ANL588 FINAL REPORT

Detecting Credit Card Fraud in Transactional Data

Submitted by Chua Wai Jie

SCHOOL OF BUSINESS
Singapore University of Social Sciences

Presented to Singapore University of Social Sciences in partial fulfillment of the requirements for the Degree of Master of Analytics and Visualisation

2025

Abstract

In today's increasingly digital economy, credit card fraud remains a problem for banks and merchants and customers. To reduce financial losses due to fraud and identify fraud patterns, this project applied data analytics and machine learning to detect credit card fraud patterns using a large transaction dataset from Kaggle. The dataset was divided into online and offline transactions before analysis and modelling separately. Seven binary classification models (logistic regression, decision tree, random forest, neural networks, gradient boosting, SVC, and KNN) were built and evaluated using 5-fold cross-validation, with random forest emerging as the best model for both datasets based on the F1-score metric. After hyperparameter and threshold tuning, the champion models achieved F1-scores of 0.8499 (offline transactions dataset) and 0.8931 (online transactions dataset), surpassing the project's F1-score target of 0.80. The models also demonstrated high accuracy (0.9997 and 0.9988), precision (0.9320 and 0.9837), and recall (0.7811 and 0.8178) for offline and online transactions, respectively. Feature importance analysis of these models revealed that merchant location was critical for offline fraud detection, while merchant id and category were the most important features for online fraud.

Business impact, based on estimated costs of undetected fraud and false positives, were substantial, saving 68.54% of fraud related costs for offline transactions and 83.37% of fraud related costs for online transactions annually for a sample of 2000 customers. While the project achieved its objectives, limitations include the absence of behavioural analysis and potential bias from under sampling. Recommendations for deployment included integration with bank's systems and periodic retraining. Future work could explore hyperparameter tuning of other models, incorporate explainable AI techniques, and address regulatory compliance requirements.

Table of Contents

Chapter One – Introduction

In today's increasingly cashless society, 48% of Americans in 2024 are projected to no longer use cash for payments in daily life, and this figure is expected to rise to 51.6% in 2025 (Capital One Shopping, 2024).

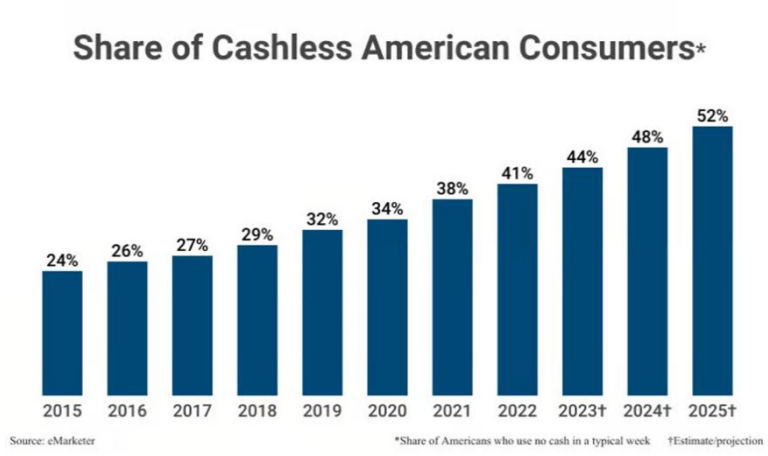


Figure 1: Share of American consumers going cashless between the years 2015 to 2025. Values for 2023 to 2025 are projected values. (Capital One Shopping, 2024)

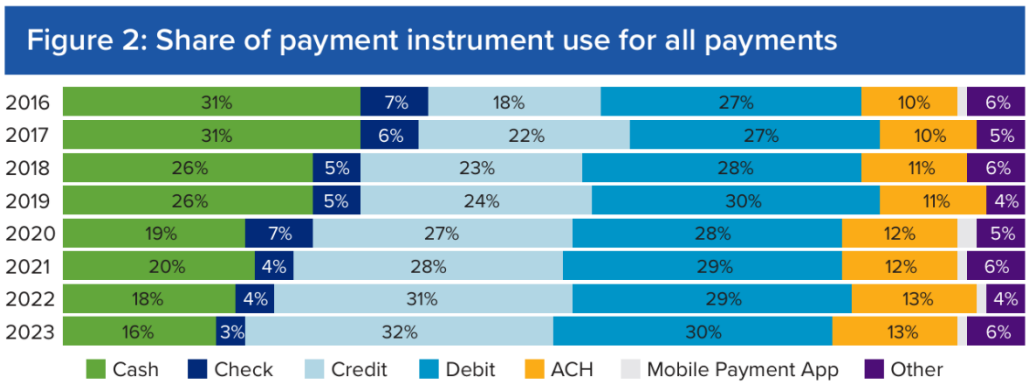


Figure 2: Share of payment instrument use by U.S. consumers for each payment method between the years 2016 – 2023 (Frbservices.org, 2024).

