Data Wrangling and EDA of Credit Card Transactions

Hongli Peng

April 8th, 2023

1 Introduction

This report presents a data wrangling and exploratory data analysis (EDA) of a credit card transaction dataset, which is the very first step in data analysis. The primary goal of this analysis is to make the data clean and to extract meaningful insights from the data to make credit card companies make better decisions and improve their performance, such as better performance in identifying fraudulent credit card transactions. These are also helpful to build predictive models based on more reliable and meaningful data. Throughout the analysis, we would treat the fraud status as our outcome variable.

2 Data Overview

The dataset is from "CreditCardFraud.csv" which simulates credit card transaction info resembling that of a financial institution's customers, and the description of features is shown in the table below. The target variable is "isFraud", which indicates whether a credit card transaction is fraudulent. The dataset has 29 variables and 786363 variables, but we only included 23 variables in Table 1 due to preliminary data quality checks in Part 3.

Table 1: Dataset Variables

Variable Name	Description
accountNumber	Unique identifier for the customer account
customerId	Unique identifier for the customer
creditLimit	Maximum amount of credit available
availableMoney	Amount of credit available at the time of the transaction
transactionDateTime	Date and time of the transaction
transactionAmount	Amount of the transaction
merchantName	Name of the merchant where the transaction took place
acqCountry	Country where the acquiring bank is located
merchantCountryCode	Country where the merchant is located
posEntryMode	Method used by the customer to enter their payment card information
posConditionCode	Condition of the point-of-sale terminal at the time of the transaction
merchantCategoryCode	Category of the merchant where the transaction took place
currentExpDate	Expiration date of the customer's payment card
accountOpenDate	Date the customer's account was opened
dateOfLastAddressChange	Date the customer's address was last updated
cardCVV	Three-digit CVV code on the back of the customer's payment card
enteredCVV	CVV code entered by the customer during the transaction
cardLast4Digits	Last four digits of the customer's payment card
transactionType	Type of transaction
currentBalance	Current balance on the customer's account
cardPresent	Whether the customer's payment card was present at the time of the transaction
expirationDateKeyInMatch	Whether the card's expiration date was entered correctly
isFraud	Whether the transaction was fraudulent

3 Data Cleaning and Preprocessing

In this section, we describe the data cleaning and preprocessing steps performed on the dataset to ensure data quality and consistency. These steps include:

1. Performing preliminary data quality checks such as identifying duplicated columns and columns with entirely missing data.
   We found that there are no duplicated columns and the columns with entirely missing data are "echoBuffer", "merchantCity", "merchantState", "merchantZip", "posOnPremises", and "recurringAuthInd", which were been removed from the dataset because they would not contribute any information to our data.

2. Detecting and handling outliers in numerical variables.
   By checking the description and types of the variables, the meaningful numerical variables to our target variable "isFraud" are "creditLimit", "availableMoney", "transactionAmount", and "currentBalance". From the plot 1, All histogram plots of four numerical variables showed a

