Credit Card Transaction Fraud Detection

Harsh Shah
Wilfrid Laurier Univerisity
Waterloo, Canada
shah5610@mylaurier.ca

***Abstract*—Consumer Fraud detection is an important sector in the finance industries and public as well as private banks. Over the years, many techniques have been proposed to save consumers from these frauds. For this project, I have used machine learning based techniques to find unusual patterns of consumer's transactions which can help in blocking these kinds of likely frauds. The dataset is from IEEE-CIS Fraud detection competition on Kaggle. In this project, I have done an extensive data analysis and basic feature engineering to find some interesting insights. Model selection and performance comparison is also covered. From this project work, it can be observed that LightGBM has outperformed the other models that are Linear Regression, Decision tree and Random forest classifier with an roc_auc score of 84%.**

***Index Terms*—Machine learning, Fraud detection, Binary classification, LightGBM**

I. Introduction

A. Project Overview

Every year customers and financial companies suffer from card fraud and lose millions of dollars. Although, there exist numerous fraud detection systems that saves a huge amount of money of consumers every year, with the help of a revolutionary and promising field of machine learning, unusual patterns of consumer's transactions can be detected more efficiently in order to block these kinds of likely frauds.

For any fraud detection system, the detection algorithm is the key component. It can be divided into two methods namely, clustering methods and classification methods. But for real-world dataset, extensive data analysis and feature engineering is also vital. So in this report, I have shown some data cleaning and exploratory data analysis outcomes along with feature engineering outputs. Finally, I have discussed the methodology to select a good model for the work and at the end compared the performances.

B. Problem Statement

1. In case of a fraud occurrence, card should be blocked immediately.
2. Trained model should not produce high false-positive rate as in fraud detection it is very much unwanted.

C. Objective

1. An extensive exploratory data analysis (EDA) as dataset is large.
2. Binary classification: Train machine learning model to detect fraudulent transactions.

D. Abstract Idea

Machine learning algorithms, can help to search millions of transactional data through which pattern spotting can be done to detect fraudulent transactions. Below are some scenarios to understand this.

Scenario 1: Unusual transactional pattern: Person X doing a transaction of 5000 on every first weekend of the month for a product named as W from his debit card X on a website named Y using his laptop Z. This can be treated as a regular pattern. Imagine if on a random day a transaction of rupees 15,000 happens from that same debit card X on some different domain website and unknown device.
Possibility: This transaction has a higher probability of being fraudulent but it can also be a legitimate transaction by the same person who might have found some better product P to try and using his friend's laptop who suggested him for the same.

Scenario 2: A particular website having higher cases of frauds

Scenario 3: Some locations mapped as higher probability of frauds areas due to past occurences.

E. Dataset

The dataset for this project is taken from Kaggle competition named IEEE-CIS-Fraud-Detection[1]. This real-world industry dataset of credit card transactions is provided by Vesta Corporation, described as world's leading payment service company. It contains wide variety of features like client identity, transaction details and payment card details.

1) Input Data: The dataset is divided into two CSV files, identity and transaction and can be joined by TransactionID. "Not all transactions have corresponding identity information."[1]

a) Transaction Table: It contains the information about the transactions done by Vesta's clients. This table has total 590540 instances and 394 features. Features: TransactionID, addr1,addr2, ProductCD, P-emaildomain, R-emaildomain, Card1–Card6. All the features information is summarised in Table I.

b) Identity Table: It contains the information about the cards and clients details. To be more precise, it contains identity details like network connection details (IP,ISP) and digital device data like Browser, Device OS, Device Version etc. which is associated with the particular transaction. All the features information is summarised in Table II. The identity

TABLE I
TRANSACTION DATA

Feature	Description	Type
TransactionID	ID of transaction	ID
TransactionDT	Transaction date	time
isFraud	target variable (binary)	categorical
TransactionAmt	Transaction amount	numerical
ProductCD	Product code	categorical
card1-card6	payment card details	categorical
addr1-addr2	address	categorical
dist1-dist2	distance	numerical
P_emaildomain	purchaser email domain	categorical
R_emaildomain	receiver email domain	categorical
M1-M9	anonymous features	categorical
D1-D15	anonymous features	numerical
C1-C14	anonymous features	numerical
V1-V339	anonymous features	numerical

table has total 144233 instances and 41 features. Features: TransactionID, DeviceInfo, DeviceType, Id12-Id38 etc.

