becomes prone to overfitting. Thus, we stick to a lower value of 100.

On increasing n_estimators, the program crashes forcing us to stick under 300. For the same max_depth of 50, different values of 100, 200, 300 of n_estimators are used independently to check for a relation between the two from iterations 11-13.

4. RESULTS

The results across the 3 methods are summarised in the table below.

Table 12: Summary of Model Results

	ROC (Train)	ROC (Test)	Precision (Train)	Precision (Test)	Recall (Train)	Recall (Test)
Logistic Regression	88.2%	86.1%	4%	3%	90%	84%
Random Forest	100%	83%	93%	76%	100%	67%
GBM	86.2%	81.8%	96%	83%	73%	64%

While the ROC and Recall value is high for Logistic model compared to GBM, the precision value is very low. This implies that the model generates a lot of false positives or identifies lot of transactions as fraud when they are actually not. This is further explored by iterating the threshold, the default threshold being 50%.

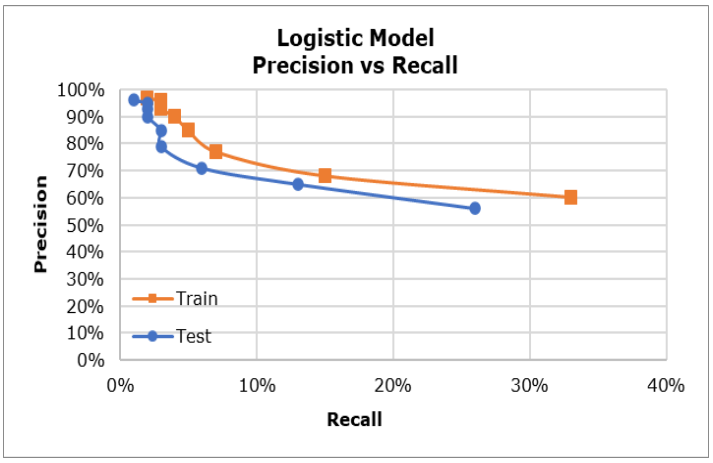

Figure 9: Precision vs Recall – Logistic Model

Table 13: Precision vs Recall in Tabular form – Logistic Model

	Train			Test		
Threshold	ROC	Precision	Recall	ROC	Precision	Recall
0.1	100%	40%	100%	93%	29%	87%
0.2	100%	61%	100%	90%	46%	81%
0.3	100%	75%	100%	88%	59%	76%
0.4	100%	86%	100%	86%	68%	71%
0.5	100%	93%	100%	83%	76%	66%
0.6	98%	97%	96%	80%	82%	61%
0.7	94%	99%	87%	77%	88%	54%
0.8	88%	100%	76%	72%	92%	44%
0.9	81%	100%	61%	64%	94%	28%

Above charts depict that precision remains low till 70% cut-off but increases to 15% and 33% when cut-off is set to 80% or 90% respectively, still lowest amongst all methods. Lower precision value leads to Customer friction as explained above. However, it is to be noted that the results between Test and Train dataset are similar and logistic model is the most robust.

Below chart also shows the important features in logistic model. Hour 22 and 23 i.e., transactions from 10PM to midnight are most significant followed by transaction category being gas, grocery, and miscellaneous and then early hours midnight to 3AM.

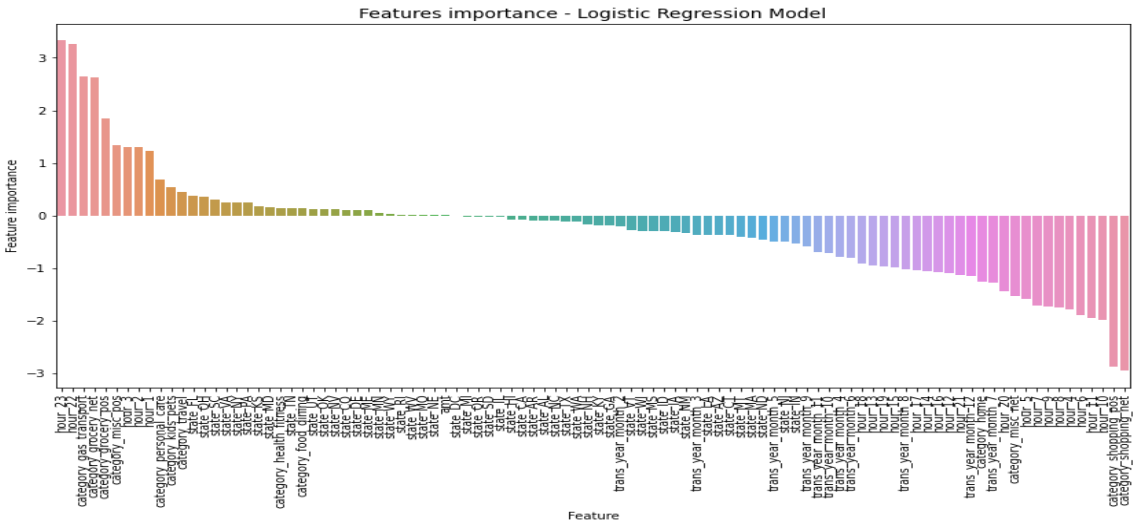

Figure 10: Feature Importance – Logistic Model

Random Forest gives superior results almost across ass metrics amongst the 3 methods. However, maximum deterioration is observed in test data. Various thresholds are also explored to understand if robustness can be improved. The default cut off of 50% seems fine to be a fine mix of precision, recall and train vs test results. While a lower cut-off has more robust results, the precision falls. Similarly, for higher cut offs, the recall value is very low and falls drastically from train to test. Hence better results cannot be achieved by changing cut-off.