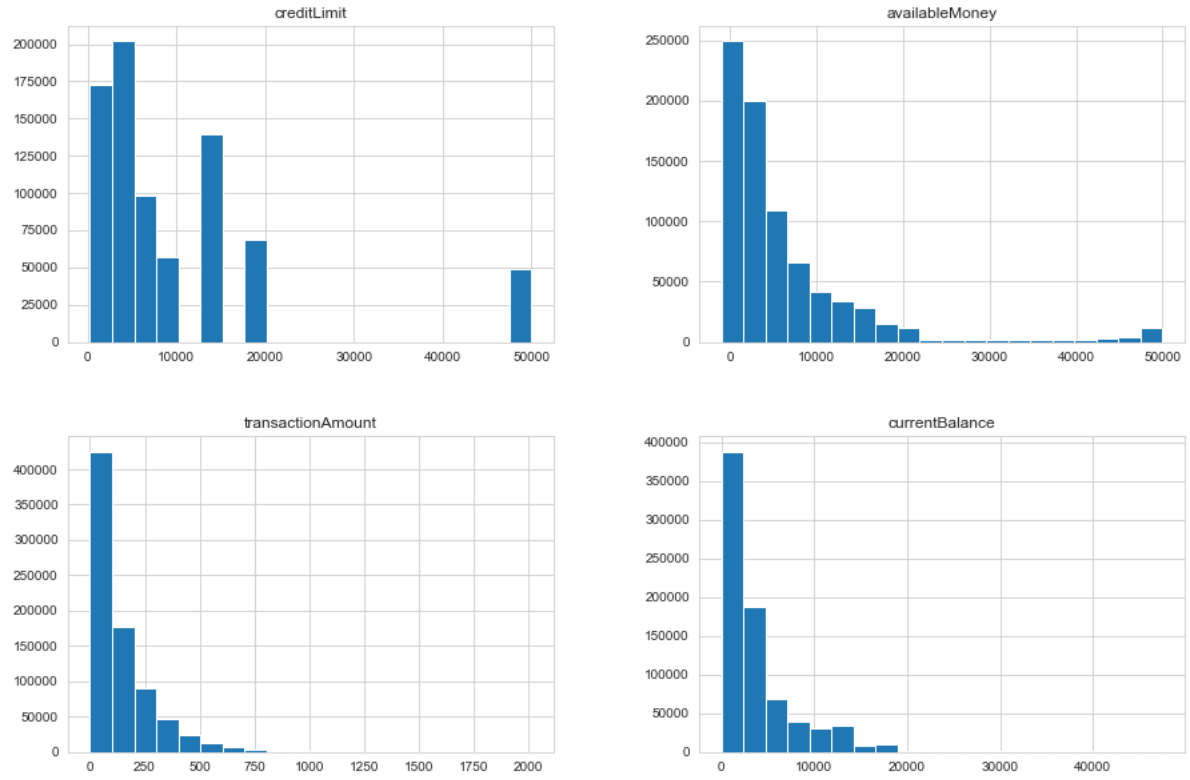

Figure 1: Histograms for Numerical variables

   tendency of right skewness, which suggests that there may exist outliers we should concern about. Also, the frequency has a decreasing trend as the unit of numerical variables increases, which means that it is possible for them not to follow a normal distribution. Hence, using The Inter-quartile Range (IQR) method to detect outliers is more appropriate. The outliers could be detected in the region outside of the interval [Q1-1.5*IQR, Q3+1.5*IQR], where Q1 is the first quantile, Q3 is the third quantile, and IQR is getting by subtracting Q1 from Q3. As for handling the outliers, we used the median of each numerical variable to make a replacement due to the property of the right skewness that the mean is biased to replace outliers.

3. Handling missing values in the dataset.
   In this part, we replaced the missing values for all numerical variables with a median. Moreover, we identified that there were 5 categorical variables having missing values, "acqCountry", "merchantCountryCode", "posEntryMode", "posConditionCode", and 'transactionType". All these

Table 2: Mean Fraud Rates by Missing vs. Non-Missing Values

Variable	**Missing**	**Non-Missing**
acqCountry	3.27%	1.57%
merchantCountryCode	11.33%	1.57%
posEntryMode	6.64%	1.55%
posConditionCode	5.38%	1.58%
transactionType	2.01%	1.58%

variables are categorical. From Table 2, for each variable, the non-missing values tend to have a higher fraud rate compared with missing values, especially for "merchantCountryCode" having a missing fraud rate of 11.33% contrasted with a non-missing one of 1.57%, which indicates that there is useful information for these missing values to detect the fraudulent transaction and they could be not missing at random. Therefore, We create a new category "unknown" to fill in missing values because this can help prevent biased or incomplete results that might occur if missing data were simply ignored, removed, or replaced by the mode from the analysis.

