DATA 606: Statistical Methods in Data Science

Credit Card Transaction Fraud Prediction

Final Report

Prepared by

Group 2

Members:

Yi-Chi Chang (30156294)
Yat Hei Fung (30183611)
Aung Khant Min (30160164)
Peerada Tarasub (30178664)

Table of Contents

1. INTRODUCTION 4

2. METHODOLOGY 5

2.1 DATA 5
2.1.1 Variable Explanation 5
2.1.2 Exploratory Data Analysis 6
2.2 MODELING PLAN 8
2.2.1 Assumption Checking 8
2.2.2 Model Selection and Fit 8
2.2.3 Model Validation 9

3. RESULTS 10

3.1 ASSUMPTION CHECKING 10
3.1.1 Multicollinearity 10
3.1.2 Normality 10
3.1.3 Homoscedasticity 11
3.1.4 Linearity 12
3.1.5 Outliers 13
3.2 GUIDING QUESTION 1: PREDICT POTENTIAL FRAUD TRANSACTIONS THAT TEND TO OCCUR IN THE FUTURE 13
3.2.1 Model Fit 13
3.2.2 Model Validation 21
3.3 GUIDING QUESTION 2: CLASSIFY FRAUD TRANSACTIONS BASED ON CATEGORIES 25
3.2.1 Model Fit 25
3.2.2 Model Validation 30

4. CONCLUSION 31

4.1 GUIDING QUESTION 1: PREDICT POTENTIAL FRAUD TRANSACTIONS THAT TEND TO OCCUR IN THE FUTURE 31
4.2 GUIDING QUESTION 2: CLASSIFY FRAUD TRANSACTIONS BASED ON CATEGORIES 31

5. REFERENCES 32

6. APPENDIX 33

6.1 R CODE: ASSUMPTION CHECKING 33
6.2 R CODE: GUIDING QUESTION 1 34
6.2.1 Logistic Regression 34
6.2.2 QDA 38
6.2.3 Classification Tree 40
6.1.4 Model Validation 43
6.3 R CODE: GUIDING QUESTION 2 45
6.3.1 LDA 46
6.3.2 QDA 46
6.3.3 Classification Tree 46
6.3.4 Model Validation 47

1. INTRODUCTION

In a digital world where credit card transactions occur more than cash transactions, it is more than essential to validate them. Credit card fraud is a major problem to obtain illegal sources of funds in transactions, such as buying products without paying or gaining an unauthorized fund [1]. It raises the price of goods and services. Since the decision of fraud detection is difficult because of changes in customer spending behaviors, artificial intelligence has been widely employed by businesses and governments to greatly reduce and even prevent the economical and reputational repercussions of fraud [2].

Fraud Detection and Prevention can generate more sales with good customers by allowing more frequent and higher-value transactions, reducing reserve account requirements based on their good track records, hence improving customer satisfaction, and protecting reputation. because there are two main groups of risk, good and bad credit risk. The former one would appear when rejecting requests of a capable applicant who can repay the loan; whereas the latter one would appear when approving requests of an incompetent applicant who would fail to repay the loan. In terms of economic and social well-being, banks provide credit to multiple divisions to maintain their sustainability, such as manufacturing and business. Failure of a bank would result in turbulence for the economy as well as the social fabric of a country [3].

Our goal is to build a regression model that correctly identifies fraud based on the available data. This model could potentially serve as a safety net for credit card transactions if every new transaction must get validated by this model prior to approving the transaction. With this model acting as a safety net, we will be taking a small stride towards a fraud-free tomorrow.

The following guiding questions are constructed to meet our objective of the analysis:

- Guiding Question 1: Predict potential fraud transactions that tend to occur in the future
- Guiding Question 2: Classify fraud transaction based on categories

2. METHODOLOGY

2.1 Data

The *Credit Card Transactions Fraud Detection Dataset* was retrieved from Kaggle, which is licensed following CC0 1.0 Universal (CC0 1.0) Public Domain Dedication, stated that the content can be copied, modified, and distributed all without asking permission even for commercial purposes. The dataset contains 22 columns and is structured in the form of tabular form and CSV format. Since the data is from Kaggle, it has already split into training and testing datasets. There are 1,296,675 rows for training set and 555,719 rows for testing set.

Additional sampling method was applied on the training set. Because the original training set contains very little amount of fraud transaction (0.6% of the total transaction), which might affect the model validation score of fraud detection – that is – there is the possibility that the validation score results are excellent even though the models perform in the opposite due to too little amount of fraud transaction. Thus, the training set was sampled once again by ensuring that the number of fraud transaction accounted for 5% of the total transaction, while the rest is the normal transaction.

2.1.1 Variable Explanation

The original column names, types, and descriptions are shown below in Table 1:

Column Name	Type	Description
trans_date_trans_time	DateTime	Transaction Date, Transaction Time
cc_num	Number	Credit Card Number of Customer
merchant	Text	Name of the Merchant
category	Text	Category of the Merchant
amt	Number	Amount of the Transaction
first	Text	First Name of Credit Card Holder
last	Text	Last Name of Credit Card Holder
gender	Boolean	Gender of the Credit Card Holder
street	Text	Street Address of the Credit Card Holder
city	Text	City of the Credit Card Holder
state	Text	State of the Credit Card Holder
zip	Text	Zip code of the Credit Card Holder
lat	Number	Latitude Location of the Credit Card Holder
long	Number	Longitude Location of the Credit Card Holder
city_pop	Number	Population of Credit Card Holder's City
job	Text	Job of the Credit Card Holder
dob	Date	Date of Birth of the Credit Card Holder
trans_num	Text	Transaction Number
unix_time	Number	Transaction time in unix time
merch_lat	Number	Merchant Latitude Location

Column Name	Type	Description
merch_long	Number	Merchant Longitude Location
is_fraud	Boolean	Fraud Flag

Table 1: Column Names and Description

As the present project attempts to predict the fraud transaction, the response variable will be the *is_fraud* column where the output is 0 or 1 based on whether it is a fraud or not. The main predictors for investigating and predicting fraud transaction consist of four variables from the original dataset, which are:

- **Category (Categorical Variable)**
- **Amount (Numerical Variable)**
- **Gender (Categorical Variable)**
- **State (Categorical Variable)**

And two variables that are newly calculated from two original dataset columns:

- **Transaction Time in a Day (Categorical Variable):** Calculated from *trans_date_trans_time* column by divided the transaction time into two period, which are, "day" for transactions occurred during 6:00-17:59, and "night" for transactions occurred during 18:00-5:59
- **Age (Numerical Variable):** Calculated from *dob* column by subtracting the year in the column from the current year (2023)

2.1.2 Exploratory Data Analysis

Before going into the phase of modelling. The exploratory analysis of the data was conducted beforehand to gain some understanding and insights of each variable on fraud transaction.

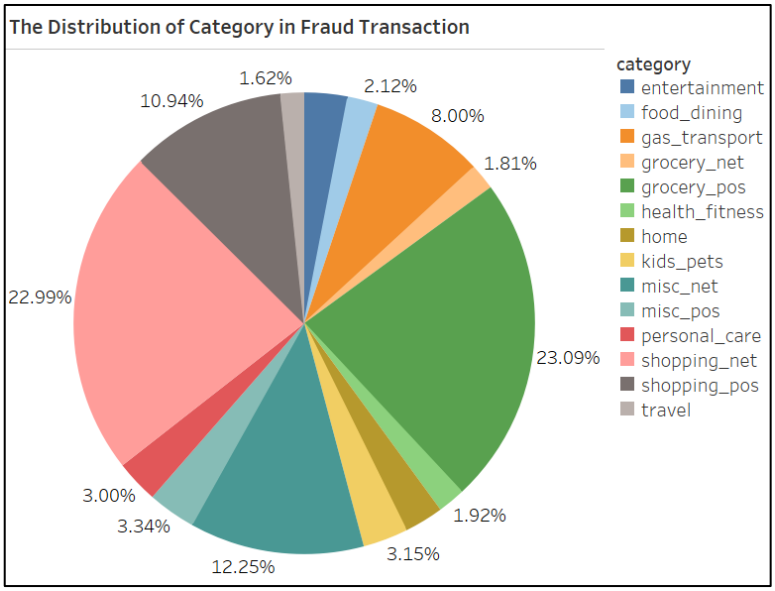

Figure 1: The Distribution of Category in Fraud Transaction

From Figure 1, it can be seen that majority of the fraud transactions are purchased for shopping and grocery's purpose, with the total of 46.08%. In contrast, the least number of fraud transactions fall into travel categories.

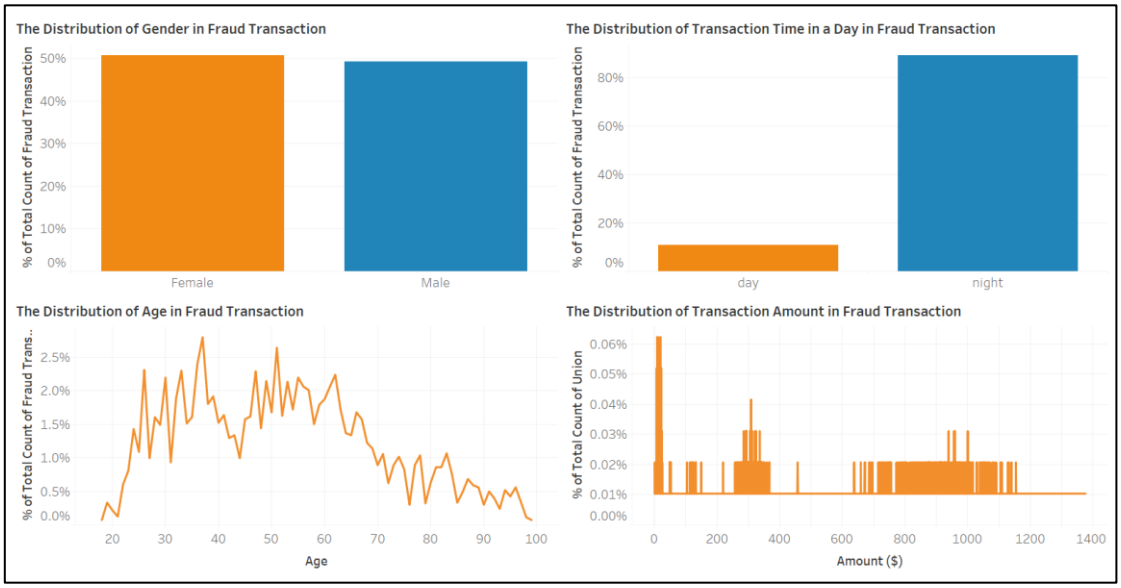

Figure 2: The Distribution of Transaction Amount in Fraud Transaction

Figure 2 shows that fraud transactions slightly occurred from female users more than male users, and most of the users that generate fraud transactions are around 30-60 years of age. Moreover, it

can be clearly seen that majority of the fraud transactions have very little amount, and those transactions mostly occur during the nighttime.

After gaining some understanding of how each predictor related to the fraud transaction, the model construction process would be conducted in the next step.

2.2 Modeling Plan

2.2.1 Assumption Checking

As for the screening procedure and pre-processing data, multicollinearity, normality, homoscedasticity, linearity, and outlines detection will be checked. Multicollinearity is presence if there are linear relationships among each variable combination using multicollinearity test by computing the large variance inflation factor (VIF) values. The factors that correlated with the other factors will be removed from the model construction in the next steps. To check the linearity, normality, homoscedasticity, and outliers of the data, different plots and statistical analysis will be used, details are shown below:

- Normality Assumption – using Histogram and Q-Q Plots, Shapiro-Wilk Normality Test
- Equal Variance Assumption – using Residual VS Fitted Plot and Breusch-Pagan Test
- Linearity Assumption – Residual VS Fitted Plot
- Outliers – using Cook's Distance Plot

2.2.2 Model Selection and Fit

2.2.2.1 Guiding Question 1: Predict potential fraud transactions that tend to occur in the future

For this guiding question, three different models will be fitted and compared to find the best model for the detection of fraud transactions.

1. **Logistic regression:** Logistic regression is commonly used in the banking industry on fraud detection, default prediction, customer behavior analysis, and collection strategy. It is easy to construct and implement. In this analysis, two different ways are implemented to fit the model: (1) Fit with original value of independent variables and (2) group the independent variables into several bins and fit with weight of evidence instead. We will also transform the probability outcome of the WOE logistic regression into score to demonstrate the implementation of scorecard on fraud detection.

2. **Quadratic discriminant Analysis (QDA):** QDA is one of the common classification algorithms that aims to classify the categorical target variables. The algorithm approximates the Bayes classifier, and the discriminant function provides a quadratic decision boundary. It also allows the data to have different covariance matrix across all classes. Considering that the outcome of this dataset has two classes with significantly different number of transactions, it is appropriate to consider using this technique.