This trend is supported by the findings from the Federal Reserve's 2024 Diary of Consumer Payment Choice report which show a decline in the share of payments made with cash over the years from 2016 to 2023. The findings also show that credit cards are the most popular form of payment among Americans, with 32% of payments in 2023 being made with them. This is an increase from the 27% of payments in 2020.

However, the increasing reliance on credit cards has also brought about an increase in the number of fraudulent credit card transactions. Credit card fraud occurs when criminals make use of stolen credit cards or credit card information to make unauthorised purchases. According to the Nilson Report (2025), payment card fraud losses increased to $33.83 billion worldwide in 2023 from $33.45 billion in 2022. The report projects that losses from credit card fraud worldwide will sum up to $403.88 billion over the next decade, with the United States accounting for 42.32% of these losses.

1.1 Business Problem

Despite various measures to prevent fraud, credit card fraud remains a challenge for card issuers, merchants, and payment processors. To combat fraud, traditional methods rely on rule-based systems or manual analysis by human experts (Martínez, Forradellas, Gallastegui, & Alonso, 2025). However, these methods suffer from several drawbacks. Rule-based systems struggle to adapt to new fraud patterns and are less effective when dealing with evolving fraud patterns, while human analysis is prone to fatigue and subjectivity, limiting both its scalability and accuracy (Olushola & Mart, 2024). The resulting undetected fraud cases directly translates into financial losses for card issuers, merchants and to a lesser extent, cardholders.

To address these limitations, a machine learning based fraud detection system is proposed in this project to offer a more adaptable, scalable and performant solution. Machine learning

models have the potential to achieve high accuracy, adapt to new data and scale up to process millions of transactions, further enhancing the effectiveness of fraud detection systems.

1.2 Objectives

This project aims to:

- Explore and identify fraud patterns based on the transaction details, merchant location, merchant categories and demographic factors using a publicly available dataset from the Kaggle platform.
- Construct and assess several machine learning models for credit card fraud detection, including logistic regression, decision trees, random forest, K-nearest neighbours (KNN) and neural networks. These machine learning models will be evaluated based on the F1 score to determine the most suitable model for credit card fraud detection in a transactional dataset.
- Target an F1 score of 80% for the final model, which would substantially reduce financial losses due to fraud.

Chapter Two - Background

2.1 Literature Review

The use of machine learning for fraud detection has gained traction in recent years, with various researchers making use of machine learning for detecting fraud in datasets such as credit card transactions, bank transactions and mobile payment transactions (Hernandez Aros, Bustamante Molano, Gutierrez-Portela, Moreno Hernandez, & Rodríguez Barrero , 2024). According to their comprehensive review on the application of machine learning in fraud detection,

supervised machine learning is the most common approach utilised by 56.73% of articles reviewed. In the review, the random forest model was found to be the most popular model with 34 mentions. This was closely followed closely by logistic regression with 32 mentions, decision tree with 29 mentions and support vector machine (SVM) with 29 mentions, indicating their widespread use in fraud detection.

In terms of performance, machine learning models are also able to achieve excellent results in predicting fraudulent transactions. Afriyie et al. (2023) built a decision tree, random forest and logistic regression model and compared their performance on a simulated credit card transactions dataset. They were able to achieve an accuracy of 0.96 and recall of 0.97 using their best random forest model. However, the precision of their models remained low, with the best model only achieving a precision of 0.09 and F1- score of 0.17.

One key challenge with fraud detection is the data imbalance commonly found in credit card fraud datasets, which significantly impact the performance of the machine learning models. To overcome this dataset imbalance, Afriyie et al. (2023) made use of under sampling to reduce the majority class of legitimate transactions to equal the minority class of fraudulent transactions. An alternative way to balance the dataset would be to use over sampling techniques. Khalid et al. (2024) compared various machine learning models using the under-sampling and Synthetic Minority Over-sampling Technique (SMOTE) and found that using SMOTE leads to superior performance in terms of both precision and recall. Their ensemble model also outperformed individual classifiers in predicting credit card fraud, achieving 99.95% precision, recall, and accuracy when combined with SMOTE.

Another challenge with using machine learning models for fraud detection is the selection of a threshold to decide how much potential fraud to block. Frequent false positives may cause

inconvenience to customers and lead to a drop in customer satisfaction, causing customers to seek alternatives (Lindemulder & Kosinski, 2024). On the other hand, false negatives can lead to losses from undetected fraud. Hence, the ideal model depends on the specific costs associated with false positives and false negatives for the business (Madarshahian, 2024).

2.2 Motivations for the project

- Analysing the transactions dataset, new patterns and anomalies in merchant details and user demographics that indicate fraud may be revealed, providing fresh insights to combat fraud.

- Completion of the fraud detection system would help to reduce financial and reputational losses to card issuers and merchants due to credit card fraud.

