Case: Identifying Fraudulent Credit Card Transactions

Data Science for Business: Technical

Team Members: Bansi Shah (bks7385), Ayush Hate (aah9103), Nikita-Tarass Heumann (nh2514)

Date: May 11, 2023
Course Number: TECH-GB.2336
Pages: 16

1. Business Understanding

South State Bank is a 50-year-old local bank with a small customer base of 2,000 customers. Currently, South State Bank does not have any way of detecting credit card fraud, not even manual methods such as reviewing transactions that exceed a certain threshold, as the small bank operates with a limited set of customers. As a result, South State Bank has decided to explore the option of installing an automated fraud detection AI to handle the increasing volume of transactions over the recent years.

We are a specialized data science consultancy that was engaged by our client, South State Bank, to develop a proof of concept for an AI-based credit card fraud detection system. Our client is interested in understanding whether the financial benefits of implementing and operating such a system will outweigh the associated costs. Our analysis assumes that South State Bank's customer base stays the same over time, allowing us to provide a robust assessment of the potential return on investment.

To conduct our analysis, South State Bank provided us with a random dataset consisting of over 2,000 users and their transactions from 2002 to 2020. Our client has confirmed that all fraudulent transactions in this dataset have been identified. This assumption is because legitimate customers usually notice and report fraudulent activity [8].

2. Data Understanding

The bank has given us a dataset containing over 20 million transactions based on 2,000 randomly selected customers that are currently residing in the United States. The records cover multiple cards and purchases spanning 17 years, from 2002 to 2019. The data is provided in CSV format.

The data is organized into three main tables: credit card transactions, user details, and card details. The credit card transaction table is rich in data, containing various information such as the transaction time and location, payment mode, involved parties, amount, error occurrence, and the nature of the error. It contains over 24 million transaction details and 10 categorical variables. The user table contains demographic data about each individual user, such as their name, age, gender, address, personal income, per-capita income for their billing zip code, total debt, FICO score, and the number of cards they possess. The card table contains information about around 6,000 cards belonging to approximately 2,000 users, including details about the card's brand, type, expiration date, and account activation date.

Most transactions in the dataset are typically non-fraudulent because of the nature of fraudulent activities and their occurrence rates. Consequently, the data is imbalanced with a class distribution of 1:800, where the minority class represents fraudulent cases.

To solve the business problem, we need to build a model that can accurately distinguish between fraudulent and non-fraudulent transactions. This is a supervised classification task where the target variable (isFraud?) is either 1 (if the transaction is fraudulent) or 0 (if it is not). As the legitimate customer and the person committing the fraud are assumed to be two different individuals with opposing intentions, we can assume that almost all instances of fraud have been identified and accurately labelled. [8]

2.1. Deep dive user table

This section explores the potential correlation between various attributes of bank customers and credit card fraud. The User table contains 2000 distinct entries, each representing a unique customer of the bank. Attributes available in the table include 'Person', 'Current Age', 'Retirement Age', 'Birth Year', 'Gender', 'Address', 'Apartment', 'City', 'State', 'Zipcode', 'Latitude', 'Longitude', 'Per Capita Income - Zipcode', 'Yearly Income - Person', 'Total Debt', 'FICO Score', and 'Num Credit Cards'.

- **Variable Name**	**Description**	**Feature (Y/N)**	Engineered
Person	Unique identifier for each person	-	N
Current Age	Age of the person at the time of data collection	N	N
Retirement Age	Age at which the person plans to retire	N	N
Birth Year	Year of birth of the person	N	N
Birth Month	Month of birth of the person	N	N
Gender	Gender of the person (e.g., Male, Female, Other)	N	N
Address	Street address of the person	N	N
Apartment	Apartment number of the person (if applicable)	N	N
City	City where the person resides	N	N
State	State where the person resides	N	N
Zipcode	Postal code of the person's address	N	N
Latitude	Latitude of the person's address	N	N
Longitude	Longitude of the person's address	N	N
Per Capita Income - Zipcode	Per capita income for the zip code of the person's address	N	N
Yearly Income - Person	Yearly income of the person	Y	N
Total Debt	Total amount of debt owed by the person	Y	N
FICO Score	FICO credit score of the person	Y	N

Num Credit Cards	Number of credit cards held by the person	Y	N
User Cluster Key	Cluster user belongs to	N	Y
Current Age Bin	Range of age user falls into	Y	Y

Table 1 User table

Our hypothesis is that 'Current Age', 'Gender', 'City', 'Per Capita Income - Zipcode', 'Yearly Income - Person', 'Total Debt', and 'FICO Score' may be correlated with the target value of credit card fraud. After conducting an analysis of the available attributes, we have identified five potential predictors of fraud:

1. Current Age: There is a positive correlation between current age and fraudulent activity. The Pearson's correlation coefficient of 0.59 suggests a moderately strong positive linear relationship between the current age and the incidence of fraudulent transactions per customer per 1000 transactions. The p-value of 8.72e-09 indicates that this correlation is statistically significant.
2. Yearly Income - Person: There is a negative correlation between yearly income and fraudulent transactions. The Pearson's correlation coefficient of -0.09 suggests a weak negative linear relationship between yearly income and the incidence of fraudulent transactions per customer per 1000 transactions. The p-value of 6.30e-05 indicates that this correlation is statistically significant.
3. FICO Score: There is a positive correlation between FICO score and the incidence of fraudulent transactions per customer per 1000 transactions. The Pearson's correlation coefficient of 0.11 suggests a weak positive linear relationship between FICO score and the incidence of fraudulent transactions. The p-value of 0.05 indicates that this correlation is marginally significant.
4. Total Debt: Like yearly income, total debt has a negative correlation with the incidence of fraudulent transactions. The Pearson's correlation coefficient of -0.12 suggests a weak negative linear relationship between total debt and the incidence of fraudulent transactions per customer per 1000 transactions. The p-value of 8.95e-08 indicates that this correlation is statistically significant.
5. Num Credit Cards: There is a strong positive correlation between the number of credit cards held by a customer and the incidence of fraudulent transactions per customer per 1000 transactions. The Pearson's correlation coefficient of 0.993 suggests a very strong positive linear relationship between the number of credit cards and the incidence of fraudulent transactions. The p-value of 6.41e-08 indicates that this correlation is statistically significant.

Based on these findings, we can conclude that these five customer attributes are indeed correlated with credit card fraud risk. Based on these findings, the five attributes are believed to be effective at differentiating the risk of credit card fraud on the customer level. Therefore, these attributes were specifically chosen to generate customer clusters using k-means. By analyzing these attributes, we can gain insights into the spending habits,

creditworthiness, and financial stability of individuals. Current age and income can provide an indication of a person's earning potential and their ability to repay credit card debt, while FICO score, total debt, and the number of credit cards can help assess a person's creditworthiness and likelihood of defaulting on payments. K-means clustering is a popular unsupervised machine learning technique used to identify similar groups or clusters of data points in a dataset. In this case, the dataset likely contained information about individuals' current age, yearly income, FICO score, total debt, and number of credit cards. To address the 'curse of dimensionality', the initial step involves conducting Principal Component Analysis (PCA). This technique maps the features to a reduced subspace. This reduction of dimensionality results in a more efficient computational process. K-means clustering would then be applied to group individuals into clusters based on similarities or differences in these attributes. The resulting clusters ideally reflect different risk classes for loan applications.

To determine the optimal number of clusters for k-means clustering, the model parameter k is derived with the silhouette analysis method. The Silhouette score can be easily calculated in Python using the metrics module of the scikit-learn. [7] A graph is plotted between different number of clusters (x-axis) and the corresponding silhouette score. The curve typically exhibits a decreasing trend, where the improvement in clustering performance diminishes as the number of clusters increases. The point on the curve where the improvement becomes marginal can be considered as the optimal number of clusters for the dataset. Figure 1 shows that the curve begins to level off at k = 7, suggesting that additional clusters beyond 7 may not significantly improve the clustering results. In addition to this, the Silhouette score was also plotted to confirm the quality of clusters produced. Therefore, k was set to 7 for the k-means clustering algorithm going forward.

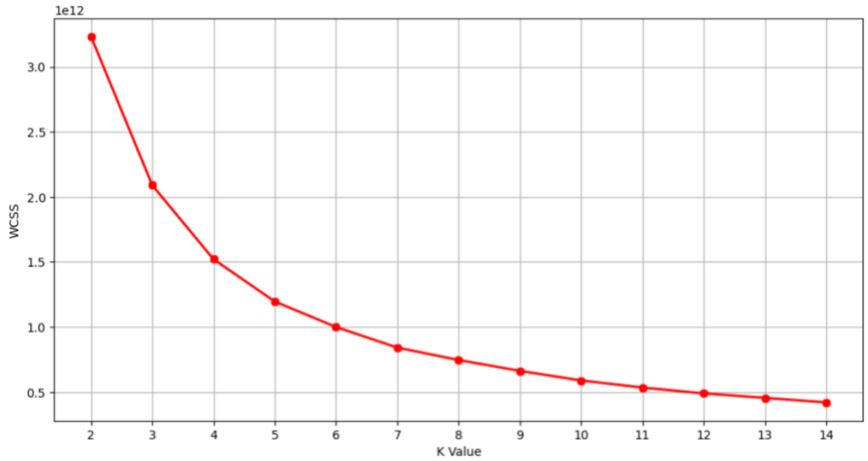

Figure 1 WCSS Plot for K-Means

The k-means clustering delivers 7 different distinct clusters, as to be seen in figure 2.