3. **Classification Tree:** Classification Tree, along with Logistic Regression and SVM, is a commonly used method for a lot of classification problems. Since the whole logic process can be seen in a tree format, we can easily interpret why we get the results we get. Unlike LDA, it handles non-linear relationships as well as linear relationships. It is very fast to train and is a scalable method of classification. It is robust to outliers as the outliers are not going to skew the model a certain direction. Although there are so many advantages to this method, it also has its fair share of disadvantages. The first problem is that decision trees are prone to overfitting. In order to prevent that, we will be setting a maximum depth of the tree when we call it and prune the tree. It can also be unstable depending on the dataset. If we change the dataset by a little, it will cause the whole tree to change as the binary splitting could potentially change and the feature importance is changed. Classification trees would not be a good fit for imbalanced datasets, but we fixed this condition by oversampling the amount of fraud in the train dataset. All in all, we believe that Classification trees are a great way to deal with this binary classification problem.

2.2.2.2 Guiding Question 2: Classify fraud transactions based on categories

For this guiding question, two out of three models as mentioned in section 2.2.2.1, which are QDA and classification tree, will be used. There will also be one additional model, LDA, to be involved in the comparison step. LDA, or Linear Discriminant Analysis, is suitable for classifying categorical target variables of more than two classes. These three models will be fitted and compared to find the best model for the categorization of fraud transaction categories. However, the variable *Transaction Time in a Day* is replaced by 24-hours format instead of the 'Day' and 'Night' flags.

2.2.3 Model Validation

For model validation, a train-test split for each algorithm will be performed. Upon training the data with the training dataset, we will be checking the accuracy on the testing dataset. We will also be drawing ROC curves and looking at the area under the curve results. After that, we will take it one step further and do a k-folds cross validation with 10 folds to reduce bias in our models. This method also allows us to use our data efficiently as the amount of data we have is limited. At last, by using the same folds, we were able to pinpoint the method with the lowest error rate and present it as our final solution.

3. RESULTS

3.1 Assumption Checking

3.1.1 Multicollinearity

Firstly, the multicollinearity test was performed to find the predictors with large variance inflation factor (VIF) values. Collinearity was not detected among all the selected variables during the initial test.

```
Call:
imcdiag(mod = Train_Fullmodel, method = "VIF")


 VIF Multicollinearity Diagnostics

            VIF detection
category 1.0124         0
amt      1.0238         0
gender   1.0024         0
state    1.0038         0
age      1.0034         0
trans_p  1.0121         0

NOTE:  VIF Method Failed to detect multicollinearity


0 --> COLLINEARITY is not detected by the test
```

Figure 3: Multicollinearity Test

3.1.2 Normality

Linear Discriminant Analysis (LDA) requires the errors between the observed and predicted values to be normally distributed. To test this assumption, a histogram of residuals and Q-Q plot was plotted below.

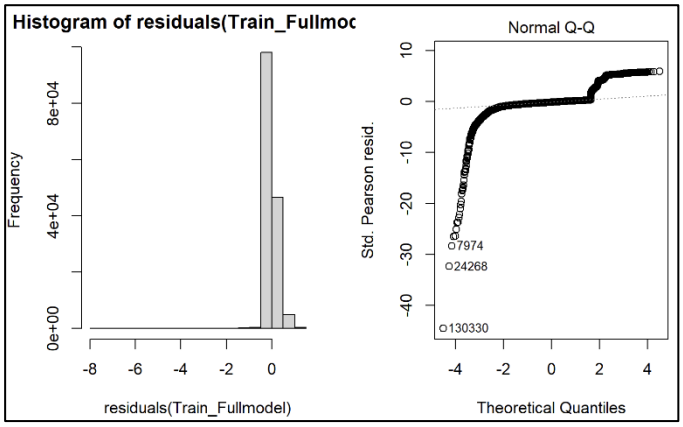

Figure 4: A Histogram of Residuals and Q-Q plot

The model could not meet the normality assumption as there was an outlier observed from the graph. To ensure this, the Shapiro-Wilk test was conducted.

Hypothesis Testing

$$H_0: The\ data\ are\ significantly\ normally\ distributed$$

$$H_A: The\ data\ are\ not\ significantly\ normally\ distributed$$

Results

Set α = 0.05

```
        Shapiro-Wilk normality test

data:  residuals(sam_Train_Fullmodel)
W = 0.55385, p-value < 2.2e-16
```

Figure 5: Shapiro-Wilk Normality Test

The null hypothesis has been rejected in favor of the alternative hypothesis. Shapiro-Wilk normality test also confirmed that the residuals were not normally distributed as the p-value < $2.2e^{-16}$, which was less than α.

3.1.3 Homoscedasticity

Equal variance assumption was then used as next test to see if our data was homoscedastic through a plot of fits to residuals as well as the studentized Breusch-Pagan test.

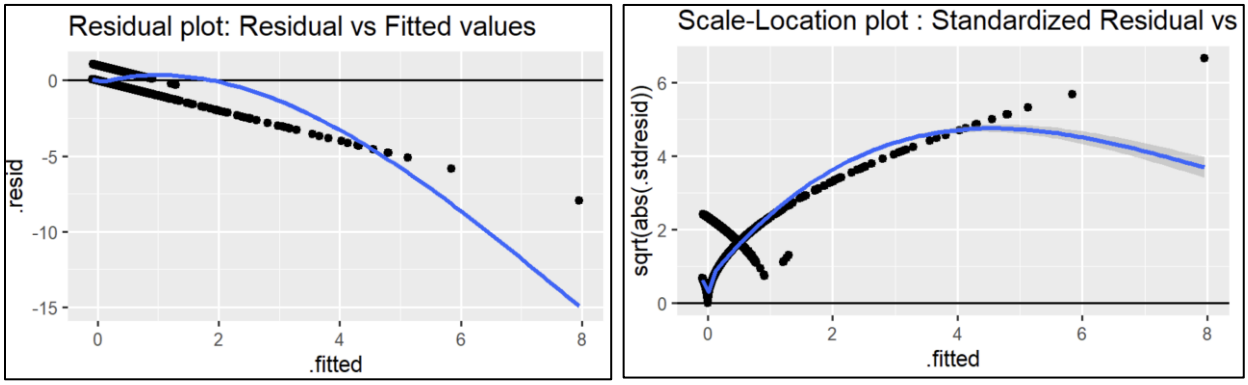

Figure 6: Residual VS Fitted Values Plot and Scale-Location Plot

From both figures above, it seems that the variance was not constant. The Breusch-Pagan test was then be conducted to confirm the homoscedasticity.

Hypothesis Testing

$$H_0: Heteroscedascity\ is\ not\ present$$
$$H_A: Heteroscedascity\ is\ present$$

Results

Set α = 0.05

```
        studentized Breusch-Pagan test

data:  Train_Fullmodel
BP = 50575, df = 67, p-value < 2.2e-16
```

Figure 7: Breusch-Pagan Test

The Breusch-Pagan test result showed that BP = 50575 with p-value of < 2.2e-16 which was less than 0.05, indicating the null hypothesis should be rejected. Thus, it can be concluded that the variance was not constant, or that the heteroscedasticity was present.

3.1.4 Linearity

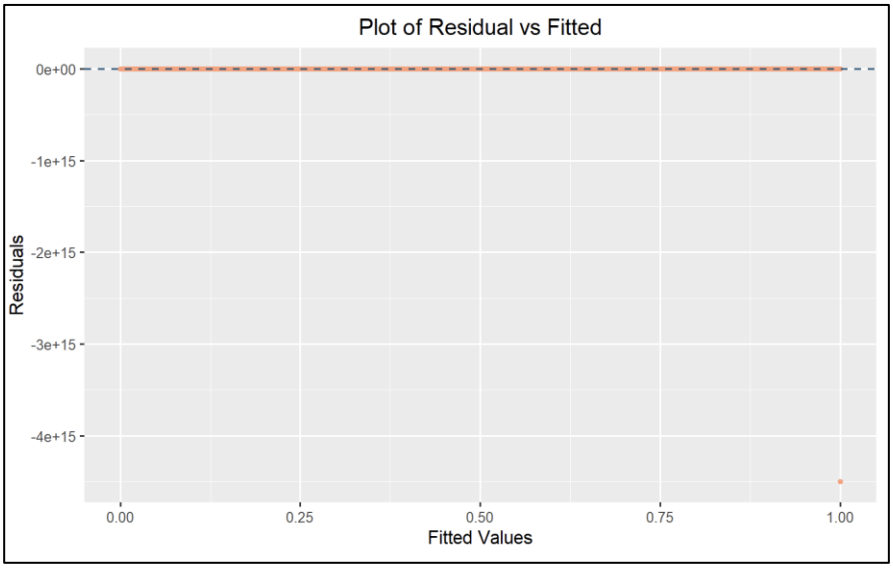

Figure 8: Residual VS Fitted Values Plot

To satisfy logistics regression assumption, linearity assumption was tested to investigate whether there was a straight relationship between the predictors and logit of response variable or not. Residuals VS fitted graph was used to identifying non-linearity. Based on the above residual plot, it was evident that there were no non-linear associations, and hence it met the linearity assumption.

3.1.5 Outliers

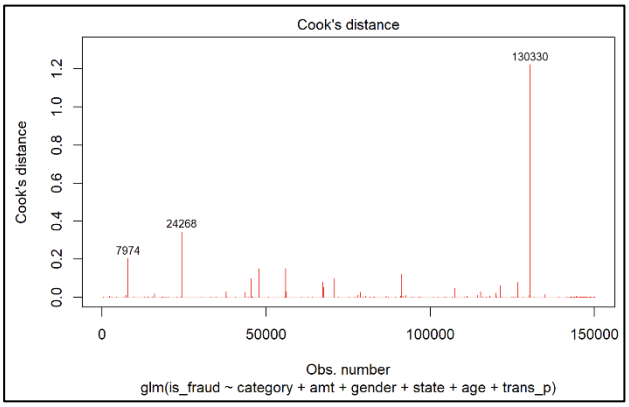

Figure 9: Cook's Distance Plot

Outlier points were then identified and checked as they could have major impact on the accuracy of the model. Cook's Distance plot was plotted for each observation to help indicate the existence of the outlier points. As shown in the graph, the most prominent points that had been identified are 7974, 24268 and 130330. They were not influential towards the model as there are over a hundred thousand of data points in this dataset, and hence they were remained for the further analysis.

3.2 Guiding Question 1: Predict potential fraud transactions that tend to occur in the future

3.2.1 Model Fit

3.2.1.1 Logistic regression

The independent variables used for logistic regression analysis included *category, gender, amount, state, transaction time in a day*, and *age*. As stated in 2.2.2.1, we fitted the logistic regression in two different ways:

(1) Normal Logistic Regression

In this section, we fitted the logistic regression with the original value of the independent variables. As shown in the table below, all independent variables in the logistic regression model were significant under the confidence level of 0.05

Variables	Coefficients	P-value
Category	-0.9737~1.4456	<0.001*** ~ 0.96
Amount (AMT)	0.0069	<0.001***
Gender	0.1212	<0.001***
State	-1.6765~14.9538	<0.001*** ~ 0.85
Age	0.0048	<0.001***
Transaction time in a day (day = 0)	2.164	<0.001***

Table 2: Result of Normal Logistic Regression

We then fitted the model with the testing dataset to see how well the model performed on predicting fraud transactions. As shown in the following confusion matrix of normal and ROC of the normal logistic regression model, the model had a misclassification rate of 0.58% and an AUC of 91.6%.