- Fraudulent transactions can be detected, and users can be notified of suspected fraudulent transactions promptly. This can lead to improved customer satisfaction for the card issuers.

Chapter Three - Data

3.1 Data Acquisition

The transactions dataset consisting of credit card transactions, customer and merchant details, as well as fraud labels was sourced from publicly available datasets on the Kaggle platform. The raw datasets were uploaded by a Kaggle user named "ComputingVictor", and can be accessed via the link Financial Transactions Dataset: Analytics.

The data source consists of five separate datasets with different file formats as follows:

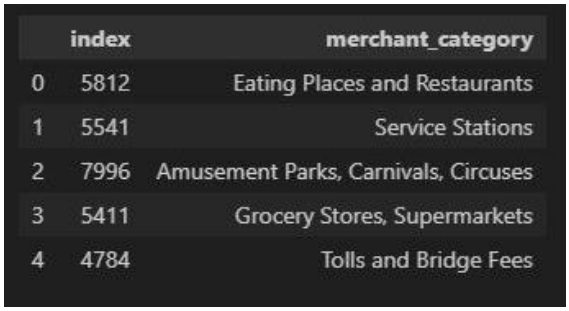
1. Transaction Data (transactions_data.csv) – The raw dataset consists of 13.3 million rows of transactions from all fifty U.S. states and 147 countries between the years 2010 and 2019. It contains details of transactions including amounts, timestamps, and merchant details such as their location and mcc code.

	id	date	client_id	card_id	amount	use_chip	merchant_id	merchant_city	merchant_state	zip	mcc	errors
0	7475327	2010-01-01 00:01:00	1556	2972	$-77.00	Swipe Transaction	59935	Beulah	ND	58523.0	5499	NaN
1	7475328	2010-01-01 00:02:00	561	4575	$14.57	Swipe Transaction	67570	Bettendorf	IA	52722.0	5311	NaN
2	7475329	2010-01-01 00:02:00	1129	102	$80.00	Swipe Transaction	27092	Vista	CA	92084.0	4829	NaN
3	7475331	2010-01-01 00:05:00	430	2860	$200.00	Swipe Transaction	27092	Crown Point	IN	46307.0	4829	NaN
4	7475332	2010-01-01 00:06:00	848	3915	$46.41	Swipe Transaction	13051	Harwood	MD	20776.0	5813	NaN

Figure 3 : Sample the of first five rows in the transactions dataset, showing transaction and merchant details.

2. Card Information (cards_dat.csv) – This dataset contains card details such as the brand of card, card limits, activation dates. It will not be utilised for this project as the data was found to be less relevant.

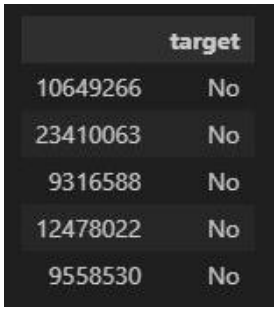
3. Merchant Category Codes (mcc_codes.json) – This dataset contains the category labels for industry-standard merchant category codes (mcc codes).



	index	merchant_category
0	5812	Eating Places and Restaurants
1	5541	Service Stations
2	7996	Amusement Parks, Carnivals, Circuses
3	5411	Grocery Stores, Supermarkets
4	4784	Tolls and Bridge Fees

Figure 4 : Sample of the first five rows in the merchant category codes dataset.

4. Fraud Labels (train_fraud_labels.json) – This dataset contains binary labels to indicate if the transaction was fraud or legitimate.



	target
10649266	No
23410063	No
9316588	No
12478022	No
9558530	No

Figure 5 : Sample of the first five rows in the fraud labels dataset.

5. User Data (users_data) – This dataset contains demographic information about 2000 customers, including details such as their age, gender, location, and financial status.

id	current_age	retirement_age	birth_year	birth_month	gender	address	latitude	longitude	per_capita_income	yearly_income	total_debt	credit_score	num_credit_cards
825	53	66	1966	11	Female	462 Rose Lane	34.15	-117.76	$29278	$59696	$127613	787	5
1746	53	68	1966	12	Female	3606 Federal Boulevard	40.76	-73.74	$37891	$77254	$191349	701	5
1718	81	67	1938	11	Female	766 Third Drive	34.02	-117.89	$22681	$33483	$196	698	5
708	63	63	1957	1	Female	3 Madison Street	40.71	-73.99	$163145	$249925	$202328	722	4
1164	43	70	1976	9	Male	9620 Valley Stream Drive	37.76	-122.44	$53797	$109687	$183855	675	1

Figure 6 : Sample of the first five rows in the users dataset.

3.2 Data Wrangling

Data wrangling was performed using the Python programming language in a Jupyter notebook environment. Since the number of rows was exceptionally large, a subset of the transactions dataset between the years 2015 and 2019 was created. This would reduce the volume of data to 6.73 million rows and shift the focus to only the more recent fraud patterns.