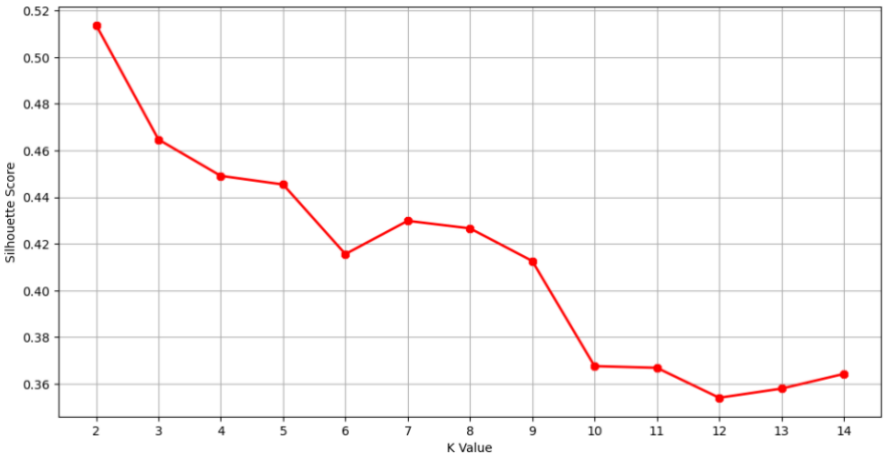

Figure 2 Silhouette Score

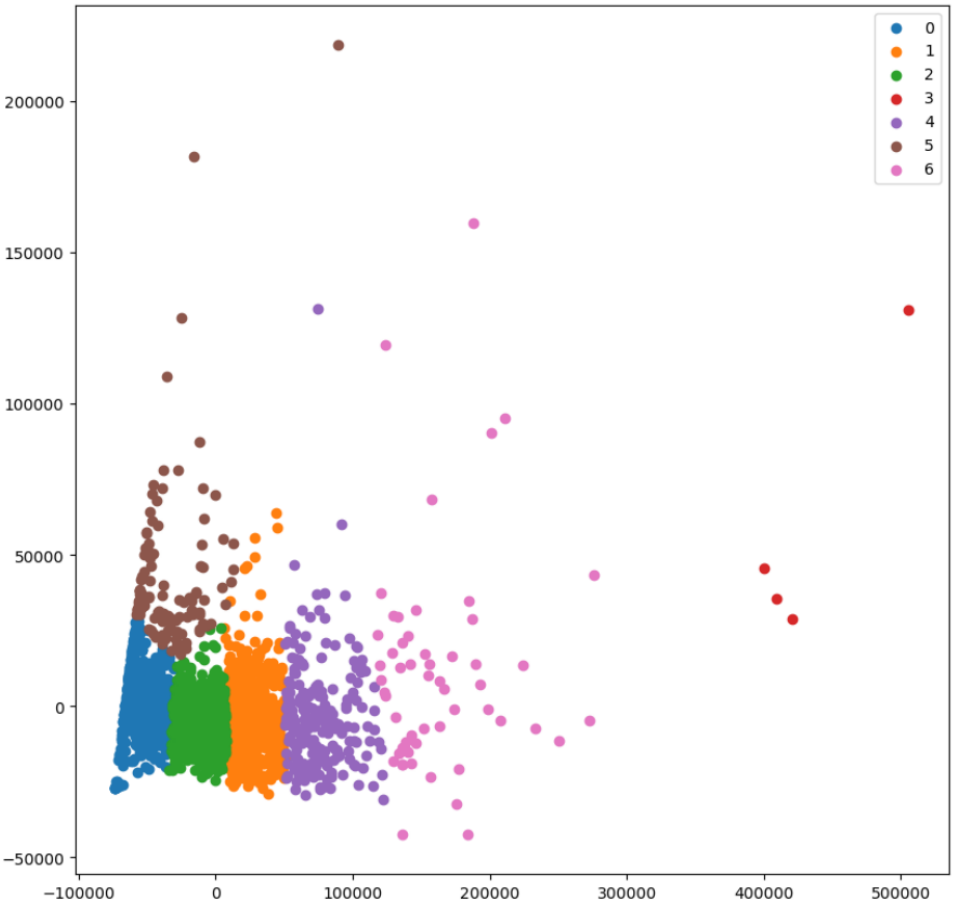

Figure 3 2D Cluster Representation

2.2. Deep dive card table

The Cards table contains around 6100 instances that provide details about card features such as 'Card Brand', 'Card Type', 'Expires', 'Credit Limit', 'Acct Open Date', 'Year PIN last Changed', 'Card Number', 'CVV', and 'Has Chip'. The table includes categorical features such as 'Card Brand', 'Card Type', and 'Has Chip', while

'Expires', 'Acct Open Date', and 'Year PIN last Changed' are time-based features, and the remaining features are numerical. However, since this table does not have the target variable, it is merged with the transactions table based on the user and card index.

The inclusion of "Card Number" and "CVV" as features for fraud detection presents challenges due to their sensitive nature. These attributes are typically encrypted or masked to ensure security and privacy. While "Card Number" and "CVV" may contain certain information that could potentially contribute to fraud detection, there are typically other attributes that hold greater relevance and significance in identifying fraudulent activities. Additionally, the attribute "Card on Dark Web" has a constant value throughout the dataset, indicating no variability. As a result, it is unlikely to provide substantial insights or contribute significantly to fraud detection. “Account Open Date” falls within an acceptable range and is logically consistent with “Expires” and “Year PIN last Changed”.

Variable Name	**Description**	**Feature (y/n)**	**Engineered**
User	Unique identifier for each user	N	N
CARD INDEX	Unique identifier for each card issued to the user	N	N
Card Brand	Brand of the card (e.g., Visa, Mastercard)	Y	N
Card Type	Type of the card (e.g., Debit, Credit)	Y	N
Card Number	Unique identifier for the card number	N	N
Expires	Expiration date of the card	N	N
CVV	Security code for the card	N	N
Has Chip	Whether the card has a chip or not	Y	N
Cards Issued	Total number of cards issued to the user	Y	N
Credit Limit	Maximum amount of credit allowed for the card	N	N
Acct Open Date	Date the account was opened	N	N
Year PIN last Changed	The last year in which the PIN was changed	N	N
Card on Dark Web	Whether the card is present on the Dark Web or not	N	N
Account Age	How many years ago was the account created	Y	Y
Pin Change Age	How many years ago was the pin changed	Y	Y

Table 2 Card table

We’ve made the following observations after analyzing the cards table:

1. For Amex and Discover brands, the table contains only credit card transactions whereas Visa and Mastercard have transactions from debit and debit(prepaid) cards as well. Amex cards exhibit a higher vulnerability to fraud when compared to other card brands. [Figure 18]
2. We calculated the time difference between the transaction and the year pin was last changed. Despite having recently changed their PIN (a characteristic associated with lower susceptibility to fraud), cards experience a higher frequency of fraudulent incidents compared to cards with unchanged PINs. [Figure 19]
3. Regardless of the presence of a chip, debit cards display a similar level of vulnerability to fraud. On the other hand, credit cards equipped with a chip demonstrate enhanced security measures, while debit (prepaid) cards without a chip are more susceptible to fraudulent activities. [Figure 20]
4. With an increase in the number of cards issued, the likelihood of fraud also tends to rise for both debit cards. However, credit cards remain more vulnerable to fraudulent activities regardless of the number of cards issued. [Figure 21]

2.3. Deep dive transaction table

The credit card transaction table contains data for over 24 million transactions, with unique user and card identifiers. The table includes time-related attributes, such as 'Year,' 'Month,' 'Day,' and 'Time' of the transaction, as well as merchant-related information such as 'Merchant Name,' 'Merchant City,' 'Merchant State,' 'Zip,' and 'Merchant Category Code (MCC).' The table also has a categorical feature denoting the reason for transaction failure ('Error?') and the target variable 'IsFraud?' indicating whether a transaction was fraudulent. There are no null values in the table except for 'Nan' values in 'Errors?'. These NaNs indicate error-free transactions.

Variable Name	**Description**	**Feature (y/n)**	**Engineered**
User	Unique identifier for each user	-	N
Card	Unique identifier for each card used in the transaction	-	N
Year	Year of the transaction	Y	N
Month	Month of the transaction	N	N
Day	Day of the transaction	Y	N
Time	Time of the transaction	N	N
Amount	Amount of the transaction	Y	N
Use Chip	Whether the card was used with a chip or not	Y	N
Merchant Name	Name of the merchant where the transaction occurred	Y	N
Merchant City	City where the merchant is located	N	N
Merchant State	State where the merchant is located	Y	N

Zip	Postal code of the merchant's address	N	N
MCC	Merchant Category Code	Y	N
Errors?	Whether there were any errors in the transaction	N	N
Is Fraud?	Whether the transaction is fraudulent or not	N	N
Hour	Hour of transaction	Y	Y
Minute	Minute of transaction	Y	Y
In Market Time	Whether the transaction occurred in market time	Y	Y
Is Online	Whether the transaction was online	Y	Y
Fraud Merchant	Whether transaction was to a top 10 fraud merchant	Y	Y
Fraud Merchant State	Whether transaction was made to a top 10 fraud merchant state	Y	Y
Fraud Merchant City	Whether transaction was made to a top 10 fraud merchant city	Y	Y
Fraud Merchant Zip	Whether transaction was made to a top 10 fraud merchant zip	Y	Y
Top Errors	Whether transaction had one of top 6 errors	Y	Y
Merchant User Distance	Distance between merchant zip and user zip	Y	Y
Daily Spend	Amount spent by a user from a card in one day	Y	Y
Weekly Average Spend	Average amount spent by a user in one week for a card based on daily spend	Y	Y
Daily Weekly Spend Ratio	Ratio of daily spend and weekly average spend	Y	Y
Hourly Spend	Amount spent by a user for a card in an hour	Y	Y
Weekly Average Hourly Spend	Average amount spent by a user in one week for a card based on hourly spend	Y	Y
Hourly Weekly Spend Ratio	Ratio of hourly spend, and weekly average hourly spend	Y	Y
Daily Frequency	Frequency of transactions of a user for a card in one day	Y	Y
Weekly Frequency	Frequency of transaction of a user for a card in one week	Y	Y
Daily Weekly Count Tx Ratio	Ratio of daily frequency and weekly frequency	Y	Y
Hourly Frequency	Frequency of transactions of a user for a card in an hour	Y	Y
Weekly Average Hourly Frequency	Average frequency of transaction of a user for a card in one week based on hourly frequency	Y	Y
Hourly Weekly Frequency Ratio	Ratio of hourly frequency and weekly average hourly frequency	Y	Y

Table 3 transaction table

Our analysis of the attributes has provided the following insights:

1. Time-based attributes: We observed patterns in yearly and hourly analyses, which revealed a higher frequency of fraud during recent times and market hours (9 a.m. - 3 p.m.). However, we found no significant patterns or insights at the minute and monthly levels. [Figure 22, 23]
2. 'Use Chip': Our analysis of chip usage during transactions suggested that online transactions have a higher likelihood of experiencing fraudulent activities. [Figure 24]
3. Location-based attributes: An analysis of the merchant's location at the state, city, and zip levels indicated a higher concentration of fraud in certain regions. [Figure 25]
4. 'Error?': Our analysis of errors linked to fraud revealed that more cases were associated with CVV, card number, expiration date, pin, and balance. [Figure 26]
5. 'MCC': By analyzing the merchant category code (MCC), we identified patterns and correlations between specific merchant categories and fraudulent activities. Certain merchant categories exhibited a higher likelihood of fraudulent transactions.