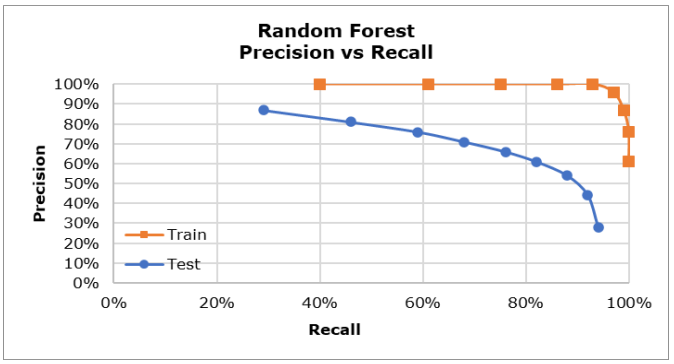

Figure 11: Precision vs Recall – Random Forest

Table 14: Precision vs Recall in Tabular form – Random Forest

	Logistic - Train			Logistic - Test		
Threshold	ROC	Precision	Recall	ROC	Precision	Recall
0.1	82%	2%	97%	82%	1%	96%
0.2	87%	3%	96%	87%	2%	95%
0.3	89%	3%	95%	88%	2%	93%
0.4	89%	3%	93%	88%	2%	90%
0.5	88%	4%	90%	87%	3%	85%
0.6	87%	5%	85%	85%	3%	79%
0.7	85%	7%	77%	83%	6%	71%
0.8	83%	15%	68%	82%	13%	65%
0.9	80%	33%	60%	78%	26%	56%

Transaction amount is the most significant variable in Random Forest followed by same variables as logistic regression such as transaction hour and category of transaction.

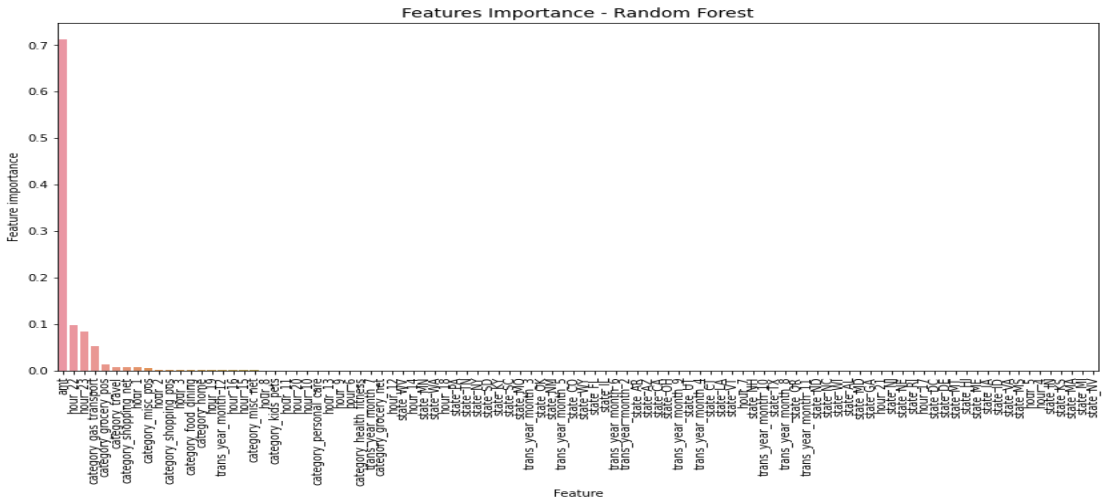

Figure 10: Feature Importance – Random Forest

The GBM model performs well across all aspects though the model is not robust as compared to logistic regression. Further, there are issues with the lack of interpretability of the features in the GBM model, making it a black box approach.

Figure 11: Feature Importance - GBM

5. CONCLUSION

As the digital era matures, the number of transactions that are processed with credit card rises continuously. Fraudsters could abuse that rise and could convert a possible advantage into a disadvantage. Research is continued to be conducted for how to detect these kinds of transactions. This survey presented how machine learning and artificial intelligence methods are utilized to detect credit card frauds. Subsequently, some common challenges such as imbalanced dataset, feature engineering, and real-time working scenarios, that are encountered during the progress are examined by the research basis.

It could be concluded that the path is not over to adapt a machine learning fraud detection system to real-time environment since most of the works conducted still prefer an offline detection mechanism and the number of research for solving the problem is relatively low. On the other hand, the improvement on managing imbalanced dataset and feature engineering challenges is apparent. There exist some operative and effective methods to solve each problem and they considerably increase the model performances. Future work can be performed to improve real-time scenarios combined with sufficient feature engineering and state-of-the-art machine learning methods.