Moreover, we may consider dropping "acqCountry" as it appears about the same as "merchantCountryCode". And we found that they were only 67 observations that these two variables have different countries when we didn't include missing values on both variables.t Also, "merchantCountryCode" has a higher mean fraud rate for missing values while having much fewer missing values.

4. Ensuring consistency and usability of time variables.
   We observed that the time variables are "transactionDateTime", "accountOpenDate", and "dateOfLastAddressChange". Firstly, we transformed the time variables to a suitable datetime format for consistency. Secondly, there were no missing values or irregular timestamps after transforming. Thirdly, there were no transactions that happened before the account was opened. Fourthly, we extracted a new variable "days_since_acc_open" that means the number of days of a transaction since the account was created, which could be used to observe its relationship with fraud transactions. It was because we perceived that a small number of "days_since_acc_open" tends to have more fraudulent transactions.

5. Integrating unique characteristics of variables, such as cardCVV, enteredCVV, and cardLast4Digits, into the analysis.
   First of all, we created a new feature "cvv_match", which represents whether the entered CVV matches the actual CVV on the credit card. This information could be relevant for detecting fraudulent transactions, as fraudsters might tend to enter incorrect CVV codes multiple times. What's more, from Table 3, when CVV is not matched, it has a higher fraud rate of 2.89% than that of 1.57% when CVV is matched. Hence, the new feature "cvv_match" may provide meaningful information for fraudulent transactions.

Table 3: Mean Fraud Rates by cvv_match

cvv_match	**Mean Fraud Rate**
0(not matched)	2.89%
1(matched)	1.57%

Note that all the variables were transformed into the appropriate types. For example, categorical variables should be factored and numerical variables should be float or integer.

4 Exploratory Data Analysis

In this section, we present the results of our EDA, including visualizations and insights for various variables and their relationships with the outcome variable, "isFraud".

4.1 CVV Code and Fraud Transaction

In part 3, we established a new variable named "cvv_match" for examining the relationship between "cardCVV", "enteredCVV", "cardLast4Digits", and "isFraud'. The aim of this part is to use a grouped bar chart to analyze to the relationship. From Figure 2, the grouped bar chart showed the proportion of fraudulent transactions by CVV Match, where the x-axis represents whether or not the CVV code entered matches the true CVV code, and the y-axis represents the proportion of transactions that are fraudulent within each "CVV_match" category. The blue color indicates fraudulent transactions and the red color indicates non-fraudulent transactions. The grouped bar chart indicates that the

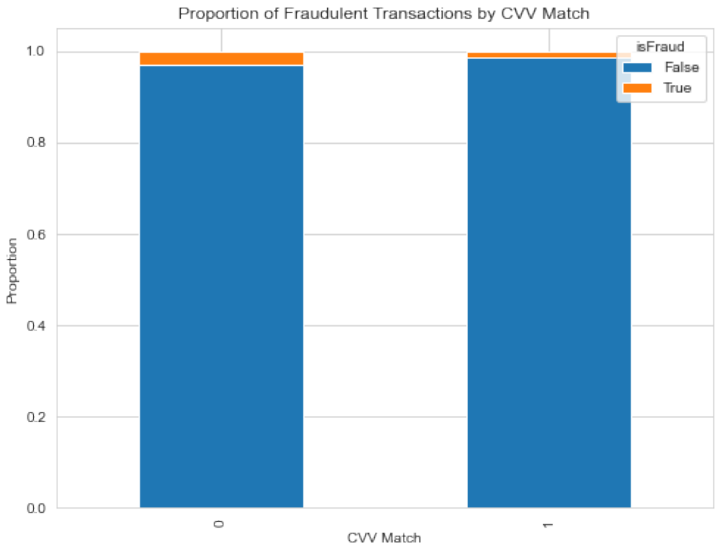

Figure 2: Proportion of Fraudulent Transactions by CVV Match

proportion of fraudulent transactions is higher when the CVV code does not match (0.0289) compared to when it does match (0.0157), which is consistent with Table 3. In other words, transactions, where the CVV code does not match, have a higher probability of being fraudulent.