3. Data Preparation

The transactions data alone is not sufficient for detecting fraud. We integrate all three tables based on user and card index to engineer features that contribute more to the target variable. We ensure that there's no data leakage as all the attributes will be available at the time of model use, becoming available as soon as the transaction is initiated. (We'll initially split the data, preprocess the training set, perform analysis on it and then preprocess the evaluation set. We'll only see ROC curves and other performance metrics for the training set. Overall, these insights will be invaluable in developing a more effective fraud detection system for our client, South State Bank.)

The data is initially divided into training (all the transactions before the year 2019) and evaluation sets (all the transactions in the year 2019). The training set undergoes independent preprocessing and analysis to uncover feature insights and correlations, ensuring it remains unaffected by the evaluation set. Subsequently, the evaluation set is processed to incorporate the required engineered features. The training set is further divided into a training and testing set. The sub-training (and testing) set is used to test out different models and evaluate ROC curves and other performance metrics.

For categorical variables such as 'Use Chip', 'Gender', 'Card Brand', 'Card Type', 'Has Chip', and 'Current Age Bin' with fewer than 5 distinct values, we performed one hot encoding. This encoding technique transforms these categorical variables into binary vectors, where each distinct value becomes a separate binary feature. For categorical variables including 'Merchant Name', 'Merchant City', 'Merchant State', 'MCC', and 'Errors?' With a

lot more distinct values, we implemented label encoding based on the value count of each column. This allows the machine learning algorithms to effectively incorporate these categorical attributes into the fraud detection process and capture any potential patterns or correlations associated with them. Label encoding assigns a numerical label to each unique value in the categorical variable based on its frequency of occurrence.

We convert the 'age' attribute to a 'Current Age Bin' to simplify the representation of age demographics. Next, we find the top 10 merchants with highest frequency of fraud transactions and engineer a Boolean attribute that tells us whether the transaction was made to one of the top 10 fraud merchants (value = 1) or not (value = 0). We evaluate and add features in the same manner for top 10 fraud states, cities, zip codes and top 6 errors at the time of transaction. Then, we calculate the distance between the user and the merchant using the haversine formula [4]. It uses latitude and longitude to measure the distance between two points. Since we do not have this information for the users, we used a zip code mapping dataset with all the zips and their respective latitude and longitude within the US sourced from the government census [2]. It is also important to engineer velocity features using the transactions to identify spending patterns and detect anomalies or unexpected behavior. ‘Daily Spend’ and ‘Hourly Spend’ are calculated by taking the sum of amounts spent on transactions made with each card of every user on a day-to-day and hour-by-hour basis respectively. ‘Weekly Average Spend’ and ‘Weekly Average Hourly Spend’ are measured by taking the average of amounts spent on transactions made with each card of every user on a daily spend and hourly spend basis respectively. Similarly, we can compute ‘Daily Frequency’, ‘Weekly Frequency’, ‘Hourly Frequency’ and ‘Weekly Average Hourly Frequency’ by considering the number of transactions instead of the transaction amounts. Additionally, we calculate the ratio of the daily and weekly attributes. By computing these ratios, we gain valuable insights into the relative proportions and distributions of transaction activity on a daily and weekly basis. This analysis allows us to identify any significant variations or deviations in transaction patterns between different time periods. By considering these ratios, we can further enhance our understanding of the temporal dynamics and trends within the transaction data. As date type attributes have limited direct contribution to training the model for fraud detection, we transform these attributes to yield better results. Specifically, we create new features such as 'pin change age' (calculated as the difference between the transaction year and the year when the PIN was last changed) and 'account age' (calculated as the difference between the transaction year and the year when the account was opened).

Relative feature importance is evaluated when testing the model using the sub-training data. This feature importance will be used to prune low-importance features when the model is run on the evaluation data.

Our final preprocessed tables have the attributes both raw and engineered as mentioned in the deep dive sections and will be the training set for the machine learning models. The instance for these tables is a transaction with these attributes describing it. Our objective is to leverage this and deduce insights to develop an effective AI-based fraud detection system for our client, South State Bank.

4. Modeling

4.1. Model Selection

In the current scenario, the bank does not have a fraud detection model in place. So, it becomes necessary to establish a naïve baseline that can be used as a reference for comparing the performance of more advanced models. Even though the naive model may not be highly accurate or sophisticated, it provides a basic mechanism to flag potentially fraudulent transactions. It demonstrates the need for a dedicated fraud detection model to stakeholders within the bank. After establishing a naïve baseline, we applied three distinct machine learning algorithms to the dataset: Logistic Regression, Random Forest Classifier, and XGBoost Classifier and evaluated and contrasted their effectiveness.

1. Logistic Regression is commonly employed for binary classification tasks. The advantages of logistic regression include its simplicity, interpretability, and computational efficiency. It can handle many input features. We run logistic regression with different values of the regularization parameter (c) for each cluster in the data separately and visualizes the corresponding AUC scores to help determine the optimal c value for the model. We find that for all the clusters the optimal value of c is 0.35.

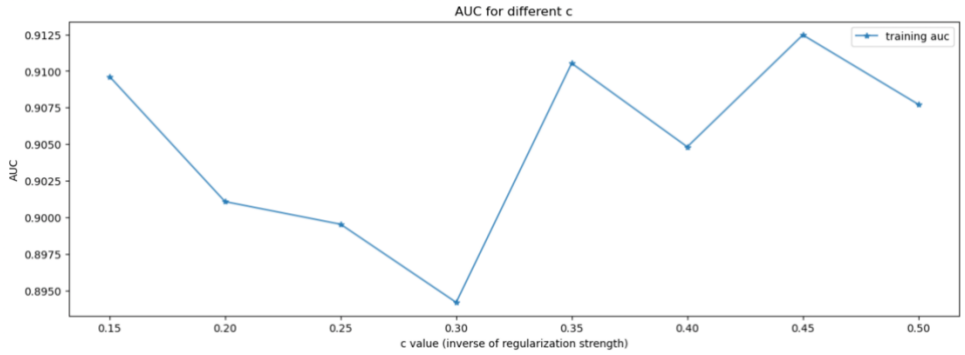

Figure 4: AUC curve for optimal c

2. Random Forest Classifier works by creating a set of decision trees on random subsets of the data and then aggregating their predictions to produce the final output. One of the key advantages of Random Forest is it can effectively handle imbalanced data. It mitigates the risk of overfitting by using bagging and random feature selection techniques. It can handle complex datasets and capture non-linear relationships between the input features and the target variable. We run random forest classifier with different values

of ‘n_estimators’ parameter or each cluster in the data separately and visualizes the corresponding AUC scores to help determine the optimal number of trees for the model. We find that for all the clusters the optimal value of ‘n_estimators’ is 125.

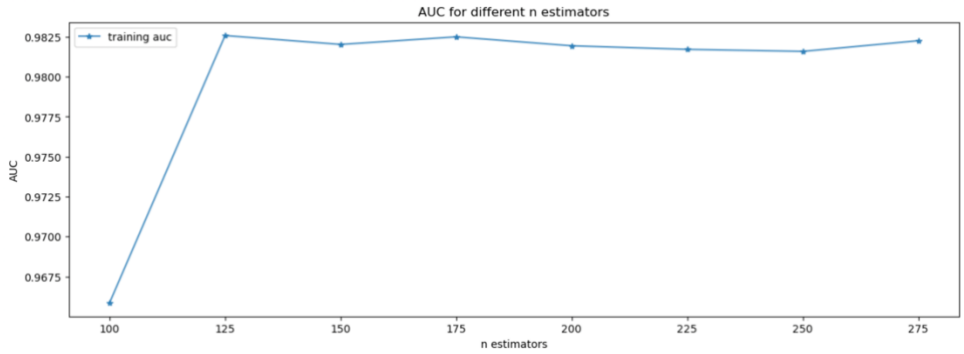

Figure 5: AUC curve for optimal n_estimators

3. XGBoost Classifier is an optimized implementation of gradient boosting algorithms. By iteratively adding weak learners (decision trees) and optimizing a specific loss function, XGBoost effectively captures intricate patterns and relationships in the data. One advantage of XGBoost is its scalability, allowing it to handle large datasets with a high number of features. It also provides built-in mechanisms for handling missing values and dealing with imbalanced datasets. Additionally, XGBoost offers flexibility in terms of hyperparameter tuning, enabling fine-grained control over the model's behaviour. We run XGBoost Classifier with different values of ‘n_estimators’ parameter or each cluster in the data separately and visualizes the corresponding AUC scores to help determine the optimal number of trees for the model. We find that for all clusters ranging from 0 to 6 the optimal values of ‘n_estimators’ are 300, 250, 300, 150, 200, 300 and 350 respectively.