Figure 7: Code snippet demonstrating the creation of a subset of transactions data between the years 2015 to 2019.

Since the merchant_state column contained both country names and individual U.S. states, the data from the individual U.S. states was also aggregated into a single country (USA), to keep the merchant_state column consistent for all countries. Subsequently, the dataset was filtered

by only the top five countries that have the largest number of transactions, based on the merchant_state column. These countries are “USA”, “Mexico”, “Italy”, “Canada”, and “United Kingdom”. The online transactions without country information were also retained, leaving the filtered dataset with about 6.71 million rows.

All variables from the transaction dataset except “zip” were kept, while all variables except “address”, “latitude”, “longitude” ,“birth_year”, and “birth_month” from the users dataset were chosen for use in this project. The transactions, fraud labels, merchant category codes and user data datasets were then merged into a single transaction dataset to enable a more detailed analysis of the problem. An overview of the data wrangling process can be found below in figure 8.

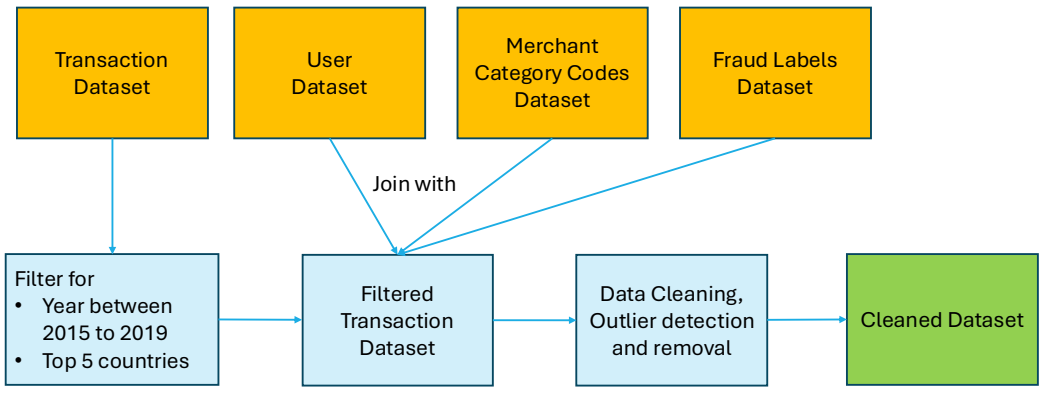


Figure 8: Workflow of the data wrangling process from the raw datasets to the cleaned dataset.

After filtering, an inner join of the transactions dataset with the fraud labels was performed to include the fraud labels in the transactional data. As a number of transactions do not have fraud labels, an inner join was used to keep only rows that have a matching fraud label. This results in a transaction dataset of 4.49 million rows with the new target column, as seen in the figure below.

```
# Data Wrangling 1 - Joining the fraud labels to the transactions dataset based on the transaction id.

df = transactions.merge(fraud_labels, left_on='id', right_index=True, how='inner')
df.head()
```

✓ 5.8s

	id	date	client_id	card_id	amount	use_chip	merchant_id	merchant_city	merchant_state	zip	mcc	errors	target
0	7475327	2010-01-01 00:01:00	1556	2972	$-77.00	Swipe Transaction	59935	Beulah	ND	58523.0	5499	NaN	No
1	7475328	2010-01-01 00:02:00	561	4575	$14.57	Swipe Transaction	67570	Bettendorf	IA	52722.0	5311	NaN	No
2	7475329	2010-01-01 00:02:00	1129	102	$80.00	Swipe Transaction	27092	Vista	CA	92084.0	4829	NaN	No
4	7475332	2010-01-01 00:06:00	848	3915	$46.41	Swipe Transaction	13051	Harwood	MD	20776.0	5813	NaN	No
5	7475333	2010-01-01 00:07:00	1807	165	$4.81	Swipe Transaction	20519	Bronx	NY	10464.0	5942	NaN	No

Figure 9: Sample rows of the transaction dataset after performing an inner join with the fraud labels dataset.

Secondly, the user data was joined to the transactional dataset with fraud labels based on the client identifier. This was done using a left join to the transactional dataset in the previous step to ensure that transaction data rows are retained. No missing user data was found for any of the transactions after the join was executed. The combined transactions dataset includes client demographic data such as gender, geographic location, and financial information. The new columns are shown in figure 10 below.

```
# Data Wrangling 2 - Joining the user details to the transactions dataset based on the client id.

df = df.merge(user_details, left_on='client_id', right_on='id', how='left')
df.head()
```

✓ 0.6s Python

chant_state	zip	...	retirement_age	birth_month	gender	latitude	longitude	per_capita_income	yearly_income	total_debt	credit_score	num_credit_cards
FL	32222.0	...	69	11	Female	30.33	-81.65	$20710	$42229	$0	688	3
CA	94577.0	...	65	11	Male	37.71	-122.16	$24971	$30962	$15336	743	5
NaN	NaN	...	66	2	Female	45.45	-122.79	$23687	$48295	$109558	707	3
NaN	NaN	...	65	11	Male	36.34	-83.28	$13668	$27861	$108313	782	5
OH	43228.0	...	68	11	Male	39.98	-82.98	$18420	$37556	$0	735	6

Figure 10: Sample rows of the transaction dataset after performing a left join with the user dataset.