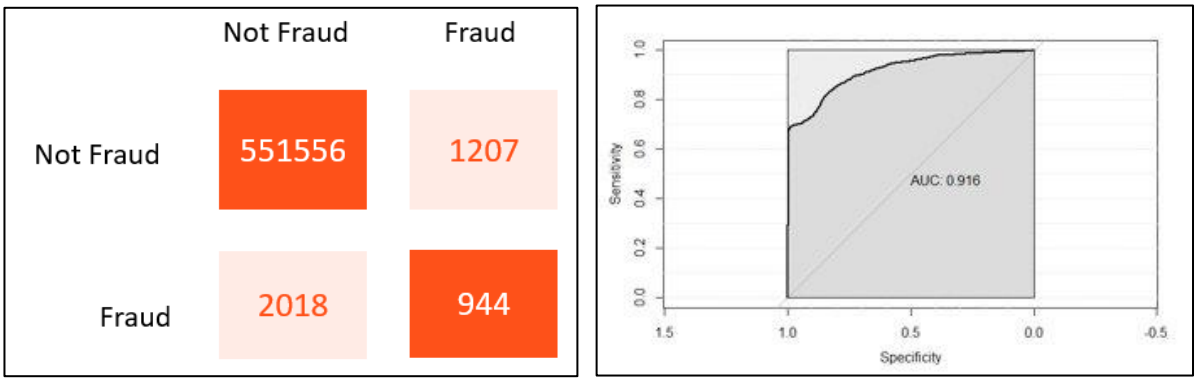

Figure 10: Confusion Matrix and ROC/AUC Curve of Logistic Regression

(2) Weight of Evidence (WOE) Logistic Regression

In this section, we used "scorecard" package in R to generate a fraud detection scorecard. First, all independent variables were grouped into different bins base with the target of maximizing the weight of evidence in each bin. Weight of evidence (WOE) is a statistical measure used in data analysis and predictive modeling to assess the strength of a relationship between an independent variable and a binary dependent variable. WOE is often used in credit scoring and fraud detection applications but can be used in any situation where a binary outcome is being predicted. The WOE is calculated as the natural logarithm of the ratio of two proportions: the proportion of observations with the outcome of interest (e.g., fraud in this analysis) among the observations in a particular category of the independent variable, divided by the proportion of observations without the outcome of interest (e.g., non-fraud) among the observations in the same category of the independent variable. The resulting value reflects the degree to which the presence of the independent variable in a particular category increases or decreases the likelihood of the outcome of interest.

$$WOE = Ln(\frac{\%\ of\ Fraud}{\%\ of\ Non-Fraud})$$

After grouping the variables and calculating the WOE, we will be able to calculate Information Value (IV) from WOE for each variable and use it as a criterion for variable selection. Information Value (IV) is a statistical measure used to evaluate the predictive power of a variable in a binary classification problem.

$$IV = \sum (\%\ of\ Non\ Fraud - \%\ of\ Fraud) * WOE$$

The result of information value showed that *amount* and *transaction time in a day* had the highest information, meaning that they provided the most information when detecting the fraud transactions. Practically, variables with information values lower than 0.3 would be removed from the regression. In this analysis, the IVs of *state*, *age*, and *gender* were lower than 0.3. However, in this analysis, to compare with the result of the normal logistic regression, we still kept them in the model.

After grouping and variable selection, the logistic regression was fitted with the WOE calculated from the previous step. Through stepwise model selection method, variable *gender* was removed from the model. The summary of the model is shown in the table below:

Variables	Coefficients	P-value
Category_woe	0.0950	<0.001***
Amount (AMT)_woe	0.9783	<0.001***
Gender_woe	-	-
State_woe	0.7869	<0.001***
Age_woe	0.9761	<0.001***
Transaction time in a day (day = 0)_woe	0.8062	<0.001***

Table 3: Result of WOE Logistic Regression

After fitting the model, by using the scorecard package in R, the probability outcome of logistic regression was transformed to score for easier implementation. By setting the base point (600, in this analysis) and Points to Double the Odds (PDO) (50, in this analysis, meaning the odds will be doubled when the score increases by 50 points), we could easily transform the outcome of the logistic regression into score. The transformation of the model would make it easier to execute for users without any statistical background.

variable	bin	count	Prop	Fraud_Prob	Score	woe
category	health_fitness , home , food_dining , kids_pets , personal_care , entertainment	66,908	45%	1.75%	7	-1.0807606
	travel, grocery_net, misc_pos, gas_transport	33,940	23%	3.29%	3	-0.4351187
	shopping_pos, grocery_pos, misc_net, shopping_net	49,272	33%	10.58%	-6	0.8102793

variable	bin	count	Prop	Prob	Score	woe
Amt	0~5	17,824	12%	0.14%	256	-3.6235828
	5~25	34,627	23%	4.41%	9	-0.1317893
	25~255	88,498	59%	0.40%	181	-2.5673347
	255~	9,171	6%	61.04%	-239	3.3934428

variable	bin	count	Prop	Prob	Score	woe
Age	0~34	27,442	18%	5.22%	-3	0.0457665
	34~54	68,424	46%	3.85%	15	-0.2723414
	54~68	29,002	19%	6.45%	-15	0.2708062
	68~80	14,623	10%	4.96%	0	-0.0074393
	80~	10,629	7%	7.88%	-28	0.4862382

variable	bin	count	Prop	Prob	Score	woe
Trans_P	day	79,335	53%	1.03%	114	-1.6160701
	night	70,785	47%	9.44%	-48	0.6838608

variable	bin	count	Prop	Prob	Score	woe
State	ID , CT , MT , HI , AZ , ND , LA	8,245	5%	3.04%	30	-0.5165546
	NJ , NC , TX , MO , NM , MI , IN , WA , AR , AL , IA	45,672	30%	4.41%	8	-0.1323443
	MS , GA , OK , UT , WI , IL , WY , WV , KY , MA , CA , DC , PA , SD	45,679	30%	4.89%	1	-0.0227925
	VT , MD , MN , FL , NY , SC , OH , VA , KS , NH , NE	41,705	28%	5.70%	-8	0.1392333
	ME , TN , CO , OR , NV , AK , RI , DE	8,819	6%	7.12%	-22	0.3761879

Figure 11: Binning and score transformation of the variables

After the transformation, the binning process at the start of this section was executed again on the outcome score of the training dataset, to separate the transaction into different risk level. In this analysis, transaction scores were separated into eight different risk levels by WOE method, with level 1 having the highest proportion of fraud transactions and level 8 having the lowest.

Can set up a threshold based on the company's strategy

Bin	Score Range	Total Transaction	Fraud	Fraud_rate	Fraud_Accu	Good	Good_Rate	Good_Accu
1	~310	2797	2316	82.80%	38.54%	481	0.42%	17.20%
2	310~450	2770	1814	65.49%	68.73%	956	0.84%	34.51%
3	450~540	3617	536	14.82%	77.65%	3081	2.70%	85.18%
4	540~660	10806	891	8.25%	92.48%	9915	8.67%	91.75%
5	660~730	18862	147	0.78%	94.92%	18715	16.37%	99.22%
6	730~800	31004	266	0.86%	99.35%	30738	26.89%	99.14%
7	800~920	37865	33	0.09%	99.90%	37832	33.10%	99.91%
8	920~	12591	6	0.05%	100.00%	12585	11.01%	99.95%
	Total	**120312**	**6009**	**4.99%**		**114303**	**95.01%**	

Figure 12: Result of the scorecard

After the construction of the scorecard, the testing dataset was fitted, with an AUC of 95.3%, which was higher than 91.6%, the AUC of the normal logistic regression.

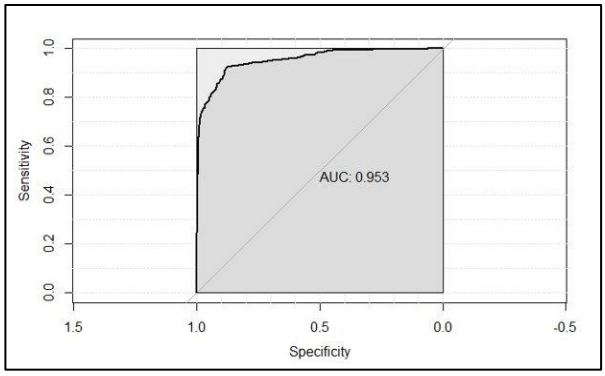

Figure 13: Result of the scorecard

3.2.1.2 Quadratic discriminant Analysis (QDA)

First, all categorical predictors, which were *category, amount, gender, state, age*, and *transaction time in a day*, were converted into numerical values before proceeding with any analysis. This was because some analysis could not cope with categorical variables.

category <chr>
travel
shopping_net
grocery_pos
gas_transport
grocery_pos
shopping_pos

→

category <int>
14
12
5
3
5
13

Figure 14: An example of categorical to numerical values conversion

Then, the model was fitted using the training set. The partition plot was plotted to display the relationship between each predictor.

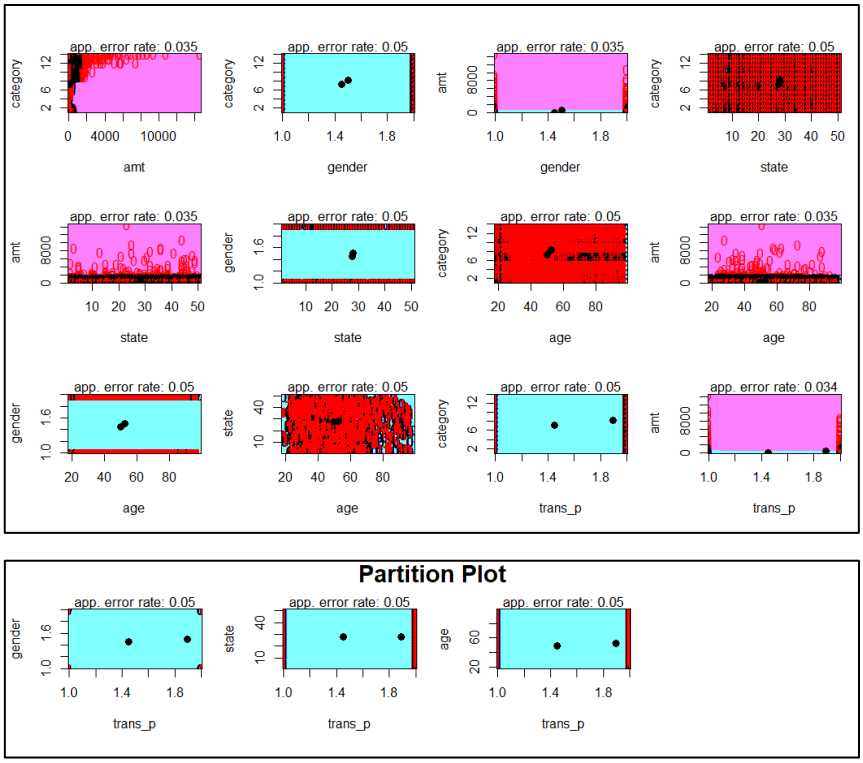

Figure 15: Partition plot of QDA

After obtaining the model, it was used to run with the testing set to evaluate the performance for fraud prediction.

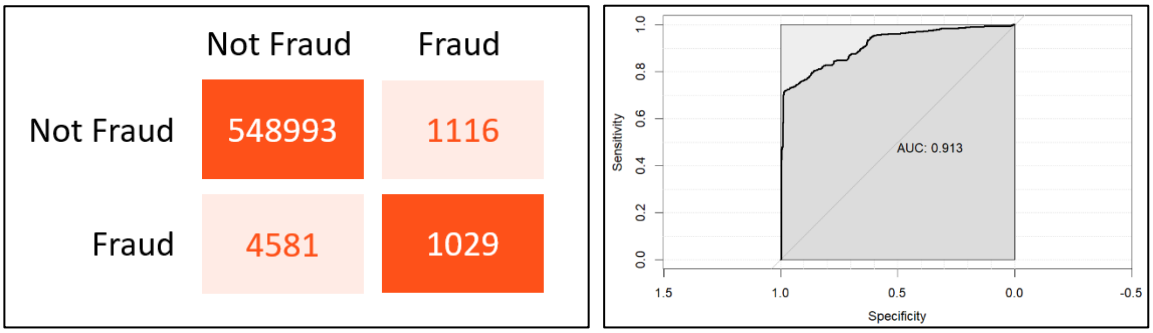

Figure 16: Confusion Matrix and ROC/AUC Curve of QDA

The confusion matrix and ROC curve of the error rate were calculated and resulted in 1.025% of misclassification rate and 91.3% of AUC as displayed in Figure 13.

3.2.1.3 Classification Tree

For this approach, the first thing we did was fit the model using the training dataset that we modified to make sure there is at least 5% of fraud in the dataset. Our unpruned tree is shown below.

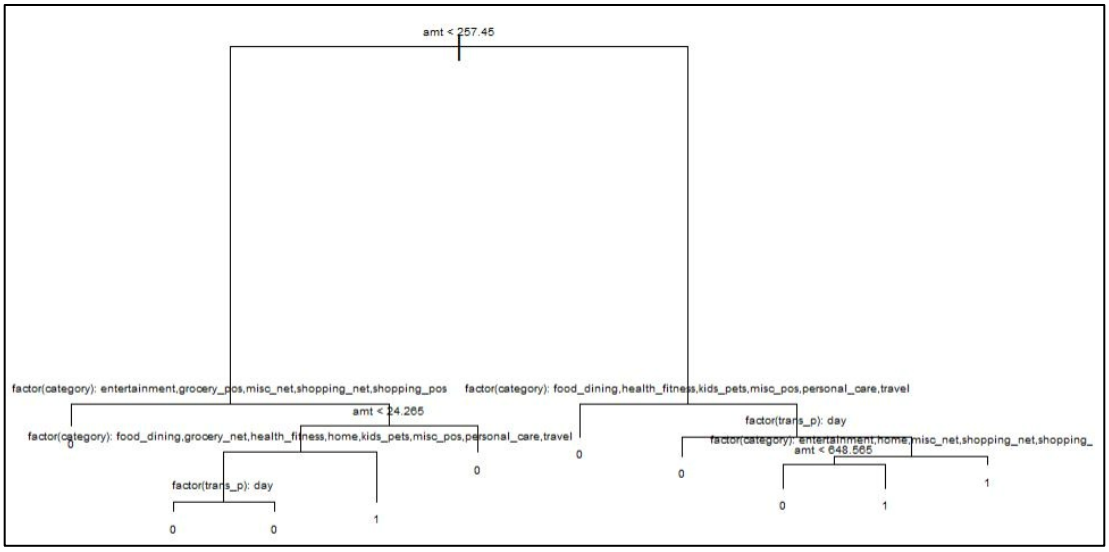

Figure 17: Classification Tree for Fraud Prediction

After getting the classification tree up and running, we checked whether pruning was necessary or not. The plot below shows the elbow curve for pruning. It seemed that the best number of nodes was 9. Since the number of nodes on the unpruned tree was already 9, we decided to keep the model we had.