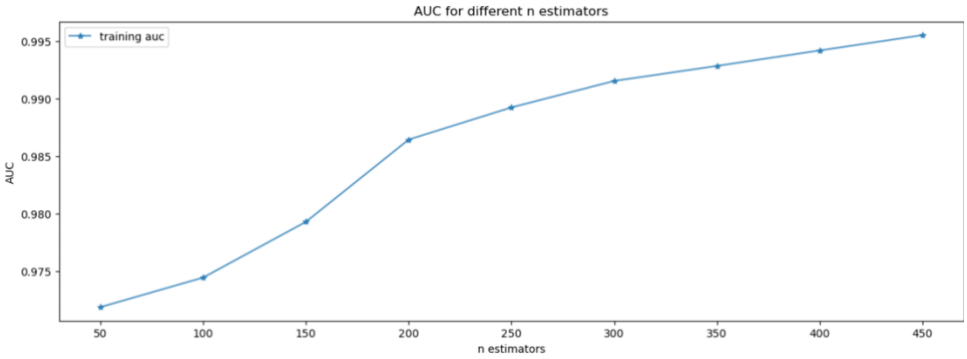

Figure 6: AUC curve for optimal n_estimators for cluster 0

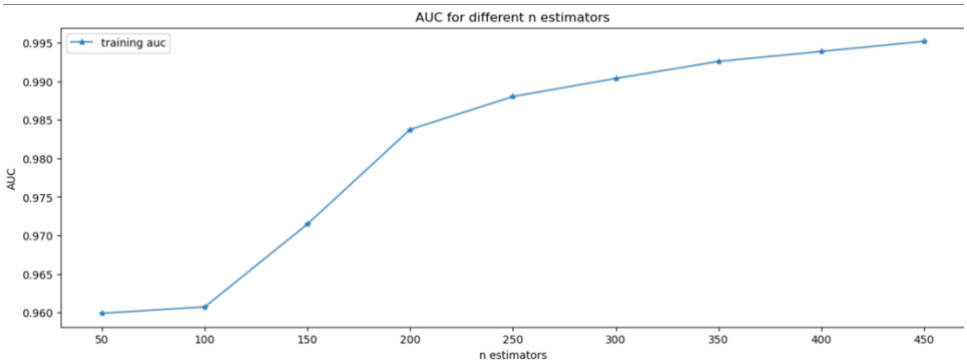

Figure 7: AUC curve for optimal n_estimators for cluster 1

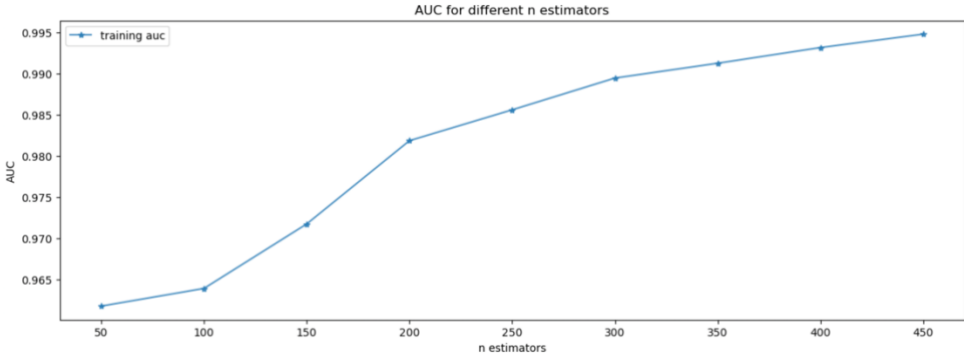

Figure 8: AUC curve for optimal n_estimators for cluster 2

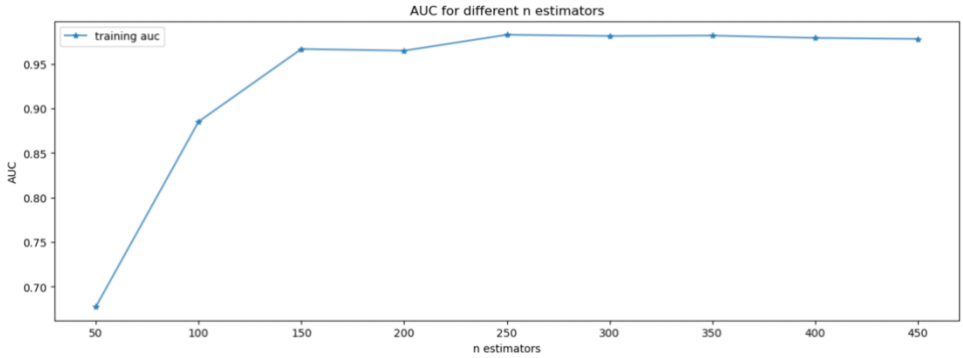

Figure 9: AUC curve for optimal n_estimators for cluster 3

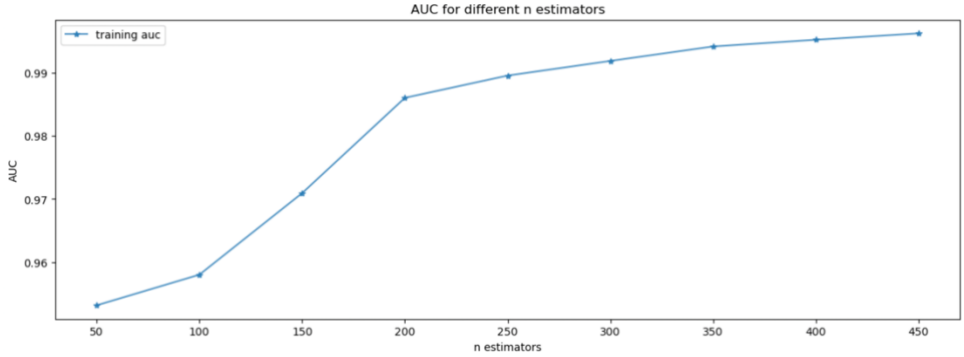

Figure 10: AUC curve for optimal n_estimators for cluster 4

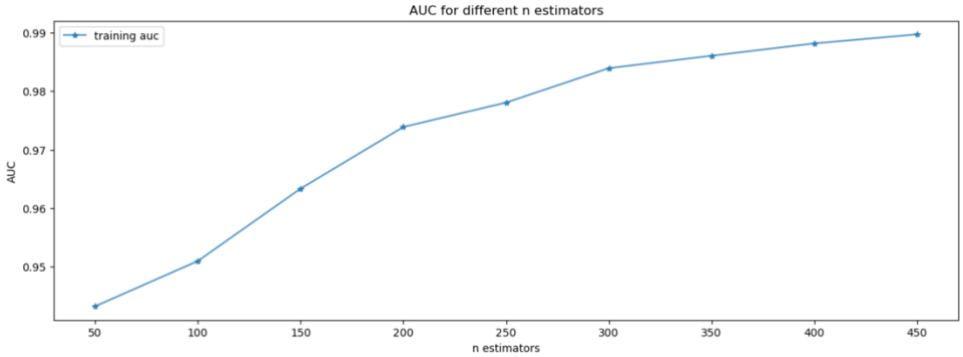

Figure 11: AUC curve for optimal n_estimators for cluster 5

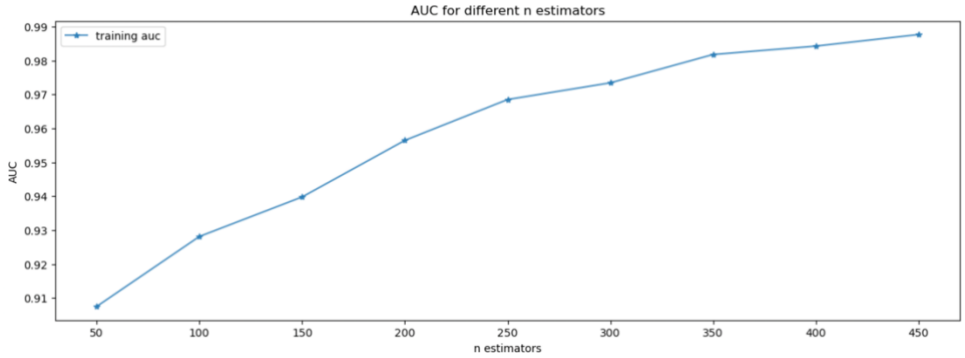

Figure 12: AUC curve for optimal n_estimators for cluster 6

4. Alternatively, SVM or Naïve Bayes could be used for a similar problem formulation. SVM works by finding the optimal hyperplane to separate different classes of data points while maximizing the margin between classes. However, SVM can be computationally intensive and may not perform well with very large datasets. Naive Bayes assumes independence between features, which may not hold true in some datasets.

4.2. Model Justification

We have opted to use XGBoost classifier for our problem for several compelling reasons. Despite the advantages of other models such as logistic regression and random forest, there are certain limitations that make them less suitable for our specific problem. Logistic regression, for instance, tends to struggle with imbalanced datasets where the majority class heavily outweighs the minority class. While random forest is also capable of handling imbalanced data to some extent, XGBoost's specialized techniques and optimization algorithms give it an edge in terms of handling imbalance more efficiently. Moreover, XGBoost provides advanced regularization techniques, such as L1 and L2 regularization, which can help prevent overfitting and enhance the model's generalization capabilities. Also, XGBoost offers interpretable feature importance analysis, enabling us to identify the most influential features in detecting fraudulent transactions. The feature importance matrix that was used for feature selection is in figure 27 (Appendix B).

The following are the performance metrics of the models tried on the sub-testing data:

Model	Expected Profit	Max Profit
Logistic Regression	-0.0059	108738.16
Random Forest	0.024	424885.78
XGBoost	0.025	434290.90
Naïve	-0.0000002	-5759.52

Based on the evaluation results of training set, it appears that the XGBoost classifier outperforms both the Random Forest and Logistic Regression models in terms of profitability for fraud detection. The XGBoost model achieves a maximum profit of $434,290.90 with an expected profit per customer of $0.025, which is higher than the profits obtained from the Random Forest (maximum profit of $424,885.78 and expected profit per customer of $0.024) and Logistic Regression (maximum profit of $108,738 and expected profit per customer of -$0.0059 (loss)) models.