TABLE II
IDENTITY DATA

Feature	Description	Type
TransactionID	ID of transaction	ID
id01-id11	Identification data	numerical
id12-id38	Identification data	categorical
DeviceInfo	Device Information	categorical
DeviceType	device type	categorical

Both of these identity and transaction data have to be merged to perform EDA(Exploratory Data Analysis) and training. On merging the total number of instances are 590540 with 434 features.

2) Output Data: In the provided data, the target is a binary variable 'isFraud' with '0' or '1' meaning a transaction being fraudulent as 'no' or 'yes' respectively. It's distribution can be visualized in Fig. 1.

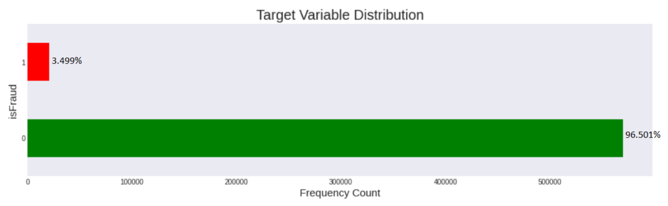


Fig. 1. Target variable distribution

II. EVALUATION METRIC

As this is a binary classification problem, accuracy metric can be think as the performance evaluator at first but as this dataset is highly imbalanced, so even if model predicts a fraudulent transactions as non fraudulent, it can achieve 99% accuracy which can be trivial for the performance measurement as the objective of the problem is different then this.
So, I have selected the ROC AUC Score for evaluation. This score is actually an area under ROC(Receiver Operating Curve) curve. It can be created by plotting the TPR that is true positive rate against the FPR means false positive rate at different threshold settings.
TP = True positive, TN = True Negative,

$$TPR = \frac{TP}{TP + FN}$$

$$FPR = \frac{FP}{FP + TN}$$

FP = False Positive, FN = False Negative.

III. DATA ANALYSIS

In this section I am going to explain data cleaning techniques, Exploratory findings of my analysis, feature engineering, data transformation.

A. Data Cleaning and Exploratory Analysis

Data Cleaning and EDA(Exploratory Data Analysis) is a step consisting mixture of techniques to better understand the data by applying visualization and wrangling techniques to remove unnecessary data and finding interesting insights.

1) Understanding the variables: In this section, few important variables and their findings are reported.

a) TransactionDT: This is a time based feature and is given in seconds. By plotting a histogram for both train and test, Fig. is generated.

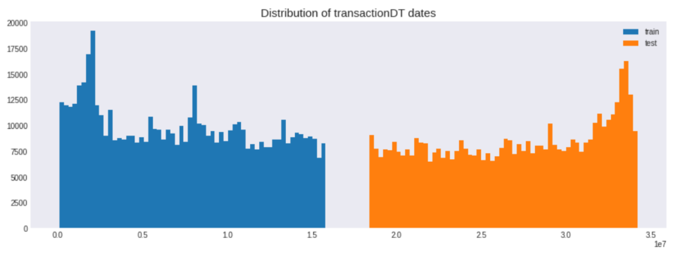


Fig. 2. TransactionDT distribution

Observation: It is interesting to observe here that train and test data DATES does not overlap which according to me means, train data was captured before test data-which is most recent and hence careful model selection and validation techniques are required.

b) Transaction Amt: Another important feature in the dataset is the transaction amount of the clients. Firstly, I applied the distribution plot but as it was skewed, no observation could be made so I did log transformation.

Observation:Here in Fig. 3, it can be observed that the graph has long tails which means it has outliers. Now to see if those outliers are actually a fraud transaction a distribution by isFraud was necessary which can be seen in Fig 4.