Overall, these findings suggest that including "CVV_match" as a variable in predictive models for credit card fraud detection could help improve the accuracy and effectiveness of these models.

4.2 Transaction Amount Distribution

We visualized the distribution of "transactionAmount" using a histogram. This histogram plot from Figure 3 shows a right-skewed distribution, which means that small transaction amounts are more frequent than large transaction amounts.

This finding has several implications for credit card fraud detection. First, it suggests that there may be more fraudulent transactions happening at lower transaction amounts. One possible explanation for this is that people may not perceive small transaction amounts as significant, and therefore may be less likely to notice fraudulent transactions at these amounts. Additionally, people may make more transactions for small amounts, which could provide more opportunities for fraudsters to carry out fraudulent transactions.

Another reason that could contribute to more fraud at lower transaction amounts is that it may be safer for fraudsters to make smaller transactions. Smaller amounts are less likely to trigger fraud alerts or other security measures and therefore are drawn less attention.

Overall, these findings suggest that it may be important to pay extra attention to lower transaction amounts when detecting fraudulent transactions, and this could offer a more comprehensive predictive model.

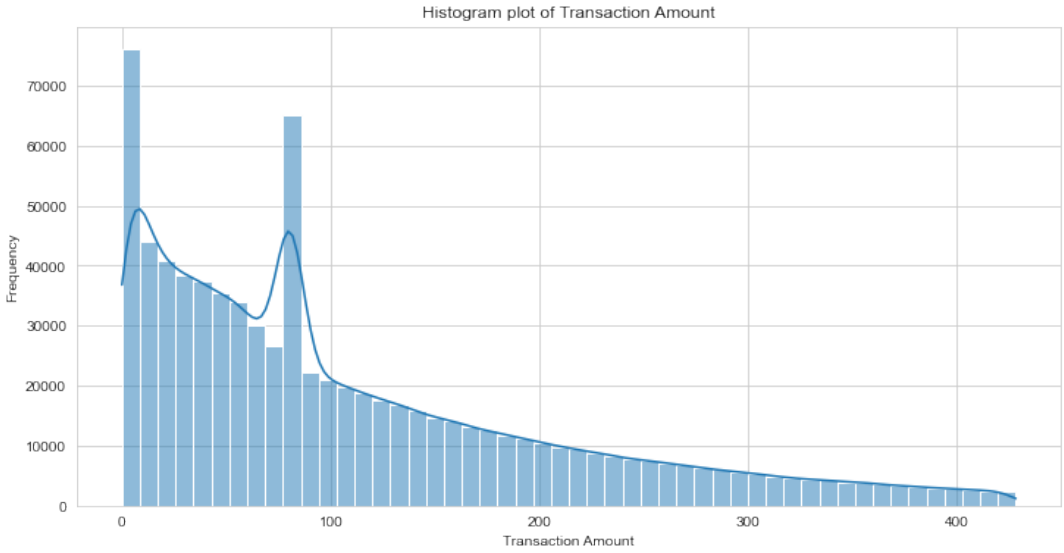

Figure 3: Transaction Amount Distribution Plot

4.3 Categorical Variables and Fraud

We investigated the relationship between "isFraud" and categorical predictors, such as "merchantCategoryCode", "posEntryMode", "transactionType", "posConditionCode", and "merchantCountryCode", by creating bar charts to display the fraud rate for each category. We aimed to understand how the fraud rate varies across different categories within each predictor. For "merchantCategoryCode", the airline has the highest fraud rate, which suggests that transactions related to airlines might be more susceptible to fraud than other types. For the rest of the categorical predictors, the new variable "unknown" we created for replacing missing values has the highest fraud rate, which implies that transactions with missing information in these predictors may be more likely to be fraudulent.

Therefore, we should incorporate these categories within the specific predictors to provide a more accurate predictive model for detecting fraudulent transactions.