4.3. Business Impact and Problem Solving

Our solution addresses the business problem effectively by improving the following dimensions of interest to the bank:

1. Increased detection of fraudulent transactions: The model selected has a greater capability of identifying true fraudulent transactions when compared to the current solution and other models. This can help the bank to detect more cases of fraud that might have gone unnoticed, and thus prevent significant losses.
2. Improved customer satisfaction: Fraudulent transactions can lead to negative impacts on customers, such as unauthorized transactions or account freezes. By detecting and preventing fraud more effectively, the bank can minimize the inconvenience caused to customers, and thereby improve their satisfaction.
3. Increased profitability: By reducing the number of fraudulent transactions and the associated losses, the model can help the firm to improve profitability.
4. Effective Feature Importance: The model highlights the variables that have significant impact on fraud detection. This information can help the bank understand the key indicators of fraudulent activities and prioritize their prevention efforts accordingly. By focusing on the most relevant features, the model allows for more targeted and efficient fraud detection strategies.

Therefore, the bank can minimize financial losses, preserve its reputation, and maintain customer trust

5. Evaluation

The ROC curve is a graphical representation of the trade-off between the true positive rate (sensitivity) and the false positive rate (1 - specificity) at various threshold settings. It helps to assess the model's ability to correctly classify fraud cases and non-fraud cases. A higher area under the ROC curve indicates better discrimination power of the model. However, in fraud detection, simply focusing on the accuracy of classification may not be sufficient. It is also important to consider the associated costs and benefits. This is where the profit curve comes into play. The profit curve considers the costs of false positives and false negatives, along with the benefits of true positives, to determine the optimal threshold for classifying transactions as fraudulent or non-fraudulent. The combination of ROC curve and profit curve allows us to make informed decisions regarding the model's performance in real-world scenarios, considering the potential financial implications.

In the cost structure, we have analyst fee as $15. This fee is charged for the analysis and processing of each claim. There is currently no additional claim fee associated with filing a claim. However, there is a penalty of $0.04/dollar for a false positive claim that the bank incurs. Using these we generate the cost - benefit matrix given below:

	Actual		
Predicted		p	n
	Y	Amount - Analyst Fee	-Analyst Fee
	N	claim Penalty * Amount	0

Figure 3 Cost-Benefit matrix

To precisely assess the financial impact of the model’s prediction, we calculate the maximum expected profit and the expected profit per customer. By considering the potential profit or loss associated with different prediction outcomes (true positive, false positive, true negative, false negative) and weighted by the probability of the target variable, we can measure the overall profitability of the model's predictions on the test data. This evaluation metric provides a practical and meaningful assessment of the model's performance in terms of its ability to generate profits or minimize losses for the business. Since our data is imbalanced, we use the expected profit equation with priors factored in. [8]

Expected Profit = (Probability of Fraud) * [(True Positive Rate * True Positives) + (False Negative Rates * False Negatives)] + (Probability of Not Fraud) * [(False Positive Rate * False Positives) + (True Negative Rate * True Negatives)]

In the case of the naive baseline model, the expected profit per customer is -$0.0000025, indicating a slight expected loss. On the other hand, when using XGBoost, the expected profit per customer improves significantly to $0.011. This suggests that the XGBoost model has the potential to generate a positive profit per customer. Additionally, the model achieves a maximum profit of $94679.37. Hence, XGBoost model outperforms all other models, including the current system in terms of generating profits and improving the financial outcomes for the business.

6. Deployment

We assume that the implementation of the AI system would cost $60,000, $50,000 for initial implementation and $10,000 for annual operating costs. Our outcome from evaluation in chapter 5 shows that we are to expect a yearly profit of $94,679.37. The analysis suggests that the implementation of an AI-based credit card fraud detection system has the potential to provide a ROI of 58% in the first year of operation. This indicates that the initial investment of $60,000 could be recouped in just under one year of operation.

Next to the financial benefits, the implementation of the AI-based credit card fraud detection system is expected to reduce the potential for fraud, increase customer satisfaction, and help maintain the bank's reputation. Furthermore, the AI system is likely to assist the bank in complying with regulatory and compliance requirements.

Based on these findings, it is recommended that South State Bank consider implementing the AI-based credit card fraud detection system due to its high potential ROI and the benefits it offers to the bank and its customers.

Appendix A: List of References

(1) *Credit Card Transactions*. Kaggle. (n.d.). https://www.kaggle.com/datasets/ealtman2019/credit-card-transactions?resource=download&select=sd254_cards.csv

(2) Index of /Geo/Docs/Maps-Data/Data/gazetteer/2020_gazetteer. (n.d.). https://www2.census.gov/geo/docs/maps-data/data/gazetteer/2020_Gazetteer/

(3) Banerji, A. (2023, April 13). *K-mean: Getting the optimal number of clusters*. Analytics Vidhya. https://www.analyticsvidhya.com/blog/2021/05/k-mean-getting-the-optimal-number-of-clusters/

(4) Upadhyay, A. (2018, June 20). *Haversine formula - calculate geographic distance on Earth*. Haversine formula - Calculate geographic distance on earth. https://www.igismap.com/haversine-formula-calculate-geographic-distance-earth/

(5) *Sklearn.ensemble.randomforestclassifier*. scikit. (n.d.). https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

(6) *Sklearn.linear_model.logisticregression*. scikit. (n.d.-b). https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html

(7) *sklearn.metrics.silhouette_score.* scikit. https://scikit-learn.org/stable/modules/generated/sklearn.metrics.silhouette_score.html

(8) Provost, F., & Fawcett, T. (2013). *Data Science for Business: What you need to know about data mining and data-analytic thinking*. O'Reilly.