Observation: As in Fig. 4, the left side shows fraud transactions, the positive skewness shows the amount of transaction in fraud was high compared to non-fraud transactions.

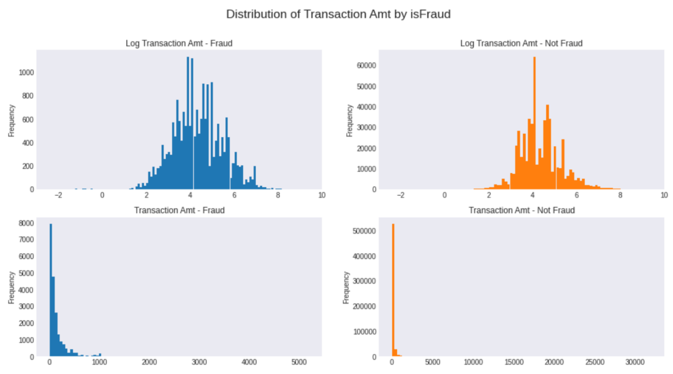


Fig. 3. Distribution of TransactionAmt

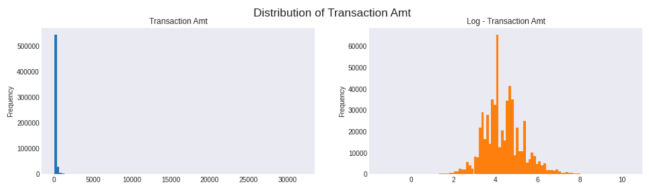


Fig. 4. Distribution of TransactionAmt by isFraud

c) ProductCD: ProductCD is the product code, the product for each transaction.

Observation: In Fig. 5., W is the most frequent value, followed by C and R. But, product C has the highest fraud percentage which means if product C is purchased, there is more chances of fraud then a non fraud transaction.

d) P_emaildomain-R_emaildomain: These both are purchaser and receiver email domains. For better analysis, I have grouped email-domains by the service provider and have grouped domains that have less than 1000 entries as 'Others'.

Observation in Fig. 6.: Most of the transactions done from P_emaildomain are from Google, Yahoo and Microsoft. Microsoft P_emaildomain has the highest rate of frauds in comparsion to other P_domains. Observation in Fig. 7.: Most of the transactions done to R_emaildomain are from Google, anonymous mail, Yahoo and Microsoft.