Figure 18: Elbow curve for Pruning

The misclassification rate of the tree turned out to be 0.003044704. 1014 incidents were predicted no fraud while it was a fraud (Type II) and 678 incidents were predicted fraud while it was not a fraud (Type I). The variables that were mainly used are *amount, category*, and *transaction time in a day*. You can see from the tree that amount was the most important variable as it was in the root node of the tree. After that category would be the second variable of importance. Lastly, transaction time in a day was the third.

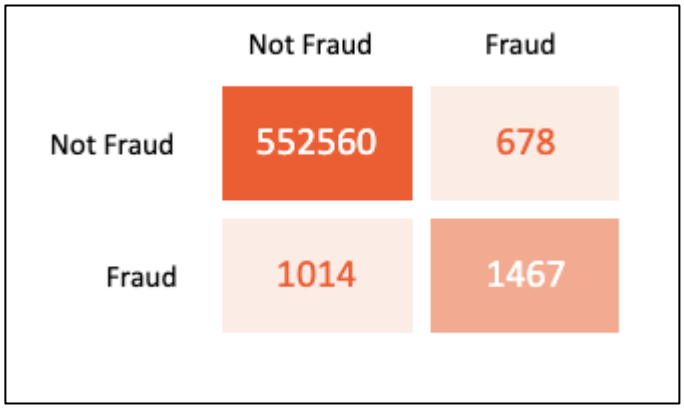

Figure 19: Confusion Matrix for Classification Tree

Since the model has 1014 Type II error, we ran random forest algorithm on the training set with ntree = 500. After the random forest had been generated, the best tree was picked out to examine the model. The misclassification rate was 0.00302311. It happened to be slightly lower than that of the classification tree we had previously. The amount of Type II error increased to 1252 but Type I error reduced to 428. Since Type II error was a lot more important compared to Type I error, we decided to discard the best tree from the random forest model.

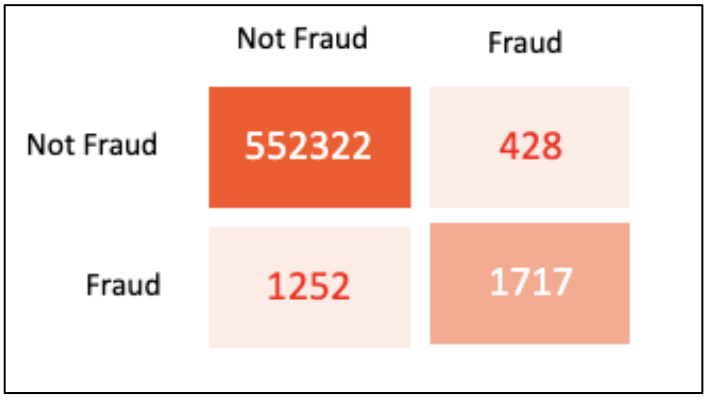

Figure 20: Confusion Matrix for the Best Tree from Random Forest

Although we could not use the best tree from the random forest model, we were able to draw insights regarding the weights of the variable. Since the random forest model uses greedy algorithm for each node, we can check which variables has the highest impact and which has the lowest. For the feature engineering of the random forest, we investigated a measure called mean decrease Gini. The mean decrease in Gini coefficient is a way to measure how each variable contributes to the homogeneity of the nodes. It mainly measures the importance of a variable in random forest model. The higher the value, the more important the variable in the model. From our model, it seemed that *amount* had the highest importance with a mean decrease in Gini at 8709. In second place was *category* with 2004. The third place was *transaction time in a day* with 584. At the last place we had *gender* with 87 mean decreases in Gini. It showed gender was not significant in this fraud dataset.

Feature	Mean Decrease Gini (Impurity)
Category	2004.4661
Amount	8709.7266
Gender	87.5816
State	327.3736
Age	460.1737
Transaction Period	584.6260

Table 4: Feature Importance from Random Forest

3.2.2 Model Validation

3.2.2.1 Misclassification Rate

The first thing we did for model validation was checking the misclassification rate of each model with k-folds misclassification rate. We used 10 folds and applied it to all three models. The plot below shows that Logistic Regression had the highest amount of misclassification rate with 0.0125 and classification tree had the lowest misclassification rate with 0.0011. After plotting out all the k-fold results, we noticed that misclassification rates of QDA and Logistic Regression across the folds seemed to have higher variance compared to that of Classification Tree.

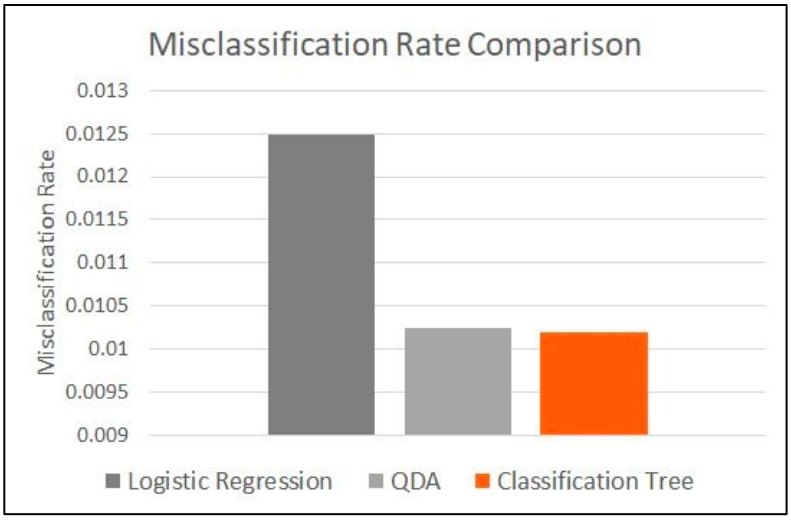

Figure 21: Average Error Rate of 10 Folds

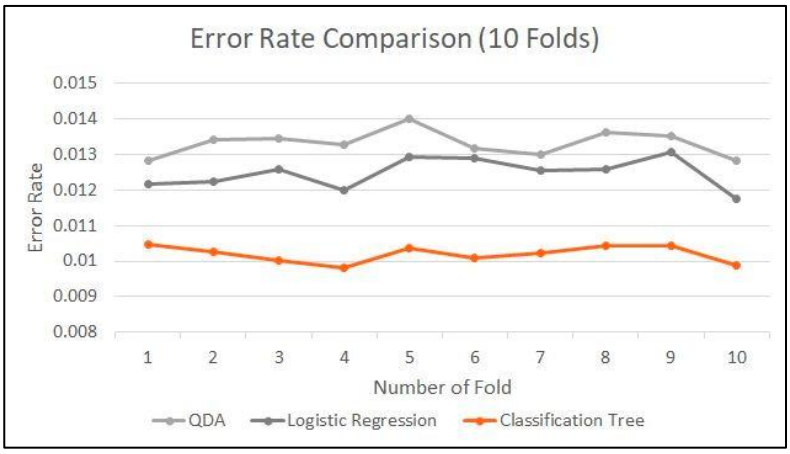

Figure 22: Total Error Rate Across 10 Folds

3.2.2.2 Type II Error Rate

Although total error was a good measurement for checking a model's validity, for our credit fraud problem, type II error serves more of a threat compared to type I error. If a non-fraud transaction gets misclassified as a fraud transaction, the customer will be notified, and the customer will ensure that he or she is the one making the transaction. After that, there are no more complications. If a fraudulent transaction gets misclassified as a non-fraud transaction, the customer will lose money affecting the trust between the bank and the customer. As a result, we decided to repeat the k-fold procedure and this time, we only looked at the Type II error rate. The results astounded us. Logistic Regression turned out to have the lowest Type II error rate at 0.0013799. The decision tree came out second with 0.00264 misclassification rate and QDA came in third with 0.006 misclassification rate. When we looked at the Type II error rate across each fold individually, Logistic Regression

performed well across all 10 folds with minimal variation compared to the classification tree and QDA.

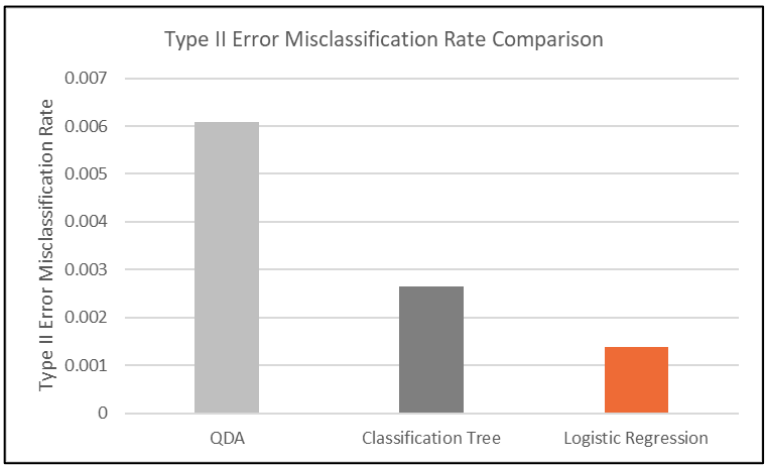

Figure 23: Average Type II Error Rate

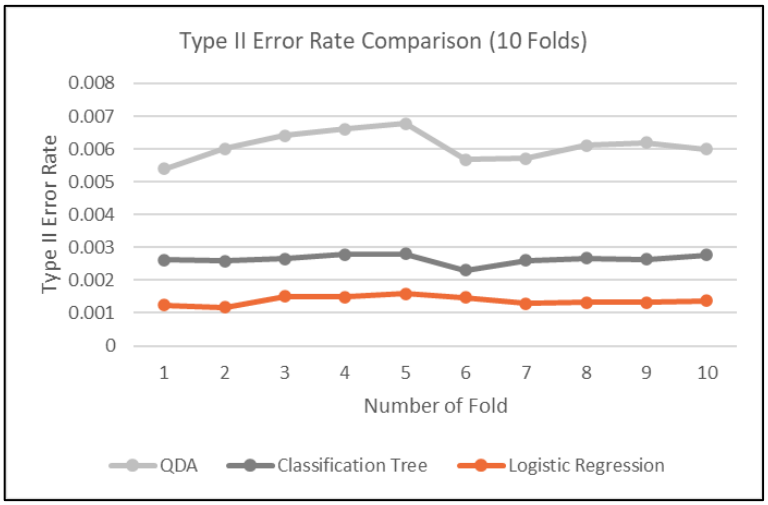

Figure 24: Type II Error Rate across the folds

3.2.2.3 ROC Curves

Lastly, we plotted out ROC curves for all the models and compared the area under the curve. From the plots below, we can see that the logistic regression method and classification tree had a close area under the curve value. The area under the curve for Logistic Regression turned out to be 0.953 and the area under the curve for Classification Tree was 0.968. For QDA, the area under the curve was 0.913, lowest among the three by a margin. All in all, we believed that logistic regression method was the best performing method among the three.