Appendix B: Visualization of data

User data analysis

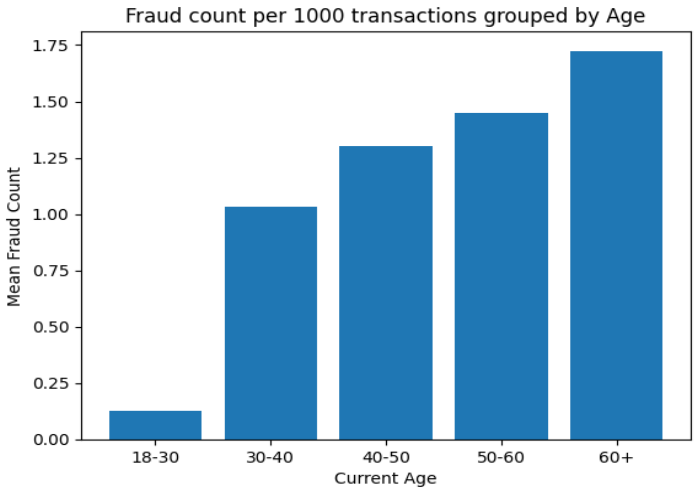

Figure 13

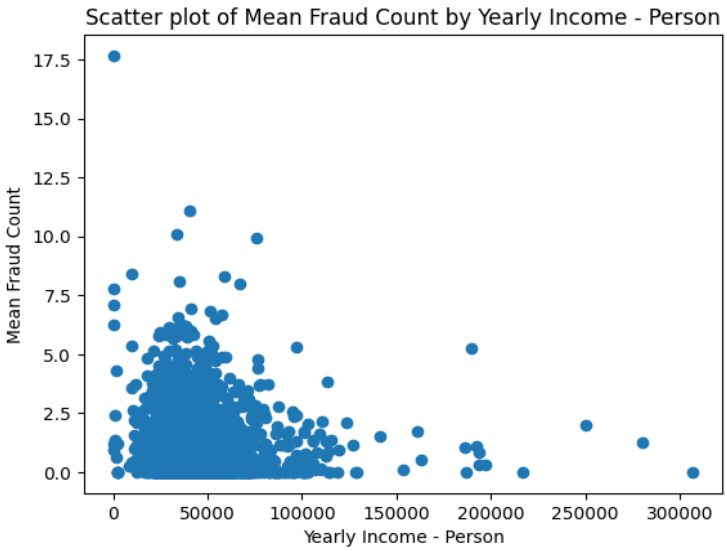

Figure 14

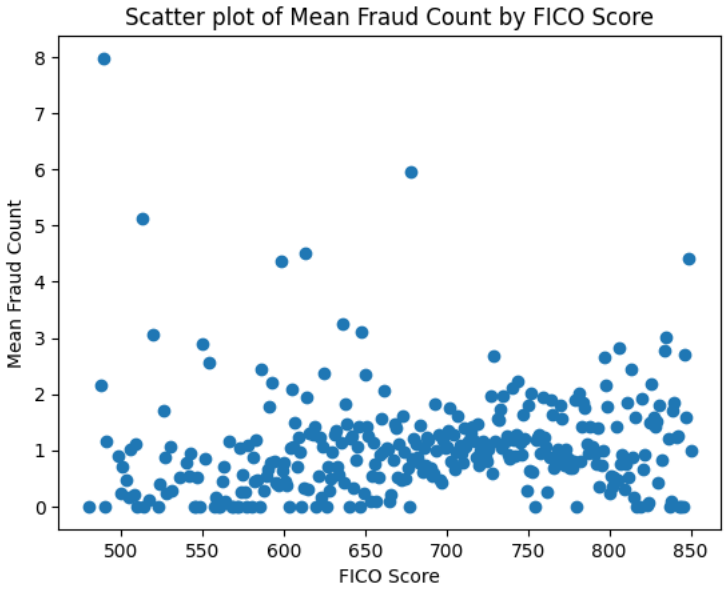

Figure 15

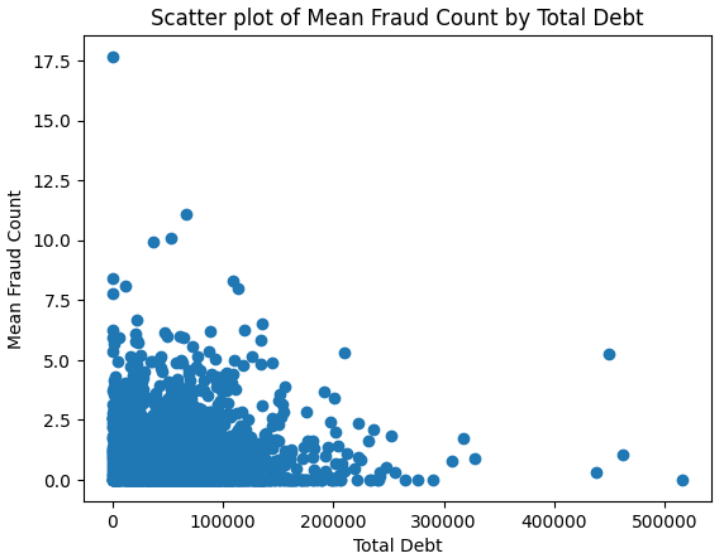

Figure 16

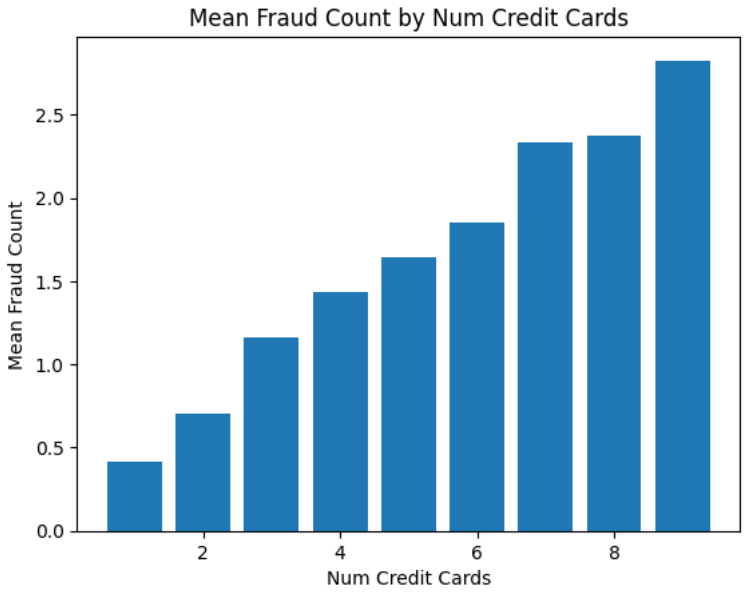
Mean Fraud Count by Num Credit Cards
Mean Fraud Count
0.0
0.5
1.0
1.5
2.0
2.5
2
4
6
8
Num Credit Cards