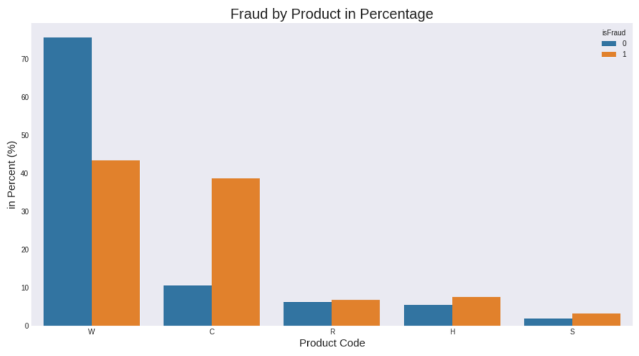


Fig. 5. ProductCD - Fraud Percentage distribution

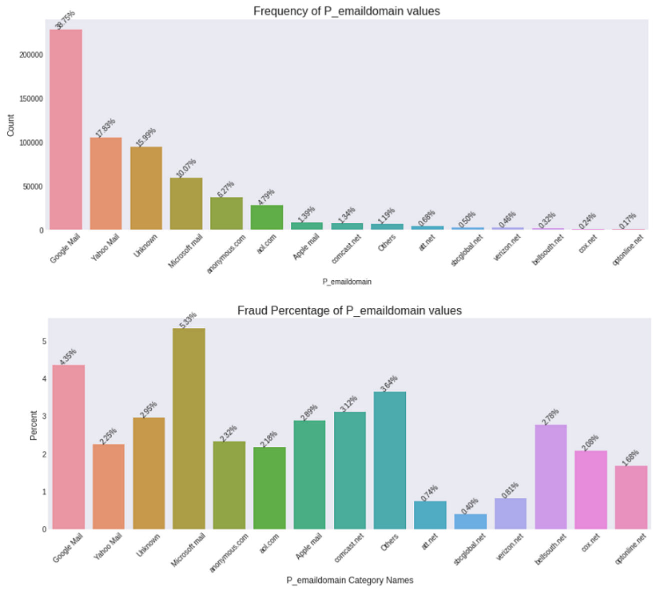


Fig. 6. Fraud Percentage by Purchaser email Domain

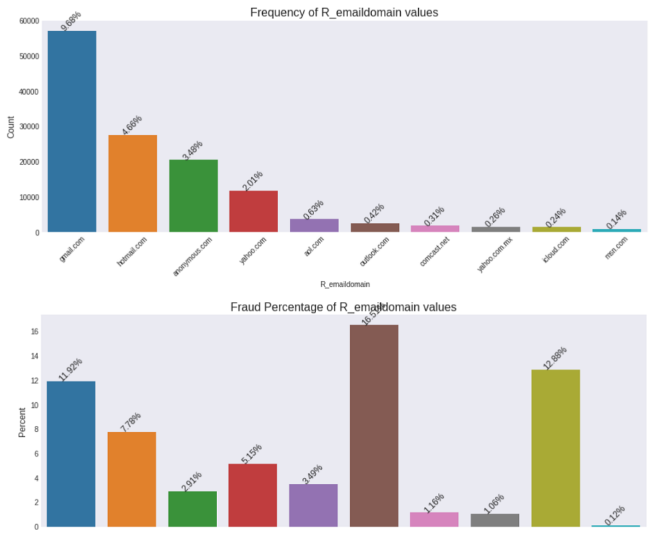


Fig. 7. Fraud Percentage by Receiver email Domain

In comparison,P_emaildomain has less missing values than R_emaildomain. AppleMail, GoogleMail and MicrosoftMail have higher frauds than other domains.

e) addr1-addr2: Data description page suggests that "both values are categorical. But these are float64 type. This means the data provider use some sort of encoding to represent each transaction's address."

addr1 Observations: In Fig. 8., the value 299.0 (7.85%), 325.0 (7.24%), 204.0(7.12%) 264.0(6.75%) are more frequent than other values. However, the fraud percentage distribution suggests that those addresses_1 with lower frequency group have more fraud cases than higher frequency categories. Location based higher frauds scenario is observed here.