4.4 Fraud Rate by Merchant Category Code and Transaction Type

In this section, we explore the relationship between "isFraud" and "transactionType" conditioned on "merchantCategoryCode" by generating a grouped bar chart to display the fraudulent rates by merchant category code and transaction type. From Figure 5, for all types of merchant category codes, there is a tendency for transaction types PURCHASE and REVERSAL to have a higher fraud rate than ADDRESS_VERIFICATION. For merchant category code AIRLINE, ENTERTAINMENT, FASTFOOD, and ONLINE GIFTS, the UNKNOWN transaction type exhibits the highest fraud rate.

As a result, we could include the interaction between "transactionType" and "merchantCategoryCode" in the predictive model. It is because this interaction could potentially improve its performance by capturing more complex patterns related to fraud transactions.

4.5 Conditional Probability Density Plots For the Numerical Variables

In this section, we constructed the conditional plots to visually compare the distributions of each numerical variable ("creditLimit", "availableMoney", "transactionAmount", "currentBalance") for fraud and non-fraud transactions.

For all four plots from Figure 6, the distribution of non-fraud transactions tends to have a higher density for each numerical variable than fraud transactions, indicating that most transactions in these

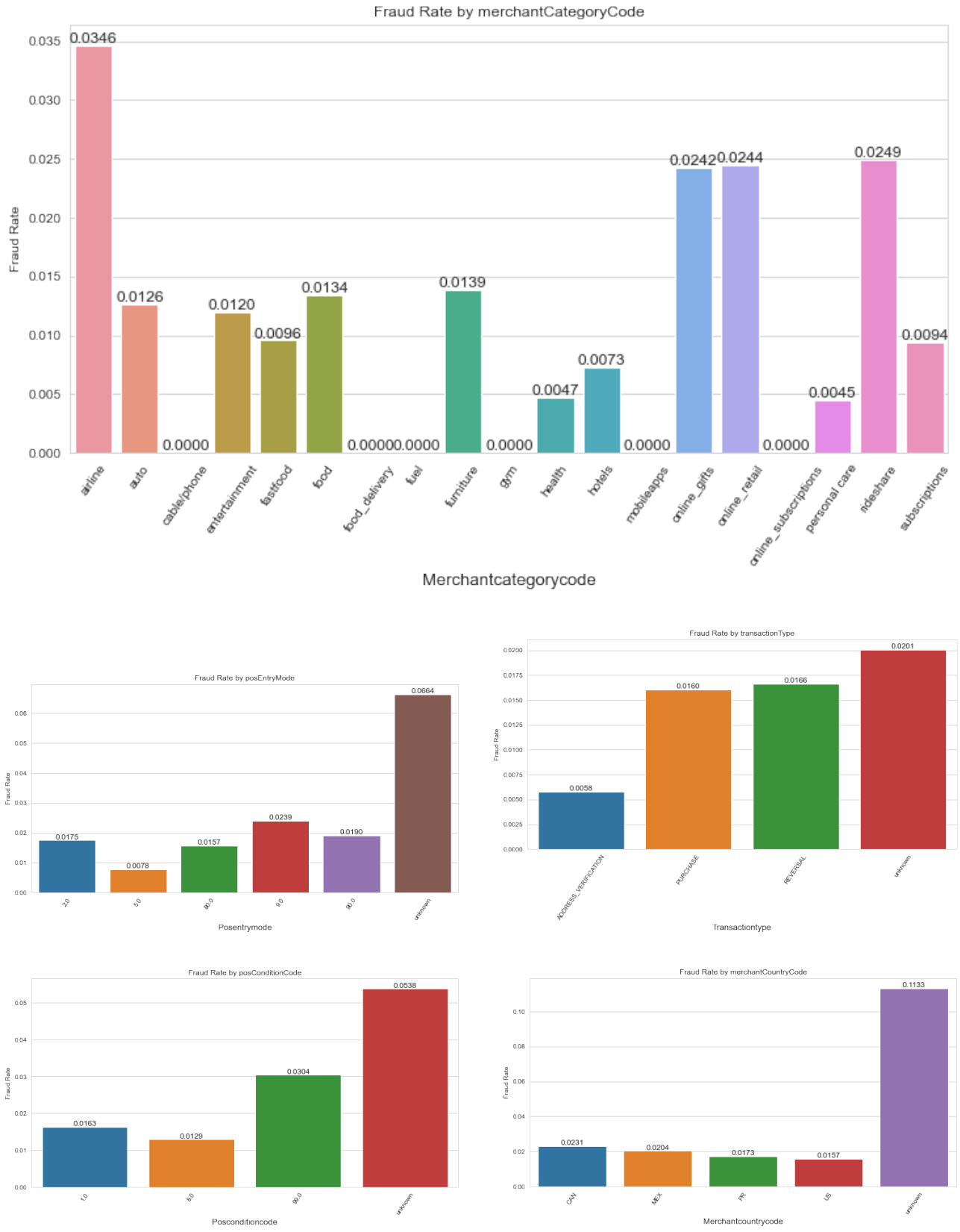

Figure 4: Caption for the entire group of figures

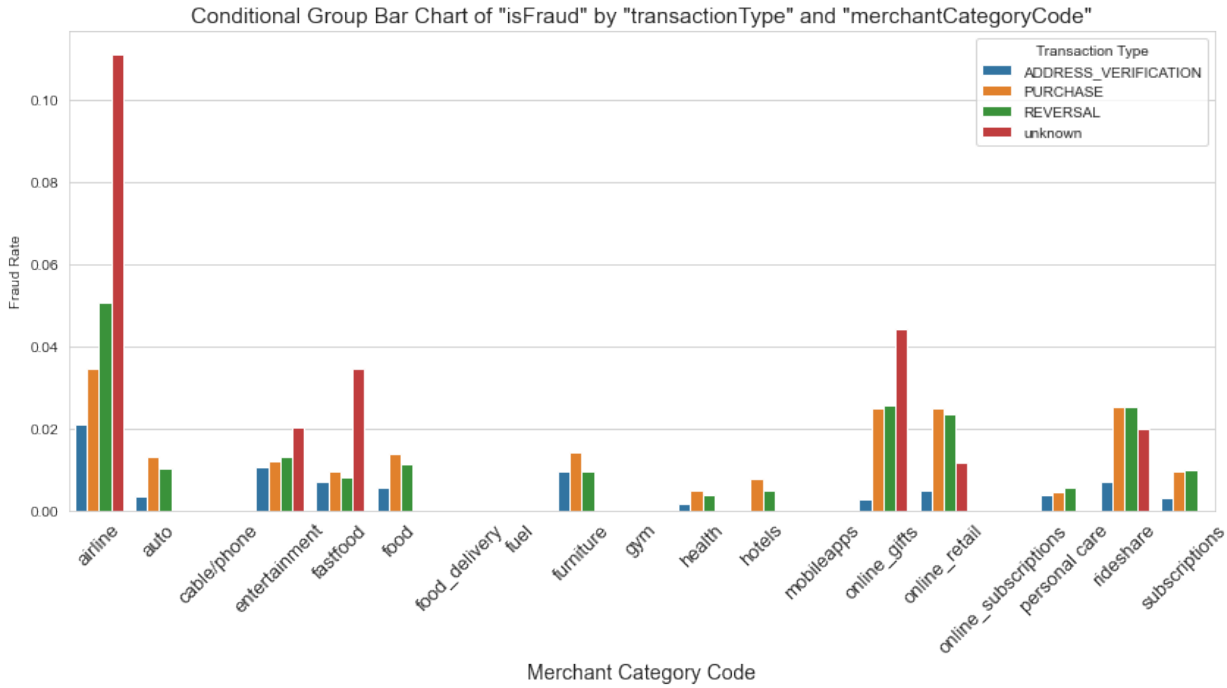

Figure 5: Group Bar Chart of "isFraud" by "transactionType" and "merchantCategoryCode"

variables are not fraudulent. All plots exhibit right-skewed distributions for both non-fraud and fraud transactions, suggesting that there is a tendency for more transactions to occur at lower amounts of credit limit, available money, transaction amounts, and current balance.