Figure 17

Cards data analysis

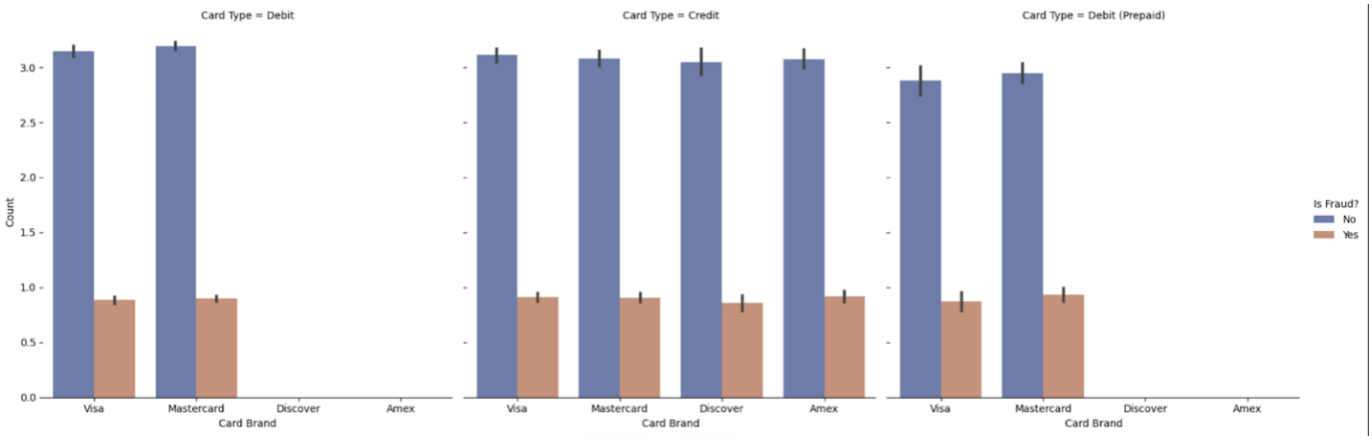

Figure 18

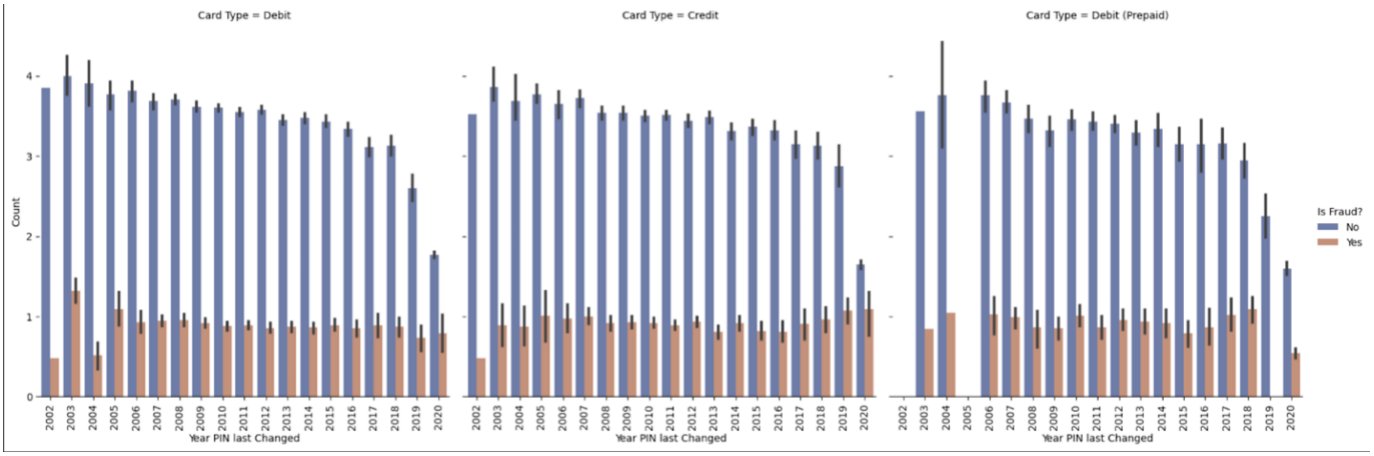

Figure 19

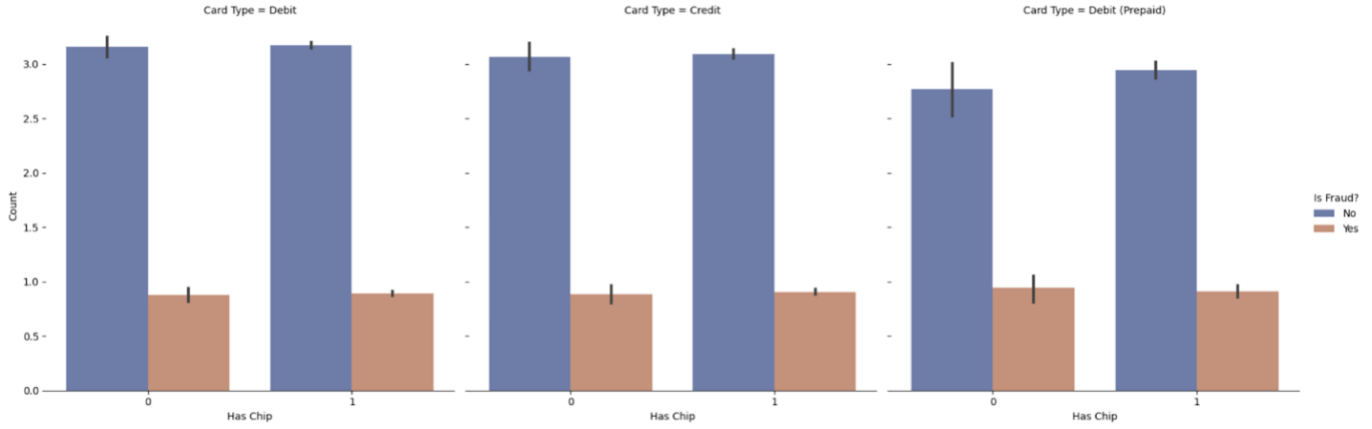

Figure 20

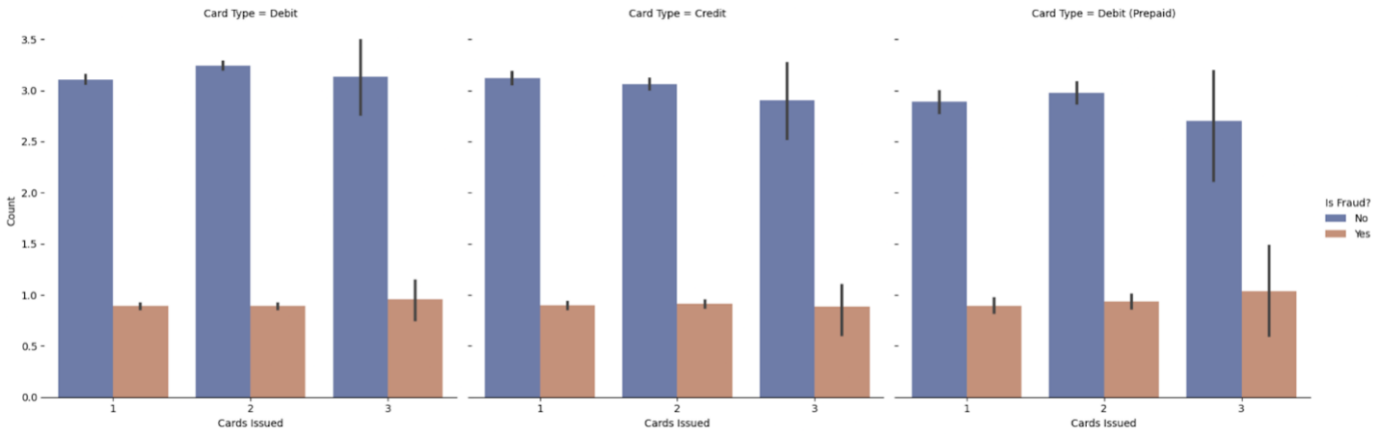

Figure 21

Transactions data analysis

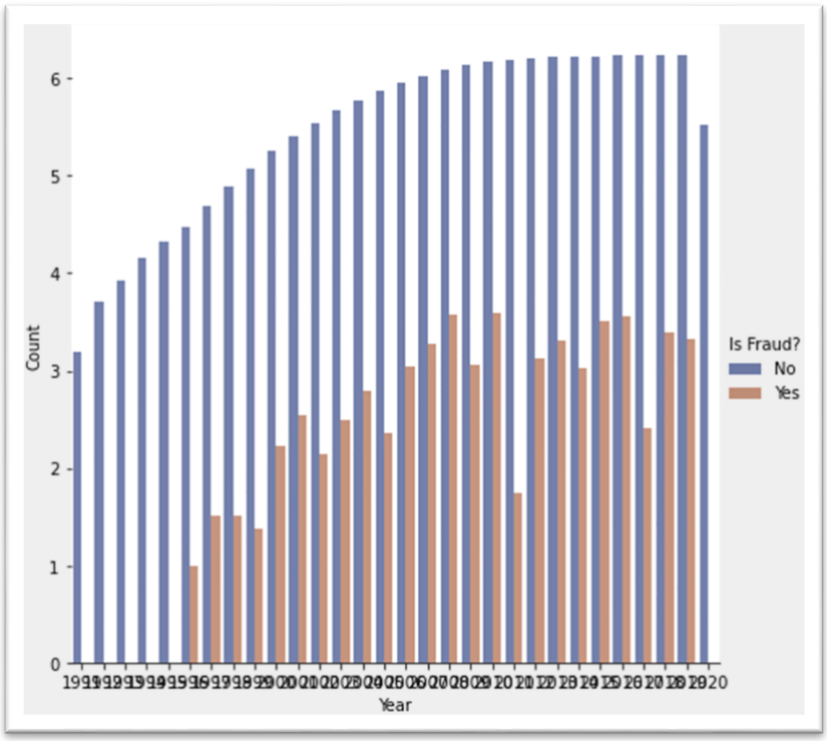

Figure 22

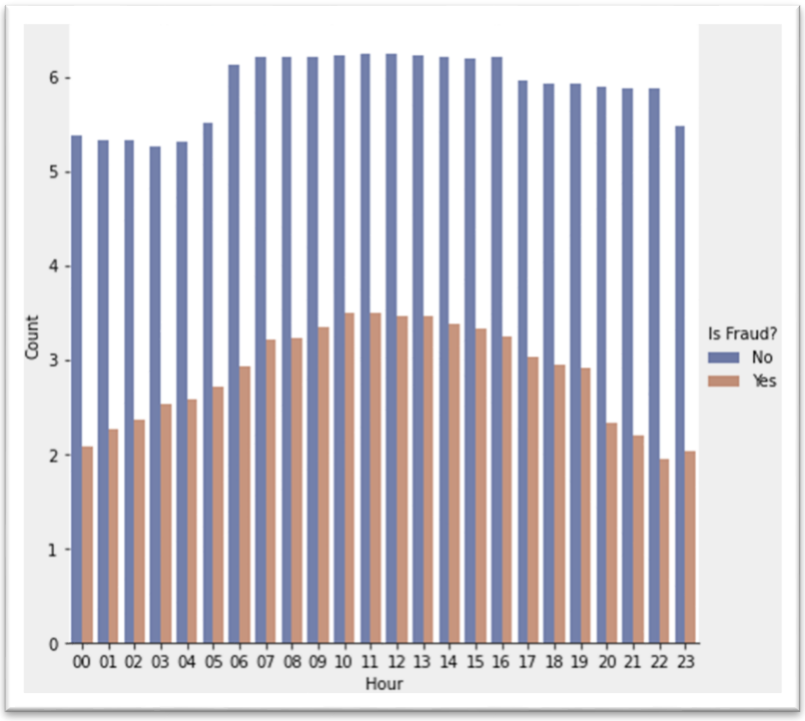

Figure 23

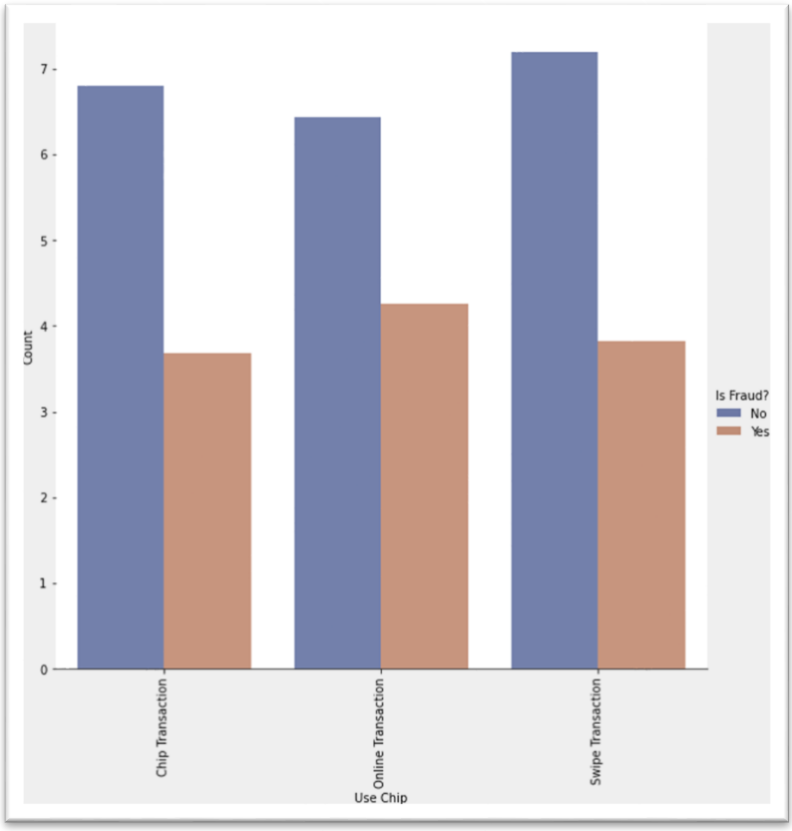

Figure 24

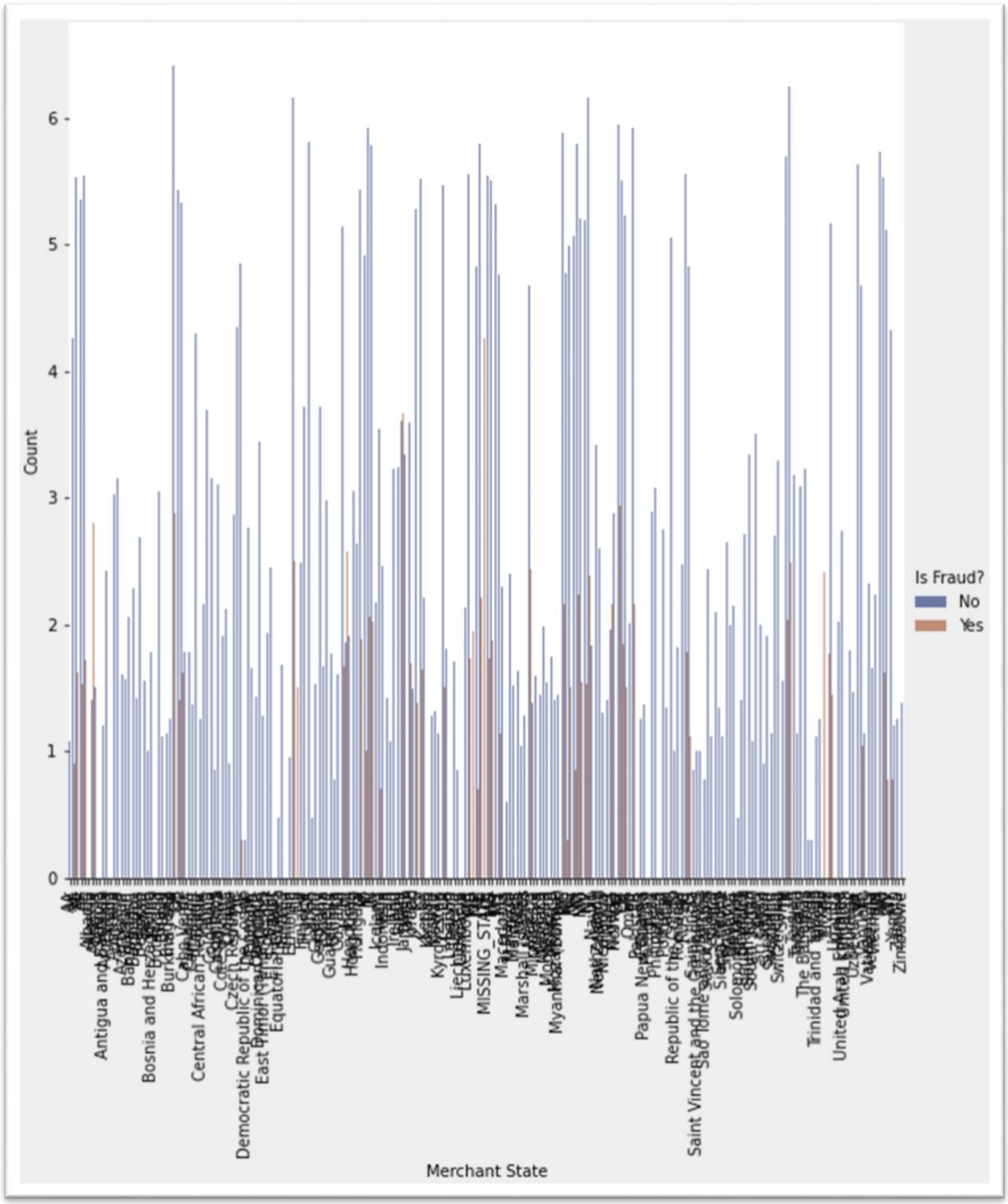

Figure 25

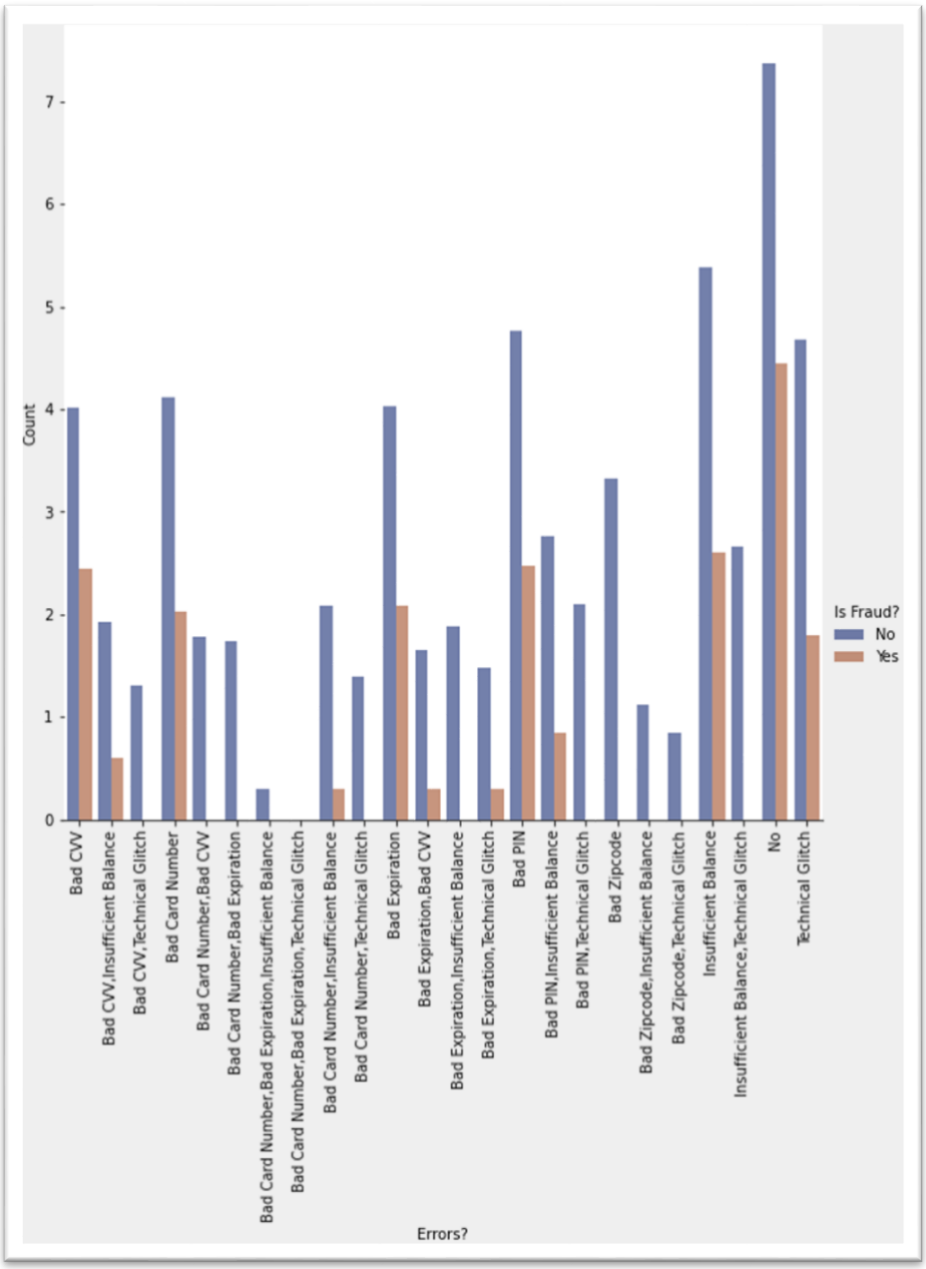
Count
Errors?
Is Fraud?
No
Yes
Bad CVV
Bad CVV,Insufficient Balance
Bad CVV,Technical Glitch
Bad Card Number
Bad Card Number,Bad CVV
Bad Card Number,Bad Expiration
Bad Card Number,Bad Expiration,Insufficient Balance
Bad Card Number,Bad Expiration,Technical Glitch
Bad Card Number,Insufficient Balance
Bad Card Number,Technical Glitch
Bad Expiration
Bad Expiration,Bad CVV
Bad Expiration,Insufficient Balance
Bad Expiration,Technical Glitch
Bad PIN
Bad PIN,Insufficient Balance
Bad PIN,Technical Glitch