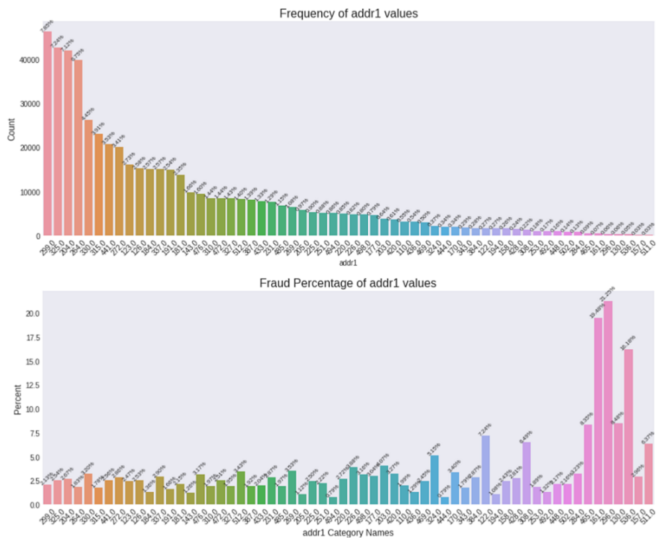


Fig. 8. Fraud Occurences of addr1 values

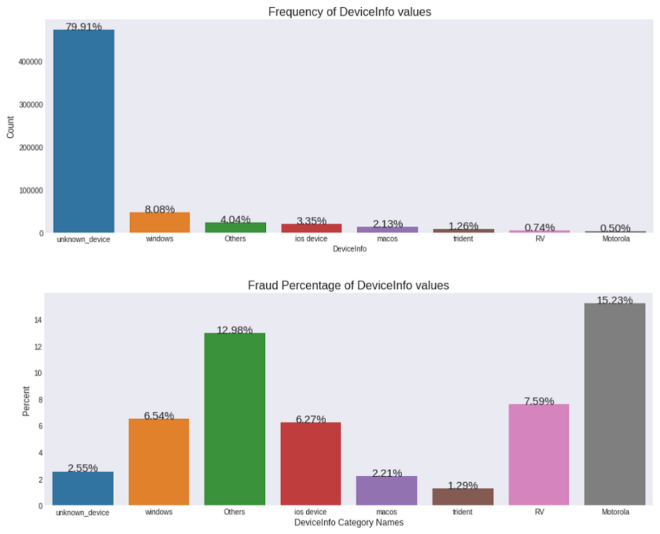


Fig. 9. Fraud occurences of DeviceInfo

f) DeviceInfo: Similar to email-domains, this feature also had so many redundant values for similar device information, so they are mapped to common category and those with less than 1000 counts are mapped to others. Also, null values are replaces by 'unknown_device'. After this mapping, a distribution plot of fraud percent for deviceinfo shows above plot in Fig. 9.
Observation: It can be seen that most of the values in DeviceInfo are null. Apart from this Motorola device has higher fraud occurences than any other devices.

IV. Feature Engineering

This step in the machine learning problem is vital and has equal importance as it can help with factors like training time, over-fitting issues, poor performance issues. For this project, I have done some basic feature selection process.

1) transaction hour feature: From transactionDT which is in seconds, I have made new feature that is transaction hour for and tried to find some interesting insights. In Fig. 10., it can be observed that, from hour 2 - 11 the number of fraud cases are more as compared to non-fraud cases.

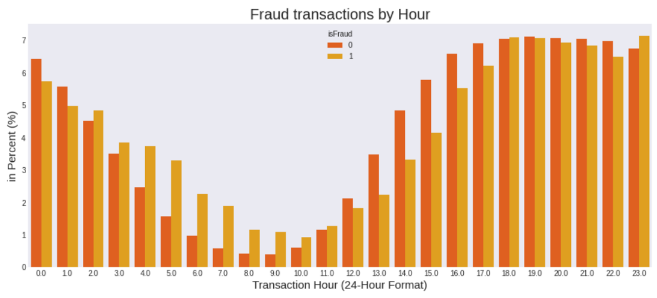


Fig. 10. Transaction Hour Fraud Distribution

2) Handling Dirty Data: For data cleaning I started my work with checking the missing data. As expected, there were many missing data with more than 90% of null values. This is actual scenario with other real world datasets too.

3) Correlation Analysis: After this, I decided to check for correlation between features as there are so many features and removing the unnecessary ones can help further. There were around 197 columns with more than 95% of correlation.

4) Data Transformation: I have used label encoding for on all categorical features. In label encoding, string based categories are encoded into numerical values. For instance, in Product Code feature, 'W' get assigned as 0, 'C' get assigned as 1 and likewise.

5) Missing Data Imputation: Next step was to impute the missing data. Typically, I have used mode to replace the NaN for all categorical features and mean for all the numerical features.

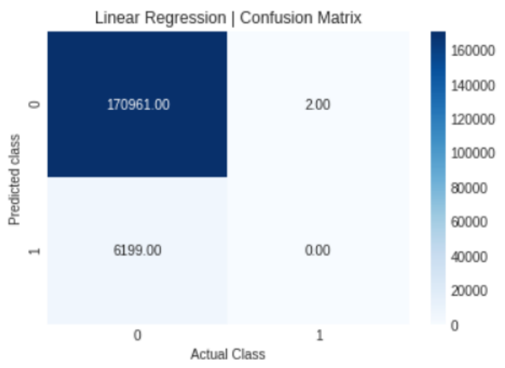
V. Model Selection

To compare different algorithmic approaches to better arrive at a good selection of model is a general practice in many research areas. This way we can compare algorithms and set the goals in a standard way to determine the performance. For this work of card fraud detection which is a binary classification problem, I have taken approach to select some common algorithms first with mostly default parameter tuning to decide further about models.