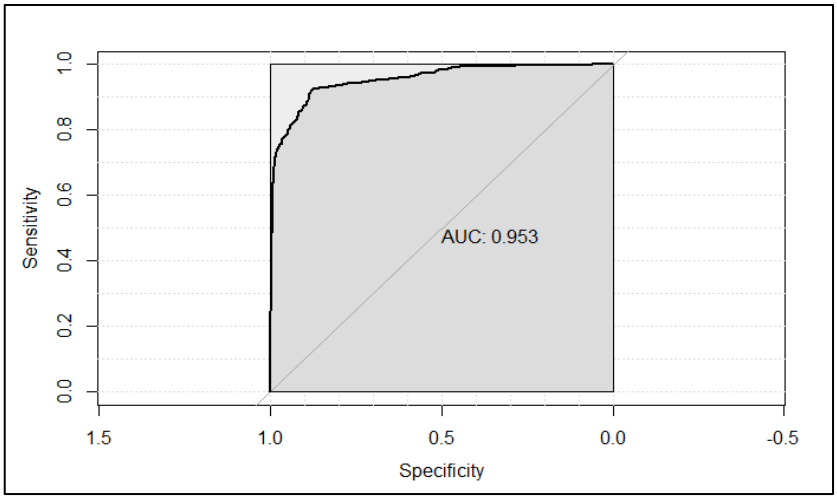

Figure 25: ROC curve for Logistic Regression- Scorecard

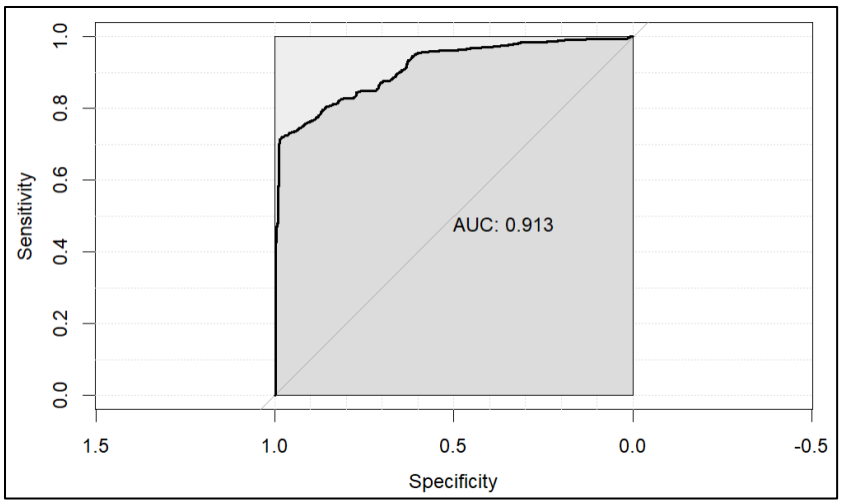

Figure 26: ROC curve for QDA

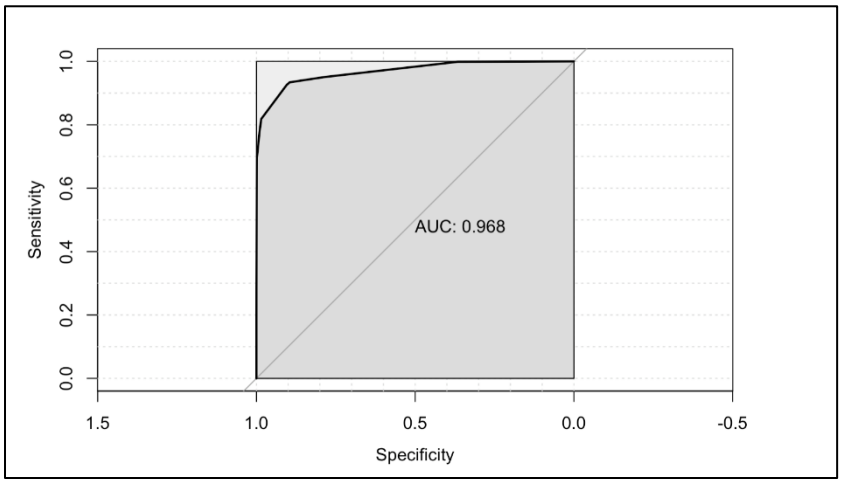

Figure 27: ROC curve for Classification Tree

3.3 Guiding Question 2: Classify fraud transactions based on categories

3.2.1 Model Fit

3.2.1.1 Linear discriminant Analysis (LDA)

Although the data did not pass the normality and homoscedasticity assumption test, LDA was still be performed to compare the misclassification with other models (QDA and classification tree).

	entertainment	food_dining	gas_transport	grocery_net	grocery_pos	health_fitness	home	kids_pets	misc_net	misc_pos	personal_care
entertainment	38	0	0	0	1	0	2	0	3	0	0
food_dining	0	8	0	0	1	2	3	2	0	2	7
gas_transport	0	9	177	44	0	3	0	5	0	18	2
grocery_net	0	0	6	0	0	0	0	0	0	1	0
grocery_pos	36	5	0	0	522	0	7	0	0	0	0
health_fitness	0	0	0	2	0	4	0	15	0	1	9
home	4	4	0	0	21	0	45	1	0	0	0
kids_pets	0	9	2	0	0	24	0	26	0	17	30
misc_net	4	0	0	0	0	0	0	0	176	15	0
misc_pos	0	3	0	0	9	0	5	0	0	0	0
personal_care	0	4	0	0	0	9	0	20	0	8	11
shopping_net	0	0	0	0	0	0	0	0	32	2	0
shopping_pos	0	0	0	0	0	0	0	0	63	4	0
travel	0	1	1	1	0	3	0	4	0	7	12

	shopping_net	shopping_pos	travel
entertainment	0	0	0
food_dining	0	0	1
gas_transport	0	0	3
grocery_net	0	0	0
grocery_pos	0	0	0
health_fitness	0	0	7
home	0	0	0
kids_pets	0	0	16
misc_net	36	85	0
misc_pos	0	0	0
personal_care	0	0	9
shopping_net	421	93	0
shopping_pos	60	51	0
travel	0	0	5

Figure 28: Confusion matrix corresponding to the LDA classifier results of category

After fitting the model with LDA, the accuracy of the testing set was 0.6733362 (misclassification rate of 0.3266638. Some relatively significant misclassifications were shown in the table above. For example, all 'grocery_net' and 'misc_pos' were misclassified as other categories, most 'health_fitness' and 'personal_care' were misclassified as 'kid_pets', most 'shopping_pos' and 'travel' were also misclassified as other categories.

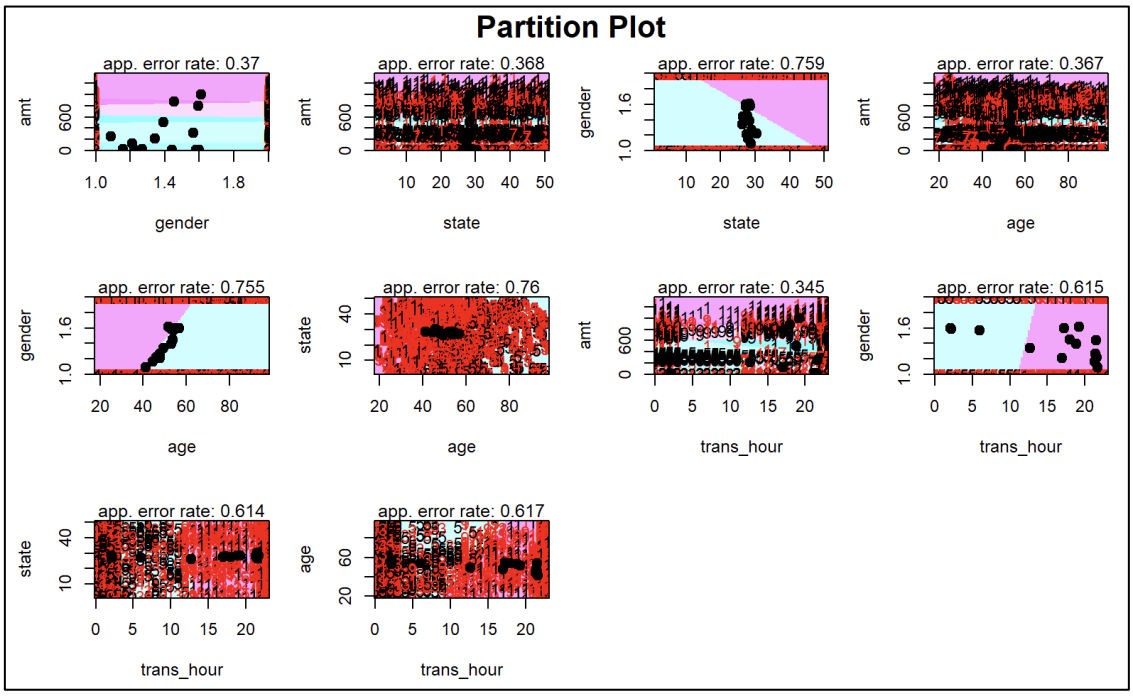

Figure 29: Partition Plot of LDA

The partition plot above shows classification for each of observation in the training dataset based on the Linear Discriminant Analysis Model, and for every combination of two variables. Most graphs had large amount of overlap.

3.2.1.2 Quadratic discriminant Analysis (QDA)

	1	2	3	4	5	6	7	8	9	10	11	12	13	14
entertainment	51	0	0	0	0	0	3	0	1	0	0	0	2	0
food_dining	0	41	0	0	0	0	0	0	0	0	0	0	0	0
gas_transport	0	0	183	44	0	0	0	0	0	12	0	0	0	0
grocery_net	0	0	0	0	0	0	0	0	0	0	0	0	0	0
grocery_pos	4	0	0	0	550	0	8	0	0	0	0	0	0	0
health_fitness	0	0	0	0	0	27	0	34	0	0	28	0	0	0
home	0	0	0	0	4	0	49	0	0	0	0	0	0	0
kids_pets	0	0	0	0	0	14	0	23	0	0	21	0	0	1
misc_net	2	0	0	0	0	0	0	0	209	11	0	43	98	0
misc_pos	25	2	3	3	0	1	2	4	2	1	2	0	2	3
personal_care	0	0	0	0	0	3	0	4	0	12	20	0	0	2
shopping_net	0	0	0	0	0	0	0	0	42	3	0	457	106	0
shopping_pos	0	0	0	0	0	0	0	0	20	6	0	17	21	0
travel	0	0	0	0	0	0	0	8	0	30	0	0	0	35

Figure 30: Confusion matrix corresponding to the QDA classifier results of category

After fitting the model with QDA, the accuracy of the testing set was 0.7250979 (misclassification rate of 0.2749021). Some relatively significant misclassifications were shown in the table above, but it had less misclassification than LDA. All 'grocery_net' and 'misc_pos' were still misclassified as other categories (mainly 'gas_transport' and 'travel' respectively), as well as the 'shopping_pos'.

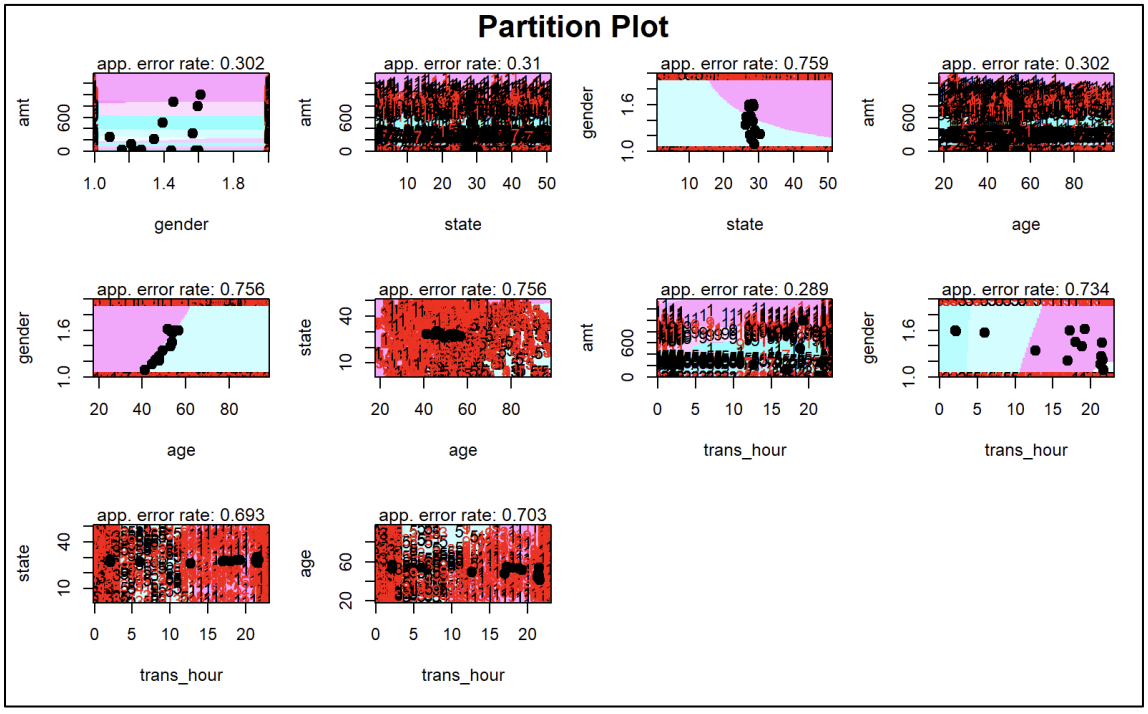

Figure 31: Partition Plot of QDA

The partition plot above shows classification for each of observation in the training dataset based on the Quadratic Discriminant Analysis Model, and for every combination of two variables. Same as the those of LDA, most graphs had large amount of overlap, but the different colored regions were slightly different.