Bad Zipcode
Bad Zipcode,Insufficient Balance
Bad Zipcode,Technical Glitch
Insufficient Balance
Insufficient Balance,Technical Glitch
No
Technical Glitch

Figure 26

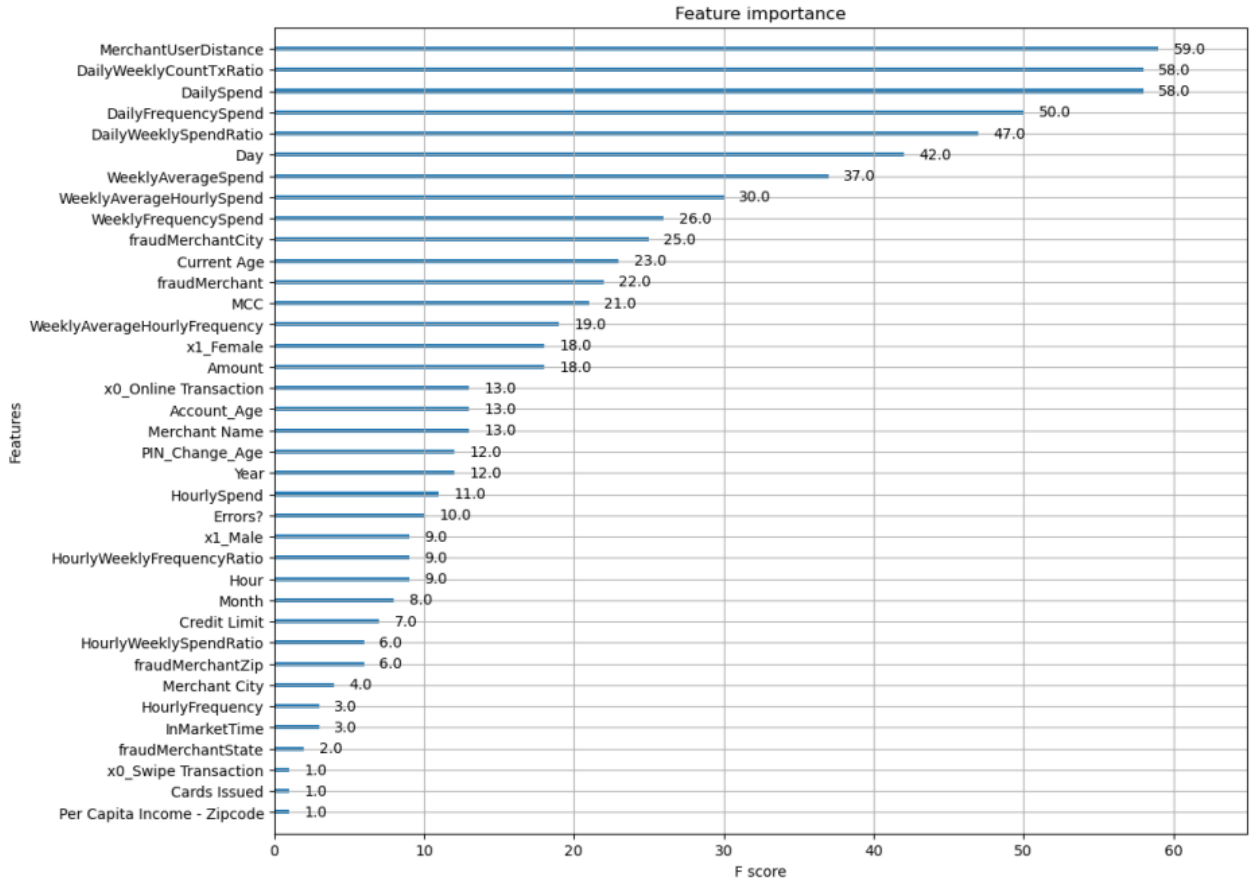
Feature importance
MerchantUserDistance 59.0
DailyWeeklyCountTxRatio 58.0
DailySpend 58.0
DailyFrequencySpend 50.0
DailyWeeklySpendRatio 47.0
Day 42.0
WeeklyAverageSpend 37.0
WeeklyAverageHourlySpend 30.0
WeeklyFrequencySpend 26.0
fraudMerchantCity 25.0
Current Age 23.0
fraudMerchant 22.0
MCC 21.0
WeeklyAverageHourlyFrequency 19.0
x1_Female 18.0
Amount 18.0
x0_Online Transaction 13.0
Account_Age 13.0
Merchant Name 13.0
PIN_Change_Age 12.0
Year 12.0
HourlySpend 11.0
Errors? 10.0
x1_Male 9.0
HourlyWeeklyFrequencyRatio 9.0
Hour 9.0
Month 8.0
Credit Limit 7.0
HourlyWeeklySpendRatio 6.0
fraudMerchantZip 6.0
Merchant City 4.0
HourlyFrequency 3.0
InMarketTime 3.0
fraudMerchantState 2.0
x0_Swipe Transaction 1.0
Cards Issued 1.0
Per Capita Income - Zipcode 1.0
Features
0
10
20
30
40
50
60
F score

Figure 27