A. Linear Regression

The initial choice for model was linear regression which is the basic algorithm to understand further directions on this classification task in which class imbalance cannot be ignored.

After applying model it can be seen that it achieved 96.4% of accuracy for both train and test but when roc_auc_score was calculated, it was only 0.49 which is very less. The major objective in this work is to predict the fraud cases and not the otherwise. In below Fig. 11. having confusion matrix, we can see that randomly 0 cases are detected as fraud out of total 6199 cases which is strange and not desirable.



```
0.0    170963
1.0      6199
Name: isFraud, dtype: int64
```

Fig. 11. Confusion Matrix — Linear Regression — No tuning

To improve this, I tried to see how model responds to different thresholds. The graph can be seen in Fig. 12. It can be clearly observed that FN(False Negative) is almost constant and so this is not going to help us in predicting fraud cases. To try out, I put threshold at 0.16 and got 54% of ROC_AUC score. Improved confusion matrix is show in Fig. 13., which is able to predict 618 fraud cases out of total 6199 cases.

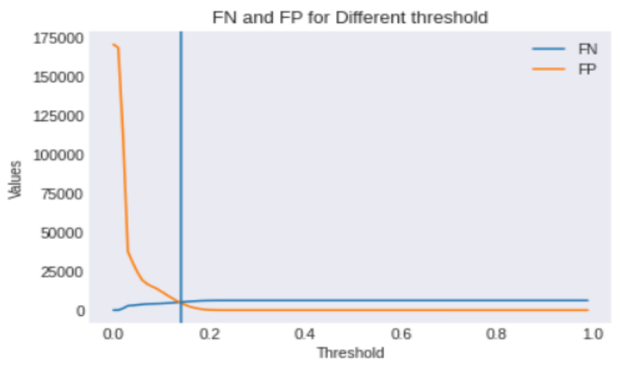


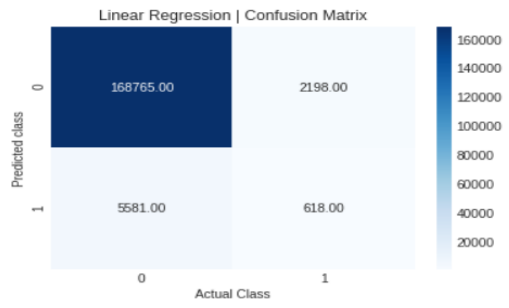
Fig. 12. FN-FP for different thresholds - LR

B. Decision Tree

As per my analysis from internet, Decision tree is also a good option to consider for fraud detection systems and so I applied it and got 0.787 of roc_auc score which is way better than simple logistic regresion. In below Fig. 14, confusion matrix shows that it is able to predict xxx numbers of fraud cases out of total xx cases.

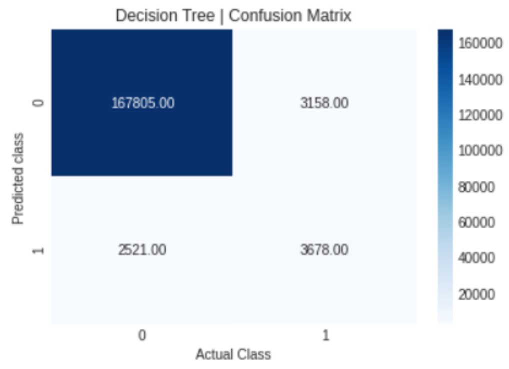
C. Random Forest Tree

Random Forest is a supevised learning algorithm which uses decision trees to classify the target variable. Confusion matrix is the evaluation metric for this algorithm. Using default parameters, I got around 0.74 roc_auc score which is not good



```
0.0    170963
1.0      6199
Name: isFraud, dtype: int64
```