3.2.1.3 Classification Tree

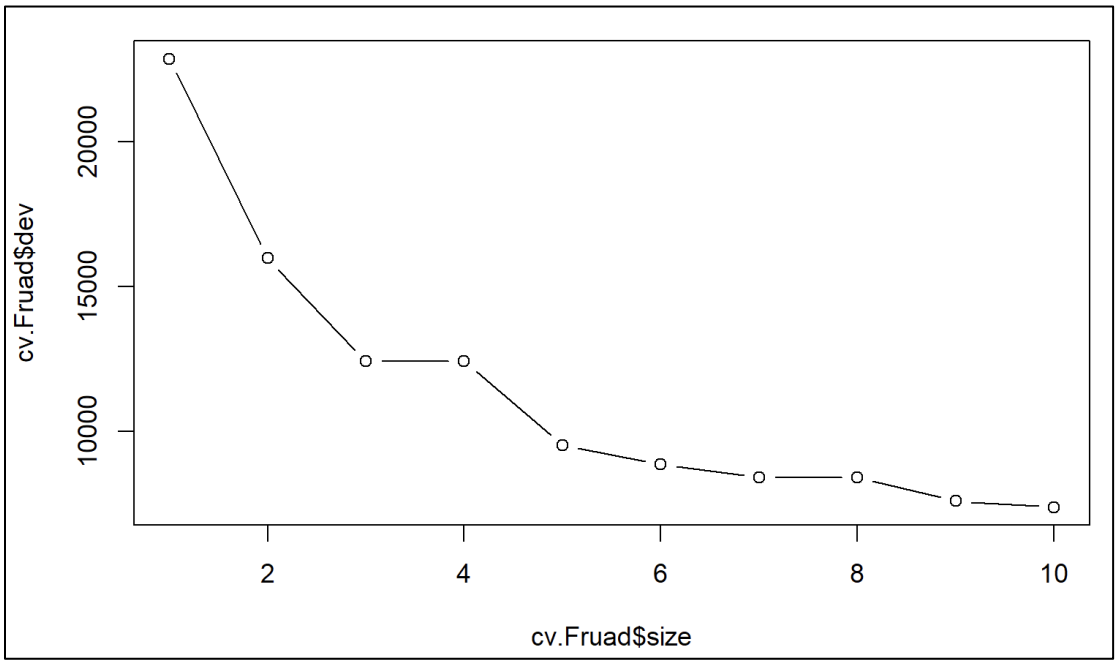

Figure 32: Elbow curve for pruning classification tree of fraud transaction type

When using cross-validation, terminal nodes of 10 had the lowest classification error rate, indicating no tree pruning was required.

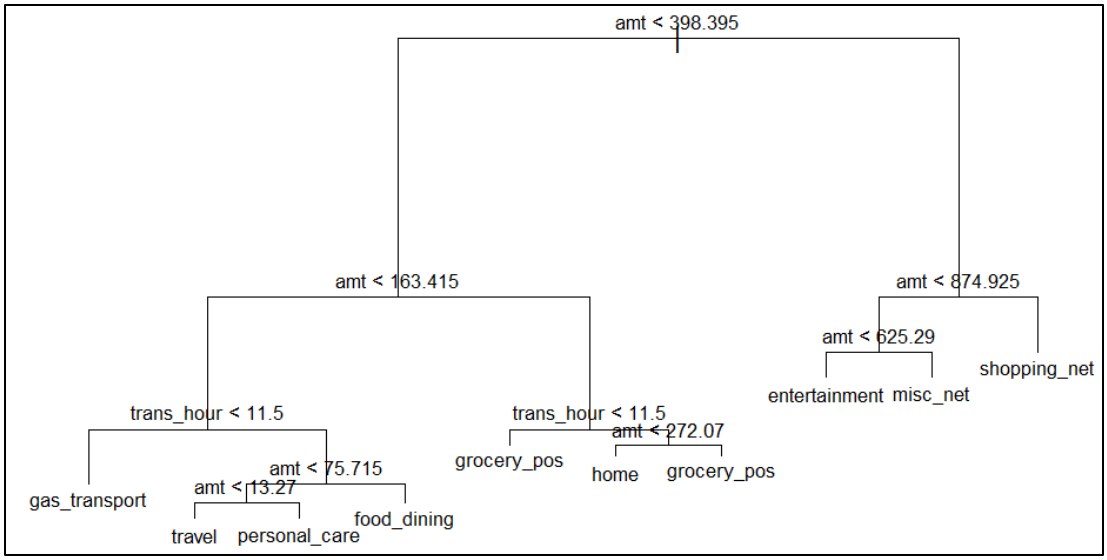

Figure 33: Selected classification tree of fraud transaction type

The graph above shows how the tree performed classification, in which most intermediate nodes were depended on the amount of transaction.

Fruad.tree.pred	entertainment	food_dining	gas_transport	grocery_net	grocery_pos	health_fitness	home	kids_pets	misc_net	misc_pos
entertainment	73	0	0	0	0	0	2	0	2	0
food_dining	0	32	0	0	0	0	0	0	0	0
gas_transport	0	11	186	47	0	0	0	0	0	12
grocery_net	0	0	0	0	0	0	0	0	0	0
grocery_pos	7	0	0	0	550	0	14	0	0	0
health_fitness	0	0	0	0	0	0	0	0	0	0
home	0	0	0	0	4	0	46	0	0	0
kids_pets	0	0	0	0	0	0	0	0	0	0
misc_net	2	0	0	0	0	0	0	0	223	16
misc_pos	0	0	0	0	0	0	0	0	0	0
personal_care	0	0	0	0	0	45	0	64	0	11
shopping_net	0	0	0	0	0	0	0	0	49	5
shopping_pos	0	0	0	0	0	0	0	0	0	0
travel	0	0	0	0	0	0	0	9	0	41

Fruad.tree.pred	personal_care	shopping_net	shopping_pos	travel
entertainment	0	0	1	0
food_dining	0	0	0	0
gas_transport	0	0	0	0
grocery_net	0	0	0	0
grocery_pos	0	0	0	0
health_fitness	0	0	0	0
home	0	0	0	0
kids_pets	0	0	0	0
misc_net	0	55	119	0
misc_pos	0	0	0	0
personal_care	71	0	0	0
shopping_net	0	462	109	0
shopping_pos	0	0	40	0
travel	0	0	0	41

Figure 34: Confusion matrix corresponding to the tree classifier results of category

After fitting the model with the classification tree, the accuracy of the testing set was 0.7324924 (misclassification rate of 0.2675076). Like the previous models, 'grocery_net', 'health_fitness' and 'shopping_net' had very high misclassification rate, and it performed worse in classifying 'kids_pets'.

3.2.2 Model Validation

Since the dependent variable was categorical that had multiple flags, ROC curves were unable to be plotted for all the models and compared the area under the curve. Alternatively, multi-class ROC analysis, which is a kind of multi-objective optimization, was implemented to evaluate the performance. This function performs multiclass AUC defined as a mean of several AUC and cannot be plotted. The multi-class area under the curve for LDA, QDA, and classification tree were 0.8904, 0.946, 0.9376 respectively.

Classification	Accuracy	Misclassification	AUC
LDA	0.645498	0.354502	0.8904
QDA	0.7250979	0.2749021	0.946
Classification Tree	0.7324924	0.2675076	0.9376

Table 5: Model Validation Score

Comparing the accuracy (misclassification) of three models, classification tree had the highest accuracy while LDA had the lowest, but QDA had the highest AUC. It might be because the calibration (threshold) of classification tree was not chosen correctly. Nevertheless, classification tree should be chosen, and random forest might be able to improve the accuracy of classification tree by getting the best tree.

4. CONCLUSION

4.1 Guiding Question 1: Predict potential fraud transactions that tend to occur in the future

In conclusion, after conducting analysis on three different machine learning algorithms, it can be concluded that ***Logistic Regression*** is the best method for predicting fraudulent transactions. Classification tree and QDA also performed adequately, but they are not as good as Logistic Regression for classifying fraudulent transactions. Logistic regression did not show the best performance in terms of total accuracy on the evaluation metrics, but it had the least amount of type II error all three methods. For our dataset, reducing Type II error is the main priority. As a result, we recommend Logistic Regression for predicting fraudulent transactions due to its superior performance for reducing Type II error and ability to handle imbalanced datasets.

4.2 Guiding Question 2: Classify fraud transactions based on categories

In conclusion, after conducting analysis on three different machine learning algorithms, it can be concluded that ***Classification Tree*** is the best method for predicting the category of fraudulent transactions. Nevertheless, QDA also performed adequately, the choice of using which model should depend on the situation.

5. REFERENCES

- Carcillo, F., Le Borgne, Y. A., Caelen, O., Kessaci, Y., Oblé, F., & Bontempi, G. (2021). Combining unsupervised and supervised learning in credit card fraud detection. *Information sciences*, *557*, 317-331.
- Caruso, G., Gattone, S. A., Fortuna, F., & Di Battista, T. (2021). Cluster Analysis for mixed data: An application to credit risk evaluation. *Socio-Economic Planning Sciences*, *73*, 100850.
- D. Bhalla, "A complete guide to Random Forest in R," *ListenData*. [Online]. Available: https://www.listendata.com/2014/11/random-forest-with-r.html. [Accessed: 17-Feb-2023].
- Finnstats, "Random Forest in R: R-bloggers," *R*, 13-Apr-2021. [Online]. Available: https://www.r-bloggers.com/2021/04/random-forest-in-r/. [Accessed: 17-Feb-2023].
- Shirgave, S., Awati, C., More, R., & Patil, S. (2019). A review on credit card fraud detection using machine learning. *International Journal of Scientific & technology research*, *8*(10), 1217-1220.
- Credit Card Transactions Fraud Detection Dataset. https://www.kaggle.com/datasets/kartik2112/fraud-detection. Accessed 18 Feb. 2023.

6. APPENDIX

6.1 R Code: Assumption Checking

```
# Read dataset and check dimension
```{r}
train = read.csv('sampled_train.csv')
train
dim(train)
```

Multicollinearity check
```{r}
library(mctest)
Train_Fullmodel <- glm(is_fraud ~ category+amt+gender+state+age+trans_p,
data=train)
imcdiag(Train_Fullmodel,method="VIF")
```

Linearity / Independence Assumption check for linear model
```{r}
library(ggplot2)
predicted = Train_Fullmodel$fitted.values
e = Train_Fullmodel$residuals
df2 = data.frame(predicted, e)
df2

ggplot(df2, aes(x = predicted, y = e)) +  geom_point(size=1, col='lightsalmon',
position="jitter") + xlab("Fitted Values") + ylab("Residuals") + ggtitle("Plot of
Residual vs Fitted") + geom_hline(yintercept=0, color="#22577a",
linetype="dashed") + theme(plot.title = element_text(hjust = 0.5))
```

Linearity check for logit model
```{r}
Train_logitmodel = glm(formula = is_fraud ~ category+amt+gender+state+age+trans_p,
data=train, family = binomial(link = "logit"))

library(ggplot2)
predicted3 = Train_logitmodel$fitted.values
e3 = Train_logitmodel$residuals
df3 = data.frame(predicted3, e3)
df3

ggplot(df3, aes(x = predicted3, y = e3)) +  geom_point(size=1, col='lightsalmon',
position="jitter") + xlab("Fitted Values") + ylab("Residuals") + ggtitle("Plot of
Residual vs Fitted") + geom_hline(yintercept=0, color="#22577a",
linetype="dashed") + theme(plot.title = element_text(hjust = 0.5))
```

3.7 Equal Variance Assumption
1. Residuals and a scale location plot
```{r}
library(ggplot2)
#residuals plot
ggplot(Train_Fullmodel, aes(x=.fitted, y=.resid)) +
  geom_point() +
  geom_hline(yintercept = 0) +
  geom_smooth()+
  ggtitle("Residual plot: Residual vs Fitted values")
```
```

```
#a scale location plot
ggplot(Train_Fullmodel, aes(x=.fitted, y=sqrt(abs(.stdresid)))) +
  geom_point() +
  geom_hline(yintercept = 0) +
  geom_smooth()+
   ggtitle("Scale-Location plot : Standardized Residual vs Fitted values")

#optional graphs for residual plots and a scale location plot
plot(Train_Fullmodel, which=1) #residuals plot
plot(Train_Fullmodel, which=3) #a scale location plot
```

```
# Breusch-Pagan Test
```{r}
library(lmtest)
bptest(Train_Fullmodel)
```

```
# 3.8 Normality Assumption and Shapiro-Wilk test
1. Hypothesis
H0: The data are significantly normally distributed.
Ha: The data are not significantly normally distributed.