The merchant category code dataset was also joined with the resulting dataset to map the category codes (mcc) into their corresponding text labels. For example, the merchant category

code 5499 corresponds to “Miscellaneous Food Stores” which are more interpretable compared to the numeric codes (Figure 11). The addition of the merchant category did not result in any changes to the number of observations, which remain at 4,498,909.

```
# Data Wrangling 3 - Joining the mcc codes to the transactions dataset based on the mcc.

df = df.merge(mcc_codes, left_on='mcc', right_on='index', how='left')
df.head()
```

✓ 3.6s Python

merchant_city	merchant_state	zip	...	address	latitude	longitude	per_capita_income	yearly_income	total_debt	credit_score	num_credit_cards	index	merchant_category
Beulah	ND	58523.0	...	594 Mountain View Street	46.80	-100.76	$23679	$48277	$110153	740	4	5499	Miscellaneous Food Stores
Bettendorf	IA	52722.0	...	604 Pine Street	40.80	-91.12	$18076	$36853	$112139	834	5	5311	Department Stores
Vista	CA	92084.0	...	2379 Forest Lane	33.18	-117.29	$16894	$34449	$36540	686	3	4829	Money Transfer
Harwood	MD	20776.0	...	166 River Drive	38.86	-76.60	$33529	$68362	$96182	711	2	5813	Drinking Places (Alcoholic Beverages)
Bronx	NY	10464.0	...	14780 Plum Lane	40.84	-73.87	$25537	$52065	$98613	828	5	5942	Book Stores

Figure 11: Sample rows of the transaction dataset after introducing the mcc codes dataset into the combined transactions dataset.

3.2.1 Data Cleaning

In the raw dataset, monetary values in columns such as amount, yearly_income and total_debt were prefixed with a dollar sign, and this caused them to be interpreted as strings rather than numeric values. This was addressed by removing the dollar sign and converting the numeric values into the float data type, which can handle decimals commonly found in monetary values.

```
# Removing the dollar sign ($) prefix from columns with monetary values.
def clean_amount(value_string):
    value_string = value_string.replace('$', '')
    return float(value_string)

df['amount'] = df['amount'].apply(clean_amount)
df['per_capita_income'] = df['per_capita_income'].apply(clean_amount)
df['yearly_income'] = df['yearly_income'].apply(clean_amount)
df['total_debt'] = df['total_debt'].apply(clean_amount)
```

✓ 7.0s

Figure 12: Data cleaning to convert monetary values from string to float.

Furthermore, negative values were removed from the “amount” column. While these may seem valid as they could be refunds that are the reverse of the flow of payments, this project will be focusing on analysing payment transactions.

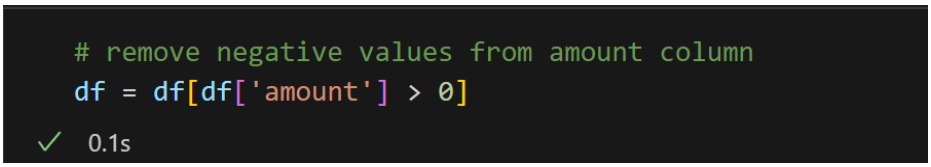


Figure 13: Data cleaning to remove negative values from the “amount” column.

Inspecting the dataset, missing values were found in the “errors” and “merchant_state” columns. The missing values in the error column were due to most transactions having no errors while the missing value in merchant_state always represents online transactions. To validate this, a filter was applied to identify missing 'merchant_state' values associated with non-online cities, returning no results. This proves that missing 'merchant_state' values are always online transactions in this dataset.

Figure 14 : Filtering for missing 'merchant_state' values associated with non-online cities returns no records in the DataFrame.

Therefore, new categories were created for each of these cases as the missing values were meaningful and not random. The missing values were then replaced with these new categories.

```
# Handle missing values in error column with 'NONE'.
# This represents a new category where no errors were eoncutered.
df['errors'] = df['errors'].fillna('NONE')

# Convert the column to category
df['errors'] = df['errors'].astype('category')
```
✓ 0.8s

```
# Handle missing values in merchant_state column with 'ONLINE'.
# Since all missing values in merchant_state arise to online transactions.
df['merchant_state'] = df['merchant_state'].fillna('ONLINE')
```
✓ 0.2s

Figure 15: Filling the new category 'NONE' for transactions with no errors and new category 'ONLINE' for the online transactions.