In the "transactionAmount" plot, the distribution for fraudulent transactions appears to be more concentrated at larger values compared to non-fraudulent transactions. This finding indicates that fraudulent transactions tend to involve more substantial amounts of money when the transaction amount is greater than 100. These insights into the distributions of numerical variables for fraudulent and non-fraudulent transactions can help for establishing a predictive model.

4.6 Multi-swipe Transactions

In this section, we focused on identifying multi-swipe transactions and assessing their potential impact on fraudulent transactions. We defined multi-swipe transactions based on the following conditions:

- The transaction has the same account number as the previous transaction.
- The transaction amount is the same as the previous transaction.
- The time difference between the current and previous transactions is less than or equal to 15 minutes.
- The merchant name is the same as the previous transaction.
- Transaction types should be the same as the previous transaction, and only transactions with types 'unknown' and 'purchase' could be considered multi-swipe transactions.

The total number of multi-swipe transactions we discovered is 7,341, representing 0.93% of all transactions. And the total dollar amount for multi-swipe transactions is 884,421.74, representing 1.01% of the total dollar amount of all transactions. Even though multi-swipe transactions are less frequent, we still need to pay attention to them since these transactions may involve larger transaction amounts, which may yield a large impact on companies and customers.

Finally, we considered adding a binary class to our dataset to represent multi-swipe transactions. This additional feature may contribute valuable information for building a predictive model to detect and prevent credit card fraud transactions.