2. R code
```{r}
library(ggplot2)
#option 1 (histogram)
qplot(residuals(Train_Fullmodel),
      geom="histogram",
      binwidth = 0.05,
      main = "Histogram of residuals",
      xlab = "residuals", color="red",
      fill=I("blue"))

#normal QQ plot
ggplot(train, aes(sample=Train_Fullmodel$residuals)) +
  stat_qq() +
  stat_qq_line()

#optional histogram
par(mfrow=c(1,2))
hist(residuals(Train_Fullmodel))
plot(Train_Fullmodel, which=2) #a Normal plot

#Testing for Normality
train_5000 = sample(1:nrow(train), size = 5000)
sam_train <- train[train_5000,]
sam_train
sam_Train_Fullmodel = glm(is_fraud ~ category+amt+gender+state+age+trans_p,
data=sam_train)
shapiro.test(residuals(sam_Train_Fullmodel))
```
```

## 6.2 R Code: Guiding Question 1

### 6.2.1 Logistic Regression

```
```{r}
library(car)
library(pROC)
library(verification)
```

```

# Reading dataset
```{r}
fraud <- read.csv("C:/Users/lawre/OneDrive/ucalgary/data
606/Project/fraudTrain.csv")
fraud_test <- read.csv("C:/Users/lawre/OneDrive/ucalgary/data
606/Project/fraudTest.csv")
dim(fraud)
colnames(fraud)
nrow(fraud)
```

Date transformation
```{r}
Date <- as.POSIXct(fraud$dob, format = "%Y-%m-%d")
fraud$Year <- as.integer(format(Date, format="%Y"))
head(fraud$Year)
fraud$age <- 2023-fraud$Year
tmp <- as.POSIXct(fraud$unix_time, origin="1970-01-01")
```
```{r}
Date <- as.POSIXct(fraud_test$dob, format = "%Y/%m/%d")
fraud_test$Year <- as.integer(format(Date, format="%Y"))
head(fraud_test$Year)
fraud_test$age <- 2023-fraud_test$Year
```
extract hour from transaction time
```{r}
fraud$trans_hour <- format(strptime(fraud$trans_date_trans_time, format = "%Y-%m-
%d %H:%M:%S"), format = "%H")
fraud_test$trans_hour <- format(as.POSIXct(fraud_test$trans_date_trans_time),
format ="%H")
```

generate transaction day night
```{r}
fraud$trans_P <- ifelse(as.integer(fraud$trans_hour) %in% seq(6,18), "day",
"night")
fraud_test$trans_P <- ifelse(as.integer(fraud_test$trans_hour) %in% seq(6,18),
"day", "night")
```

Sample the training dataset to maintain at least 5% of fraud transactions.
```{r}
set.seed(10)
fraud_subset_1 <- fraud[fraud$is_fraud == 1,]
fraud_subset_0 <- fraud[fraud$is_fraud == 0,]
idx <- sample(1:nrow(fraud_subset_0), size = 0.95*nrow(fraud_subset_1)/0.05)
new_train <- rbind(fraud_subset_0[idx,], fraud_subset_1)
left_out <- fraud_subset_0[-idx,]
table(new_train$is_fraud)
```

Fit the model with logistic regression
```{r}
fraud_LR_sample <- glm(factor(is_fraud)~category+amt+gender+state+age+trans_P ,
family = binomial, data = new_train)
summary(fraud_LR_sample)
vif(fraud_LR_sample)
```
```

Predict and generate matrix for testing dataset
```{r}
fraud_prob <- predict(fraud_LR_sample, fraud_test, type = "response")
fraud.predict = rep("NO", nrow(fraud_test))
fraud.predict[fraud_prob >= 0.5] = "Yes"
actual = fraud_test$is_fraud
table(fraud.predict, actual)
(1211+2251)/(1211+2511+551323+934)
```
Predict and generate matrixf for left out testing dataset
```{r}
fraud_prob <- predict(fraud_LR_sample, left_out, type = "response")
fraud.predict = rep("NO", nrow(left_out))
fraud.predict[fraud_prob >= 0.5] = "Yes"
actual = left_out$is_fraud
table(fraud.predict, actual)
```


```{r}
# generate the ROC curve and AUC
roc_curve <- roc(fraud_test$is_fraud, fraud_prob)
auc_value <- auc(roc_curve)
auc_value
# Visualize the ROC curve
test_roc = roc(fraud_test$is_fraud ~ fraud_prob, plot = TRUE, print.auc = TRUE)
```


```{r}
roc.plot(fraud_test$is_fraud, fraud_prob)
```


```{r}
pROC_obj <- roc(fraud_test$is_fraud, fraud_prob,
            smoothed = TRUE,
            # arguments for ci
            #ci=TRUE, ci.alpha=0.9, stratified=FALSE,
            # arguments for plot
            plot=TRUE, auc.polygon=TRUE, max.auc.polygon=TRUE, grid=TRUE,
            print.auc=TRUE, show.thres=TRUE)


#sens.ci <- ci.se(pROC_obj)
plot(pROC_obj, type="shape", col="lightblue", xlim = c(1,0))
## Warning in plot.ci.se(sens.ci, type = "shape", col = "lightblue"): Low
## definition shape.
#plot(sens.ci, type="bars")
```


Scorecard
```{r}
library(scorecard)
```
Prepare for dataset / group variables to bins
```{r}
score_card <- subset(new_train, select = c(category,amt,age,trans_P, is_fraud,
state, gender))
dt_f <- var_filter(score_card, y = "is_fraud")
dt_list <- split_df(dt_f, y ="is_fraud", ratios = c(0.8, 0.2), seed = 10)
bins <- woebin(dt_f, y = "is_fraud")
```

```
# apply bin to dataset
```{r}
dt_woe_list <- lapply(dt_list, function(x) woebin_ply(x, bins))
```

Implement logistic regression with stepwise model selection method
```{r}
lm_scorecard <- glm(is_fraud~., family = 'binomial' ,data = dt_woe_list$train)
vif(lm_scorecard)
m_step <- step(lm_scorecard, direction = "both", trace = FALSE)
m2 <- eval(m_step$call)
m2
summary(m2)
```

group variable in testing dataset as well, for prediction
```{r}
fraud_test1 <- subset(fraud_test, select = c(category,amt,age,trans_P, is_fraud,
state, gender))
bins_test <- woebin(fraud_test1, y = "is_fraud")
dt_woe_test <- woebin_ply(fraud_test1, bins_test)
```
inspect information value
```{r}
info_value = iv(score_card, y = "is_fraud")
info_value
```

prediction
```{r}
fraud_prob <- predict(m2, dt_woe_test, type = "response")
fraud.predict = rep("NO", nrow(fraud_test))
fraud.predict[fraud_prob >= 0.5] = "Yes"
actual = fraud_test$is_fraud
table_pred<- (table(fraud.predict, actual))
(9774+779)/sum(table_pred)
table_pred
```

predict and plot auc
```{r}
label_list <- lapply(dt_list, function(x) x$is_fraud)
pred_list <- lapply(dt_woe_list, function(x) predict(m2, x, type = 'response'))

perf <- perf_eva(pred = pred_list, label = label_list,  binomial_metric = c("mse",
"rmse",
"logloss", "r2", "ks", "auc", "gini"), show_plot = c("roc"))
perf$binomial_metric
```

transform the Logistic regression outcome into score
```{r}
points0 = 600
odds0 = 50
pdo = 100
card = scorecard(bins , m2)
score_list <- lapply(dt_list, function(x) scorecard_ply(x, card))
```

group the final score by woe method again, tuned.
```{r}


```
```

```
#breaks_adj <- list(
  #Score = c(26, 35, 40)
#)

#bins_adj <- woebin(sc, y = "is_fraud", breaks_list = breaks_adj)
breaks_adj <- list(
  score = c(310, 450, 540, 660, 730 ,800, 920)
  #c(310, 450, 540 ,600, 780, 930)
)


bin_for_scorecard <- cbind(score_list$train, dt_list$train$is_fraud)
bins_final <- woebin(bin_for_scorecard, y = "V2",breaks_list = breaks_adj)
bins_final
```

#export dataset
```{r}
write.csv(bin_for_scorecard, "bin_final.csv")
```

#export the grouping result
```{r}
library(dplyr)
category <- as.data.frame(card$category)
amt <- as.data.frame(card$amt)
age <- as.data.frame(card$age)
Trans_P <- as.data.frame(card$trans_P)
State <- as.data.frame(card$state)
characteristic_analysis <- rbind(category, amt,age,Trans_P,State)
write.csv(characteristic_analysis, "characteristic_analysis.csv")

card$basepoints
```
Plot ROC
```{r}
pROC_obj <- roc(dt_list$train$is_fraud , pred_list$train,
            smoothed = TRUE,
            # arguments for ci
            #ci=TRUE, ci.alpha=0.9, stratified=FALSE,
            # arguments for plot
            plot=TRUE, auc.polygon=TRUE, max.auc.polygon=TRUE, grid=TRUE,
            print.auc=TRUE, show.thres=TRUE)


#sens.ci <- ci.se(pROC_obj)
plot(pROC_obj, type="shape", col="lightblue", xlim = c(1,0))
## Warning in plot.ci.se(sens.ci, type = "shape", col = "lightblue"): Low
## definition shape.
#plot(sens.ci, type="bars")
```

6.2.2 QDA

Dummy Coding
```{r}
train = read.csv("C:/Users/ryeku/University of Calgary/Winter 2023 Group - General/606/Dataset/sampled_train.csv")
test = read.csv("C:/Users/ryeku/University of Calgary/Winter 2023 Group - General/606/Dataset/fraud_test_final.csv")
```

```{r}
```

```
head(train)
```

```{r}
# Convert characters to factors
train$category = as.factor(train$category)
train$gender = as.factor(train$gender)
train$state = as.factor(train$state)
train$trans_p = as.factor(train$trans_p)

test$category = as.factor(test$category)
test$gender = as.factor(test$gender)
test$state = as.factor(test$state)
test$trans_p = as.factor(test$trans_p)
```

```{r}
sapply(train[, c('category','gender','state','trans_p')], contrasts)
```

```{r}
# Convert categorical factors to numerical factors
train[, c('category','gender','state','trans_p')] = sapply(train[,
c('category','gender','state','trans_p')], unclass)
test[, c('category','gender','state','trans_p')] = sapply(test[,
c('category','gender','state','trans_p')], unclass)
```

```{r}
head(train)
```

QDA

```{r}
library(MASS)
library(ggplot2)
#iris[1:4] <- scale(iris[1:4])
qda.fit = qda(factor(is_fraud) ~ category+amt+gender+state+age+trans_p, data =
na.omit(train))
qda.fit
```

```{r}
plot(qda.fit)
```

```{r}
qda.pred = predict(qda.fit, test)
```

```{r}
table(qda.pred$class, test$is_fraud)
```
```{r}
cat("The misclassification rate is", (table(qda.pred$class,
test$is_fraud)[2]+table(qda.pred$class, test$is_fraud)[3])/dim(test)[1])
```

```{r}
library(klaR)
```

```
partimat(factor(is_fraud) ~ category+amt+gender+state+age+trans_p, data = train,
method="qda")
```

```{r}
library(verification)
library(pROC)
pROC_obj <- roc(test$is_fraud, qda.pred$posterior[,1],
            smoothed = TRUE,
            # arguments for ci
            #ci=TRUE, ci.alpha=0.9, stratified=FALSE,
            # arguments for plot
            plot=TRUE, auc.polygon=TRUE, max.auc.polygon=TRUE, grid=TRUE,
            print.auc=TRUE, show.thres=TRUE)

#roc.plot(test$is_fraud, qda.pred$x)
#sens.ci <- ci.se(pROC_obj)
#plot(pROC_obj, type="shape", col="lightblue", xlim =c(1,0))
```

6.2.3 Classification Tree

```{r}
library(caret)
library(AppliedPredictiveModeling)
library(truncnorm)
library(survey)
library(sampling)
library('ISLR')
library('ggplot2')
library(dplyr)
library(regclass)
library(mlbench)
library(MASS)
library(klaR)
library(car)
library(tree)
library(kmed)
library(randomForest)
library(reprtree)
library(pROC)
library(repr)
library(randomForest)
library(randomForestExplainer)
library(rpart.plot)
library(ROCR)
library(party)
```

```{r}
train= read.csv("sampled_train.csv")
test= read.csv("fraud_test_final.csv")
```
Wrangling and Merging the train and test to run tests
```{r}
# combining the rows of the two dataframes
set.seed(10)
new_train = subset(train, select = -c(X.1) )
colnames(new_train)[27]  <- "trans_p"
total_data = rbind(new_train,test)
```

```{r}


```

```
dim(train)
x = na.omit(train)
dim(x)
```

Classification Tree
```{r}
tree.class<-
tree(factor(is_fraud)~factor(category)+amt+gender+state+age+factor(trans_p),
new_train)
summary(tree.class)
```

```{r}
plot(tree.class)
text(tree.class, pretty=0)
```
#Accuracy
Error Rate: 0.003044704
6552 incidents were predicted no fraud while it is a fraud (Type2) and 681
incidents were predicted fraud while it is not a fraud.
```{r}
tree.pred<-predict(tree.class,test, type = "class")

tb = table(tree.pred,test$is_fraud)
tb
j =
(table(tree.pred,test$is_fraud)[2]+table(tree.pred,test$is_fraud)[3])/dim(test)[1]