Finally, to further simplify the target variable, the "Yes" and "No" values were mapped to 1 and 0 for fraud and legitimate transactions.

```
# Map target values whereby 'Yes' = 1 and 'No' = 0
df['target'] = df['target'].map({'No': 0 , 'Yes': 1})
```
✓ 1.0s

Figure 16: Mapping the "Yes" and "No" values in the "target" column to 1 and 0, respectively.

3.2.2 Outlier Detection and Removal

Using the standard approach of defining outliers as values more than 1.5 times the interquartile range (IQR) above the third quartile or 1.5 times the IQR below the first quartile, many outliers were detected in the dataset. It was found that the number of outliers in the variables "transaction_amount" exceeded 200,000 and outliers in "retirement_age" exceeded 400,000 which is almost 10 percent of observations in the dataset.

```
Column 'transaction_amount' has 216312 outliers.
Column 'current_age' has 28735 outliers.
Column 'retirement_age' has 444315 outliers.
Column 'per_capita_income' has 253627 outliers.
Column 'yearly_income' has 234493 outliers.
Column 'total_debt' has 93771 outliers.
Column 'credit_score' has 140259 outliers.
Column 'num_credit_cards' has 9360 outliers.
```

Figure 17 : Number of outliers detected in each column when using 1.5 times the IQR above Q3 and 1.5 IQR below Q1 as the threshold.

This large number of outliers is due to the positive skew in the "transaction_amount" column, where it has a long tail to the right of its distribution as seen in figure 18. As for the retirement age variable, it has a symmetrical distribution with some extreme values where the user wants to retire over 70 or below 60 years old.

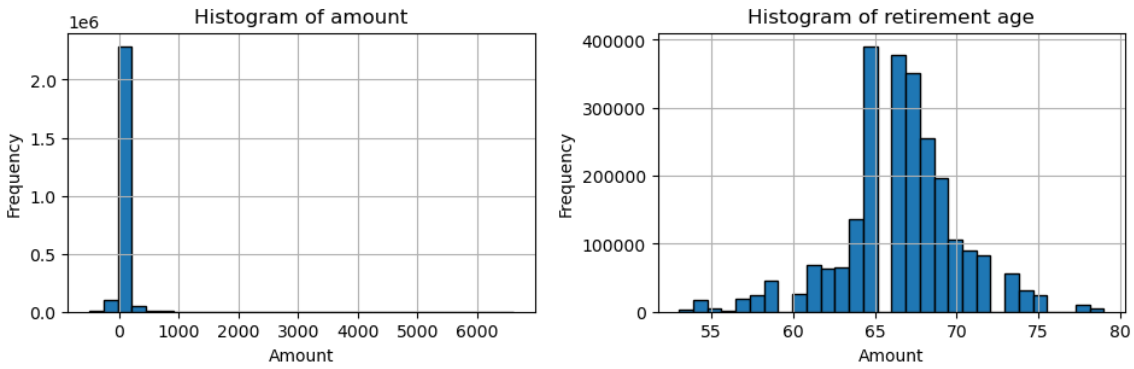


Figure 18 : Distribution of the "amount" variable and "retirement_age" variable visualised using a histogram.

To reduce the loss of large amounts of data, changes were made to only remove outliers that are more extreme, by calculating the interquartile range from the 10^{th} and 90^{th} percentiles instead of the 25^{th} and 75^{th} percentiles. This results in the removal of 33217 rows of outliers, corresponding to about 1.5% of the dataset. The number of remaining observations after the process of data cleaning and outlier removal is 2,309,828.

```
Column 'transaction_amount' has 60925 outliers.
Column 'current_age' has 0 outliers.
Column 'retirement_age' has 0 outliers.
Column 'per_capita_income' has 46131 outliers.
Column 'yearly_income' has 39998 outliers.
Column 'total_debt' has 13964 outliers.
Column 'credit_score' has 0 outliers.
Column 'num_credit_cards' has 0 outliers.
```

Figure 19 : Number of outliers detected after adjusting the interquartile threshold to the 10th and 90th percentiles.

3.3 Data Dictionary

After the data wrangling and data cleaning process, the resulting cleaned dataset has the following characteristics as seen in table 1.

Table 1 : Data dictionary of the combined dataset after data wrangling and cleaning.