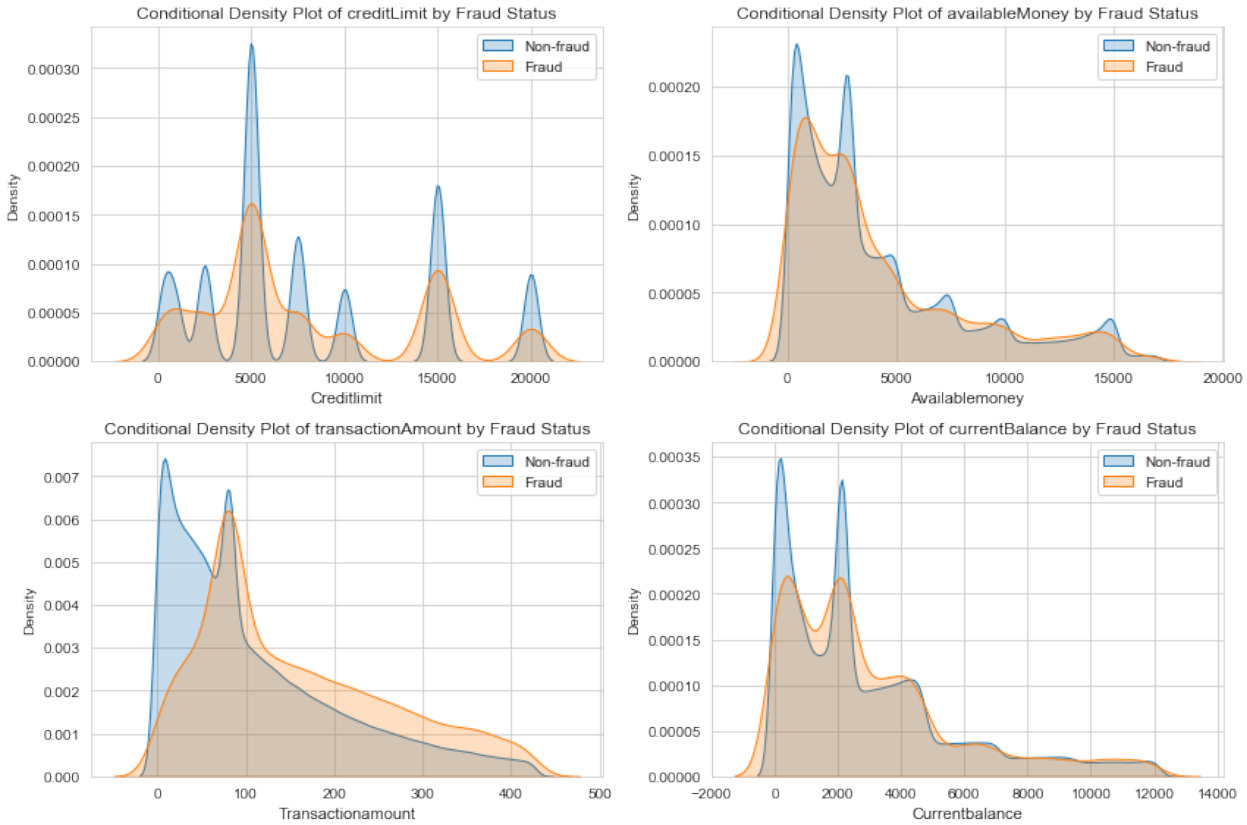

Figure 6: Conditional Density Plot of Numerical Variables by Fraud Status

5 Class Imbalance

We examined the class imbalance in the "isFraud" outcome variable. By calculating the percentage of the fraud rate in the data, there are 98.42% of non-fraudulent transactions and 1.58% of fraudulent transactions, which is unbalanced. This class imbalance can lead to biased predictive models that may prioritize detecting non-fraudulent transactions, resulting in a high number of false negatives for fraud cases, which means that the predictive model incorrectly predicts that a transaction is not fraudulent, when in fact it is. This could result in the model overlooking real fraud cases. To address this issue, techniques such as random re-sampling, synthetic data generation like SMOTE, and ensembling methods such as random forest could be applied. For the sake of simplicity, we used a random sampling method.

We performed in two ways. The first one is the random under-sampling method, which aims to balance the dataset by reducing the instances of the majority class(non-fraud cases) to match the number of instances in the minority class(fraud cases). This is done by randomly selecting and discarding instances from the majority class until the desired balance is achieved. The second one is the random over-sampling method, which is to balance the dataset by increasing the instances of the minority class to match the number of instances in the majority class. This is done by randomly selecting instances from the minority class and duplicating them until the desired balance is achieved. Finally, we derived 12,417 for both fraud and non-fraud transactions for the under-sampling method. For the over-sampling method, we get 773,946 for both cases. When using these two methods, we should realize that even though the under-sampling method is computationally efficient, it may discard potentially useful information from the majority class and may cause over-fitting. Also, though the over-sampling method retains all the information from the majority class, it can lead to over-fitting since the same minority class samples are replicated multiple times to balance the dataset.

In order to achieve a balanced dataset while avoiding overfitting risks, a combination of under-sampling the majority class and over-sampling the minority class with a synthetic data generation method like SMOTE can be used. This approach retains more useful information while maintaining

a balanced representation of both classes. What's more, when assessing the model performance, we should apply more appropriate evaluation metrics, such as precision, recall, F1-score, or ROC-AUC curve, since the accuracy may be biased.

6 Conclusion

In this report, we performed exploratory data analysis and data wrangling of a credit card transaction dataset to make the dataset well-prepared for a predictive model and gain insights relationship between predictors and fraud transactions.

It is important to note that the data wrangling and EDA presented here are only initial steps in the process of building a predictive model for credit card fraud detection. Further steps would include feature engineering(e.g., dimensionality reduction), model selection(e.g., tree models and neural networks), validation, training, and testing to ensure optimal performance in detecting fraudulent transactions.