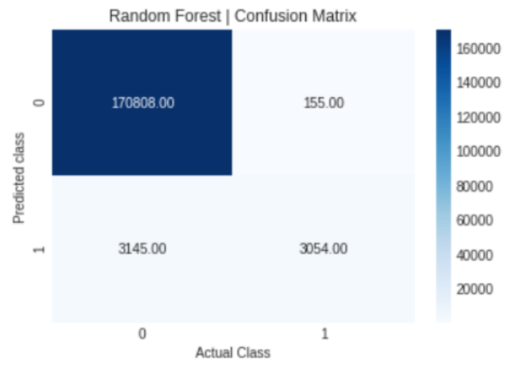
Fig. 13. Improved Confusion Matrix — Linear Regression



```
0.0    170963
1.0      6199
Name: isFraud, dtype: int64
```

Fig. 14. Confusion Matrix — Decision Tree

as compared to decision tree. Below is the confusion matrix produced by random forest tree. It is able to predict 3054 cases of frauds.



```
0.0    170963
1.0      6199
Name: isFraud, dtype: int64
```

Fig. 15. Confusion Matrix — Random Forest

D. LightGBM

”LightGBM is a gradient boosting framework that uses tree based learning algorithms.” It is a leaf-wise growing algorithm which is atypical as most other algorithms grows level-wise. The same can be understood by below figure Fig.16[2] which explain the highlevel implementation of this algorithm.

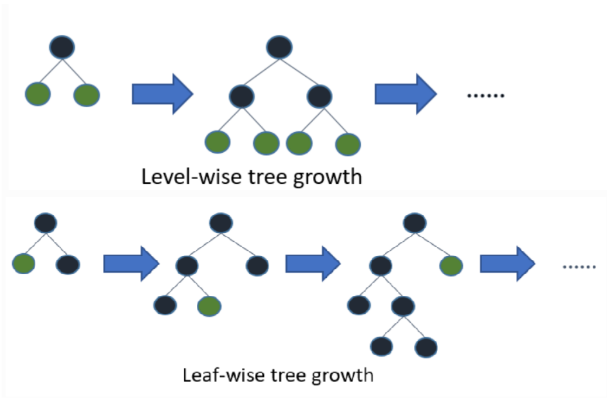


Fig. 16. Other Tree based learners vs LightGBM’s Working

As the dataset, I’m working is large and in fact the term used ”light” in LightGBM is to refer to its high speed working efficiency. It ”takes lower memory to run” and simultaneously handle large scale data. As it supports GPU learning, it has been accepted widely for data science applications.

1) Parameter Tuning:

a) Control Parameters: max-depth: it dictates the maximum depth of the tree and generally used to deal with overfitting for small data. Default value is -1 which means no limit.

feature_fraction: Typically ranges from 0 to 1 and it means the percent fraction that LightGBM will select randomly in each iteration for construction of tree. As, for this work, I have used 0.85 means 85% of parameters will be selected randomly. I have used near to 1 value to speed up the training time.

bagging_fraction: default=1.0 just like feature_fraction, this parameter specifies the fraction of data to be utilized at each iteration. Purpose of this parameter is to fasten up the training time and avoid overfitting.

early_stopping_round: This is really useful parameter as it helps to speed up the analysis. If one metric of one validation data is not able to improve from last iteration then the model will stop training. So, excessive iterations can be avoided.

b) Core parameters: boosting: defines the type of algorithm, for my model I have selected default=gdbt (Gradient boosting)

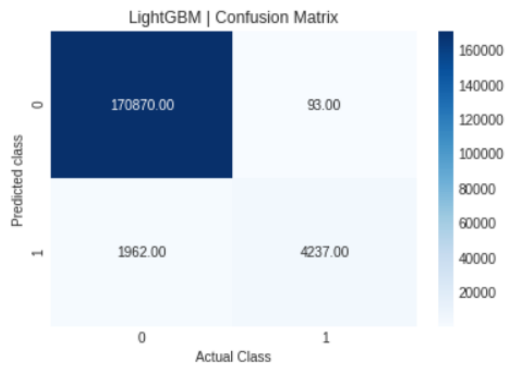
learning_rate: This is a crucial parameter that can affect final outcome. Generally, GBM starts with it as an initial estimate of learning which gets updated using output of each tree. By setting up this parameter we can control the magnitude of this change. Commonly 0.1, 0.01,0.001 are used.

num_leaves: this is total number of leaves in full tree. By default is is set to 31 but as for my work, I set it to 256 as dataset is large with around 240-260 features selected.

objective: used ’binary’ as this is classification problem metric: ’auc’ as we need roc_auc_score.