No.	Column Name	Data Type	Description	Examples
1	transaction_id	Numeric	Unique transaction identifier.	15471193
2	client_id	Numeric	Client Identifier.	1585
3	card_id	Numeric	Card Identifier.	339
4	merchant_id	Numeric	Merchant identifier.	59935
5	transaction_datetime	datetime	Date and time of transaction.	2015-01-01 00:01:00
6	transaction_amount	Continuous	Transaction amount in dollars.	34.82
7	use_chip	Categorical	Whether the card chip is used.	Swipe Transaction
8	errors	Categorical	Errors encountered during the transaction.	Insufficient Balance, Bad CVV, …
9	merchant_city	Categorical	City where the merchant is located. Value is ‘ONLINE’	E.g. Beulah , Rome, ONLINE …

			when the transaction occurs online.	
10	merchant_state	Categorical	Country where merchant is located. Value is 'ONLINE' when the transaction occurs online and the country name if the transaction occurs overseas.	USA, Italy, Mexico, Canada, United Kingdom, ONLINE
11	merchant_category	Categorical	Merchant category code labels.	Department Stores
12	current_age	Discrete	Current age of client.	53
13	retirement_age	Discrete	Retirement age of the client.	65
14	gender	Categorical	Gender of client.	Male, Female
15	per_capita_income	Continuous	Income per capita of client's region.	29278.0
16	yearly_income	Continuous	Annual income of the client.	59696.0
17	total_debt	Continuous	Total debt of the client.	127613.0
18	credit_score	Discrete	Credit score of clients.	787
19	num_credit_cards	Discrete	Number of credit cards owned by client.	5
20	target	Binary	Binary fraud label, 1 for fraud and 0 for legitimate transactions.	0 , 1

3.4 Feature Engineering for Modelling

In order to prepare the dataset for modelling, several steps were taken to extract additional features, as well as transform the existing features into a more suitable format. As the transaction timestamp given in datetime format was not usable directly, new time-based features were extracted from the transaction datetime. This includes the transaction hour, day

of week and month, which could potentially have seasonal patterns that are informative for the model when classifying if a transaction is likely to be fraud.

```
errors
NONE                                          4146140
Insufficient Balance                            41785
Bad PIN                                         10272
Technical Glitch                                 8526
Bad Card Number                                  2668
Bad CVV                                          2092
Bad Expiration                                   2075
Bad Zipcode                                       326
Bad PIN,Insufficient Balance                      100
Insufficient Balance,Technical Glitch              84
Bad PIN,Technical Glitch                           23
Bad Card Number,Insufficient Balance               20
Bad Expiration,Insufficient Balance                19
Bad CVV,Insufficient Balance                       18
Bad Card Number,Bad CVV                            14
Bad Card Number,Bad Expiration                     14
Bad Expiration,Bad CVV                             11
Bad Expiration,Technical Glitch                     8
Bad Card Number,Technical Glitch                    4
Bad Zipcode,Technical Glitch                        2
Bad CVV,Technical Glitch                            1
Bad Zipcode,Insufficient Balance                    1
Name: count, dtype: int64
```

Figure 20: Frequency of each unique category in the error column. Some of the less common categories are made up of a list of errors.

In addition, the error column was one hot encoded such that each of the seven types of errors has its own column. This is to resolve situations whereby multiple errors had occurred during one transaction, which result in a list of errors instead of a single value. Not processing this would complicate the analysis as there were various rare cases with different combinations of errors. By separating the individual errors into one hot encoded column, it was then possible to evaluate if there are certain types of errors that may indicate a higher likelihood of fraud.

```
# Process the error column such that each type of error has one dummy column
error_dummies= df['errors'].str.get_dummies(sep=',')

# View example of first five rows
error_dummies.head()
```

✓ 10.9s

	Bad CVV	Bad Card Number	Bad Expiration	Bad PIN	Bad Zipcode	Insufficient Balance	NONE	Technical Glitch
0	0	0	0	0	0	0	1	0
1	0	0	0	0	0	0	1	0
2	0	0	0	0	0	0	1	0
3	0	0	0	0	0	0	1	0
4	0	0	0	0	0	0	1	0

Figure 21: The “errors” column after one hot encoding, with each unique error type having its own column.

3.5 Dividing the Dataset – Offline and Online Transactions Dataset

Due to the different nature of the online and offline transactions, the dataset was split into the offline transactions dataset and online transactions dataset. Subsequently, data analysis and modelling were then conducted separately for each dataset. While the number of transactions differ, the number of fraudulent transactions were roughly equal in both datasets, providing sufficient data for analysis and modelling separately.