```

```{r}
tree.preds <- predict(tree.class, test)
#tree.roc <- roc(test$is_fraud, tree.preds)
tree.preds[,2]
pROC_obj <- roc(test$is_fraud, tree.preds[,2],
            smoothed = TRUE,
            # arguments for ci
            #ci=TRUE, ci.alpha=0.9, stratified=FALSE,
            # arguments for plot
            plot=TRUE, auc.polygon=TRUE, max.auc.polygon=TRUE, grid=TRUE,
            print.auc=TRUE, show.thres=TRUE)


#sens.ci <- ci.se(pROC_obj)
plot(pROC_obj, type="shape", col="lightblue", xlim = c(1,0))
```

```{r}
cv.class<-cv.tree(tree.class, FUN = prune.misclass, K=14)
plot(cv.class$size, cv.class$dev,type="b")
```
```{r}
cv.class$dev
cv.class$size
```

```{r}
prune.class=prune.tree(tree.class,best=9)
plot(prune.class)
text(prune.class,pretty=0)
```

```
The misclassification rate is 0.00655.
2470 - False Negative, 1171 - False Positive
```{r}
prune.pred=predict(prune.class,test,type="class")
table(prune.pred,test$is_fraud)
i = (1014+678)/555719
i
prune.pred
confusionMatrix(prune.pred, factor(test$is_fraud))
```

```{r}
tree.preds <- predict(prune.class, test)
#tree.roc <- roc(test$is_fraud, tree.preds)
tree.preds[,2]
pROC_obj <- roc(test$is_fraud, tree.preds[,2],
            smoothed = TRUE,
            # arguments for ci
            #ci=TRUE, ci.alpha=0.9, stratified=FALSE,
            # arguments for plot
            plot=TRUE, auc.polygon=TRUE, max.auc.polygon=TRUE, grid=TRUE,
            print.auc=TRUE, show.thres=TRUE)

#sens.ci <- ci.se(pROC_obj)
plot(pROC_obj, type="shape", col="lightblue", xlim = c(1,0))
```

Random Forest

```{r}
rf <- randomForest(factor(is_fraud)~category+amt+gender+state+age+trans_p,
data=new_train, ntree = 500, norm.votes = FALSE)
print(rf)
```

```{r}
plot(rf)
```

```{r}
importance(rf)
```

```{r}
pp = predict(rf, test)
confusionMatrix(pp,as.factor(test$is_fraud))
```

```{r}
#rf.pred <- predict(rf, test)
#tree.roc <- roc(test$is_fraud, tree.preds)
#tree.preds[,2]
p2 <- predict(rf, test,type = "prob")
tmp <- as.numeric(p2)
pROC_obj <- roc(test$is_fraud, p2[,2],
            smoothed = TRUE,
            # arguments for ci
            #ci=TRUE, ci.alpha=0.9, stratified=FALSE,
            # arguments for plot
            plot=TRUE, auc.polygon=TRUE, max.auc.polygon=TRUE, grid=TRUE,
            print.auc=TRUE, show.thres=TRUE)
```
```

```
#sens.ci <- ci.se(pROC_obj)
plot(pROC_obj, type="shape", col="lightblue", xlim = c(1,0))
```

```{r}
best_tree = getTree(rf, k=1, labelVar = FALSE)
rf$forest[[1]]
```

```{r}
reprtree(rf$forest[[1]], show.node.info = TRUE, show.variable.info = TRUE)
```

```{r}
reprtree:::plot.getTree(rf)
reprtree:::plot.getTree(rf, k = 100, label = TRUE )
reprtree:::plot.getTree(rf, k = which.min(rf$err.rate[, "OOB"]),label= TRUE)
reprtree:::plot.tree(best.tree)
```

```{r}
Training = data.frame(new_train$amt,new_train$category,new_train$gender,
new_train$state, new_train$age, new_train$trans_p)
Testing = factor(new_train[["is_fraud"]])
```

```{r}
rf2 <- randomForest(factor(is_fraud)~category+amt+gender+state+age+trans_p,
data=new_train, ntree = 500, norm.votes = FALSE, importance=TRUE)
rf3 <- randomForest(x = Training, y= Testing, importance= TRUE)
print(rf3)
#explainer <- explain(rf3, data = test)
```

6.1.4 Model Validation

```
# K-Folds Cross Validation for Total Misclassification
```{r}
library(caret)
library(AppliedPredictiveModeling)
set.seed(10)
class.folds<-createFolds(total_data$is_fraud, k=10)
```

```{r}
mqda<-function(idx){
  Train<-total_data[-idx,]
  Test<-total_data[idx,]
  # Convert characters to factors
  Train$category = as.factor(Train$category)
  Train$gender = as.factor(Train$gender)
  Train$state = as.factor(Train$state)
  Train$trans_p = as.factor(Train$trans_p)

  Test$category = as.factor(Test$category)
  Test$gender = as.factor(Test$gender)
  Test$state = as.factor(Test$state)
  Test$trans_p = as.factor(Test$trans_p)

  Train[, c('category','gender','state','trans_p')] = sapply(Train[,
c('category','gender','state','trans_p')], unclass)
  Test[, c('category','gender','state','trans_p')] = sapply(Test[,
c('category','gender','state','trans_p')], unclass)
```
```

```
  fit<-qda(factor(is_fraud) ~ category+amt+gender+state+age+trans_p, data = Train)
  pred<-predict(fit,Test)
  pred_table = table(pred$class, Test$is_fraud)
  return((pred_table[2]+pred_table[3])/dim(Test)[1])
}
```

```{r}
mtc<-function(idx){
  Train<-total_data[-idx,]
  Test<-total_data[idx,]
  tree.class<-tree(factor(is_fraud)~category+amt+gender+state+age+trans_p, Train)
  tree.pred<-predict(tree.class,Test,type = "class")
  pred_table = table(tree.pred, Test$is_fraud)
  return((pred_table[2]+pred_table[3])/dim(Test)[1])
}
```

```{r}
LR_model<-function(idx){
  Train<-total_data[-idx,]
  Test<-total_data[idx,]
  fraud_LR_sample <- glm(factor(is_fraud)~category+amt+gender+state+age+trans_p ,
family = binomial, data = Train)
  #summary(fraud_LR_sample)
  #vif(fraud_LR_sample)
  fraud_prob <- predict(fraud_LR_sample, Test, type = "response")
  fraud.predict = rep("NO", nrow(Test))
  fraud.predict[fraud_prob >= 0.5] = "Yes"
  actual = Test$is_fraud
  pred_table = table(fraud.predict, actual)
  return((pred_table[2]+pred_table[3])/dim(Test)[1])
}
```

```{r}
qda_rate=lapply(class.folds,mqda)
tree_rate= lapply(class.folds,mtc)
LR_rate = lapply(class.folds,LR_model)
print(paste("QDA Error Rate : ",mean(as.numeric(qda_rate))))
print(paste("Tree Error Rate : ",mean(as.numeric(tree_rate))))
print(paste("Logistic Regression Error Rate : ",mean(as.numeric(LR_rate))))
```

```{r}
qda_rate
tree_rate
LR_rate
```

K-Folds Cross Validation for Checking for Type II Error
```{r}
mqda2<-function(idx){
  Train<-total_data[-idx,]
  Test<-total_data[idx,]
  # Convert characters to factors
  Train$category = as.factor(Train$category)
  Train$gender = as.factor(Train$gender)
  Train$state = as.factor(Train$state)
  Train$trans_p = as.factor(Train$trans_p)

  Test$category = as.factor(Test$category)
```

```
  Test$gender = as.factor(Test$gender)
  Test$state = as.factor(Test$state)
  Test$trans_p = as.factor(Test$trans_p)

  Train[, c('category','gender','state','trans_p')] = sapply(Train[,
c('category','gender','state','trans_p')], unclass)
  Test[, c('category','gender','state','trans_p')] = sapply(Test[,
c('category','gender','state','trans_p')], unclass)

  fit<-qda(factor(is_fraud) ~ category+amt+gender+state+age+trans_p, data = Train)
  pred<-predict(fit,Test)
  pred_table = table(pred$class, Test$is_fraud)
  return(pred_table[2]/dim(Test)[1])
}
```

```{r}
mtc2<-function(idx){
  Train<-total_data[-idx,]
  Test<-total_data[idx,]
  tree.class<-tree(factor(is_fraud)~category+amt+gender+state+age+trans_p, Train)
  tree.pred<-predict(tree.class,Test,type = "class")
  pred_table = table(tree.pred, Test$is_fraud)
  return(pred_table[2]/dim(Test)[1])
}
```

```{r}
LR_model2<-function(idx){
  Train<-total_data[-idx,]
  Test<-total_data[idx,]
  fraud_LR_sample <- glm(factor(is_fraud)~category+amt+gender+state+age+trans_p ,
family = binomial, data = Train)
  #summary(fraud_LR_sample)
  #vif(fraud_LR_sample)
  fraud_prob <- predict(fraud_LR_sample, Test, type = "response")
  fraud.predict = rep("NO", nrow(Test))
  fraud.predict[fraud_prob >= 0.5] = "Yes"
  actual = Test$is_fraud
  pred_table = table(fraud.predict, actual)
  return(pred_table[2]/dim(Test)[1])
}
```

```{r}
qda_rate=lapply(class.folds,mqda2)
tree_rate= lapply(class.folds,mtc2)
LR_rate = lapply(class.folds,LR_model2)
print(paste("QDA Error Rate : ",mean(as.numeric(qda_rate))))
print(paste("Tree Error Rate : ",mean(as.numeric(tree_rate))))
print(paste("Logistic Regression Error Rate : ",mean(as.numeric(LR_rate))))
```

6.3 R Code: Guiding Question 2

```
## Fraud only
```{r}
Fruad = train[train$is_fraud == 1,]
Fruad
```
```

6.3.1 LDA

```{r}
library(MASS)
lda.Fruad = lda(factor(category)~amt+gender+state+age+trans_hour,
data=Fruad_train)
summary(lda.Fruad)

sort(lda.Fruad$prior, decreasing = TRUE)
```

```{r}
library(klaR)
partimat(factor(category)~amt+gender+state+age+trans_hour, data=Fruad_train,
method="lda")
```

```{r}
roc.multi = multiclass.roc(Fruad_test$category, lda.Fruad.pred$posterior)
auc(roc.multi)
#rs <- roc.multi[['rocs']]
#plot.roc(rs[[1]])
```

6.3.2 QDA

```{r}
qda.Fruad = qda(factor(category)~amt+gender+state+age+trans_hour,
data=Fruad_train)
```

```{r}
library(klaR)
partimat(factor(category)~amt+gender+state+age+trans_hour, data=Fruad_train,
method="qda")
```

6.3.3 Classification Tree

```{r}
set.seed(10)
sample <- sample(c(TRUE, FALSE), nrow(Fruad), replace=TRUE, prob=c(0.7,0.3))
Fruad_train  <- Fruad[sample, ]
Fruad_test   <- Fruad[!sample, ]
```

```{r}
library(tree)
Fruad.tree <-tree(factor(category)~amt+gender+state+age+trans_hour, Fruad_train)
summary(Fruad.tree)
```
```{r}
plot(Fruad.tree)
text(Fruad.tree)
```

```{r}
cv.Fruad = cv.tree(Fruad.tree)
plot(cv.Fruad$size, cv.Fruad$dev, type='b')
print('Since the tree with 10 terminal nodes has lowest error, the tree should not
be pruned')
```

6.3.4 Model Validation

```
#LDA
```{r}
lda.Fruad.pred = predict(lda.Fruad, Fruad_test)

lda.table = table(lda.Fruad.pred$class, Fruad_test$category)
lda.table

accuracy_Test <- sum(diag(lda.table)) / sum(lda.table)
cat('The accuracy is', accuracy_Test)
```

```{r}
roc.multi = multiclass.roc(Fruad_test$category, lda.Fruad.pred$posterior)
auc(roc.multi)
```

#QDA
```{r}

qda.Fruad.pred=predict(qda.Fruad, Fruad_test)
qda.table = table(qda.Fruad.pred$class, Fruad_test$category)
qda.table
accuracy_Test <- sum(diag(qda.table)) / sum(qda.table)
cat('The accuracy is', accuracy_Test)
```

```{r}
library(pROC)
roc.multi.qda = multiclass.roc(Fruad_test$category, qda.Fruad.pred$posterior)
auc(roc.multi.qda)
```

#Tree
```{r}
Fruad.tree.pred=predict(Fruad.tree,Fruad_test,type="class")
table_mat= table(Fruad.tree.pred,Fruad_test$category)
table_mat
accuracy_Test <- sum(diag(table_mat)) / sum(table_mat)
cat('The accuracy is', accuracy_Test)
```

```{r}
Fruad.tree.pred=predict(Fruad.tree,Fruad_test,type="class")
table_mat= table(Fruad.tree.pred,Fruad_test$category)
table_mat
accuracy_Test <- sum(diag(table_mat)) / sum(table_mat)
cat('The accuracy is', accuracy_Test)
```
```