TABLE III
LIGHTGBM PARAMETERS TABLE

Parameters	Value
n_estimators	1000
learning_rate	0.01
objective	binary
metric	auc
num_threads	-1
num_leaves	256
bagging_fraction	1
feature_fraction	0.85



```
0.0    170963
1.0      6199
Name: isFraud, dtype: int64
```

Fig. 17. Confusion Matrix — LightGBM

2) Results:

In Table IV., it can be seen that LightGBM is clearly outperforming other models. It is able to predict 4237 cases out of total 6199 cases. In Fig. 19., a feature importance barplot is displayed which is generated by lightGBM model. These features can help in tuning more hyper-parameters and feature selection for further improvement.

Further to this, I have used K-Fold cross validation technique to improve this score but it took 4.5 hours of training time. So, for this work, I am not selecting it as best model.

TABLE IV
PERFORMANCE COMPARISON OF MODELS

Model Name	ROC_AUC_SCORE	Training time (in seconds)
Linear Regression	0.54	30
Decision Tree	0.77	77
Random Forest	0.74	308
LightGBM	**0.84**	948
LightGBM (K-Fold cross validation)	0.96	16200

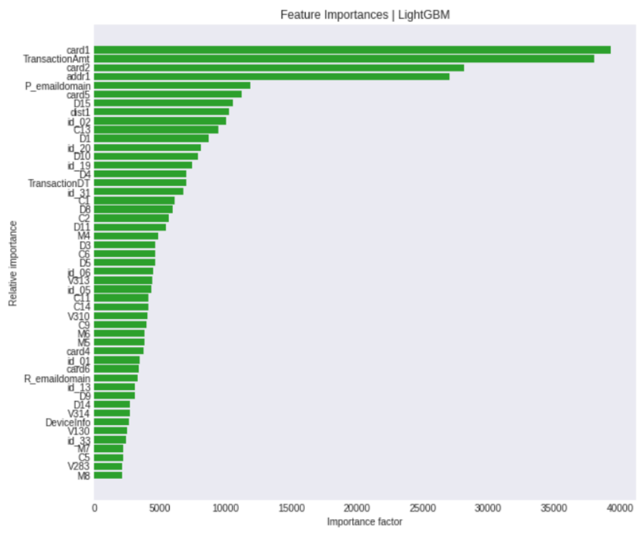


Fig. 18. Relative Feature Importance — LightGBM

VI. CONCLUSION

In my work, I present a machine learning based model to detect consumer transaction fraud from IEEE-CIS Kaggle competition data. Few data mining methods like data cleaning, exploratory data analysis, feature engineering are included in Section II. Further, in Section III, I have discussed model selection including linear regression, decision tree, random forest and LightGBM. After this, results are compared to determine the best model out of all aforementioned models. As per the result, it can be deduced that LightGBM performs better. The last Fig. 19, shows the relative importance of features which is generated by LightGBM model. This insight can be helpful for future improvements in terms of feature selection.

REFERENCES

[1] https://www.kaggle.com/c/ieee-fraud-detection/data

[2] https://lightgbm.readthedocs.io/en/latest/Features.html

[3] Yixuan Zhang, Jialiang Tong, Ziyi Wang, Fengqiang Gao, “Customer Transaction Fraud Detection Using Xgboost Model”, 2020 International Conference on Computer Engineering and Application (ICCEA)

[4] Imane Sadgali, Faouzia Benabbou, “Detection of credit card fraud: State of art”, “International journal of computer network and information security – November 2018”

[5] https://medium.com/@pushkarmandot/https-medium-com-pushkarmandot-what-is-lightgbm-how-to-implement-it-how-to-fine-tune-the-parameters-60347819b7fc