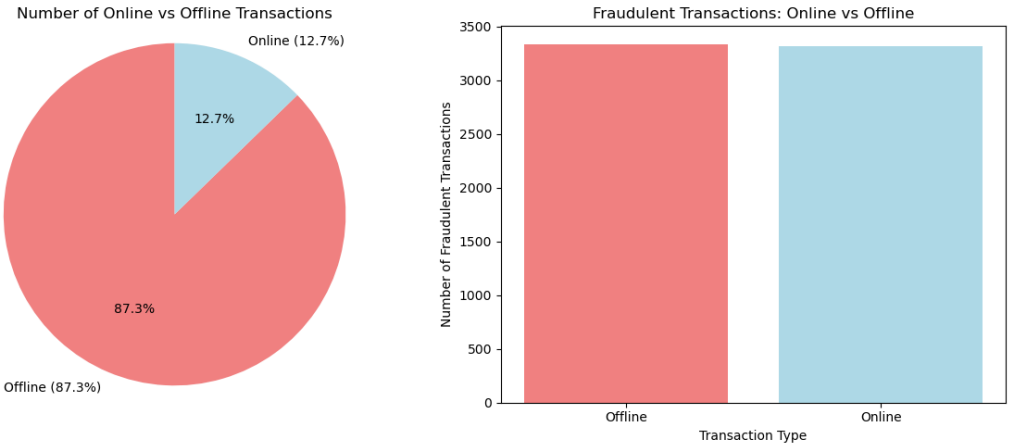


Figure 22 (left): Pie chart comparing the distribution of offline transactions and offline transactions in the dataset. **Figure 22 (right):** Bar chart illustrating the number of fraudulent transactions in online versus offline transactions. Despite the difference in total transaction counts, the number of fraudulent transactions is almost the same.

3.6 Exploratory Data Analysis (EDA)

Exploratory data analysis of the dataset was carried out on the offline and online transactions dataset to determine the variables that may indicate a higher likelihood of fraud. Key insights from each dataset are presented in this section.

3.6.1 Offline Transactions Dataset

Do fraudulent transactions differ in amount from legitimate ones?

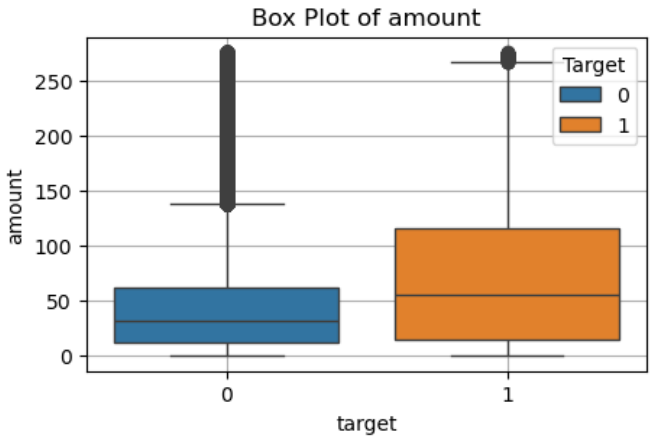


Figure 23: Box plot comparing the distribution of transaction amounts in fraudulent and legitimate transactions.

From figure 23 above, it can be observed that the median and third quartile of fraudulent transactions appear higher compared to those of legitimate transactions. This suggests that fraudulent transactions tend to have a higher amount than legitimate ones. Additionally, the body of the boxplot for the fraudulent transactions is much longer than the boxplot for legitimate transactions, signalling a higher variability in the value of these transactions.

Who are the users more vulnerable to fraud?

In terms of the demographic factors, the box plots (Figure 24) of the fraudulent and legitimate transactions do not show much difference, with only a slightly higher median for current and retirement ages for the fraud transactions. However, by binning the user's age into age groups and calculating the proportion of fraudulent transactions, a pattern emerges whereby seniors above the age of 60 were found to be more vulnerable to fraud. As seen in figure 25, the

proportion of fraud in the “61+” age group were higher at 0.001 compared to the other age groups.

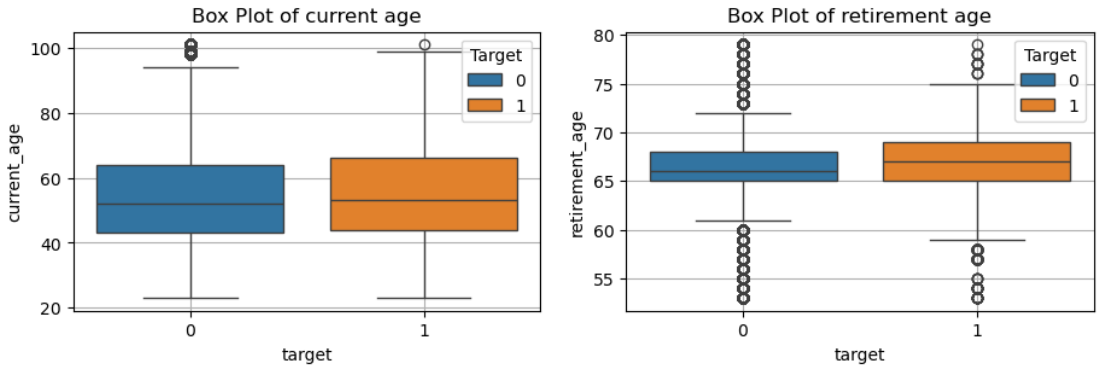


Figure 24: Box plot comparing the distribution of the user’s age and retirement age.

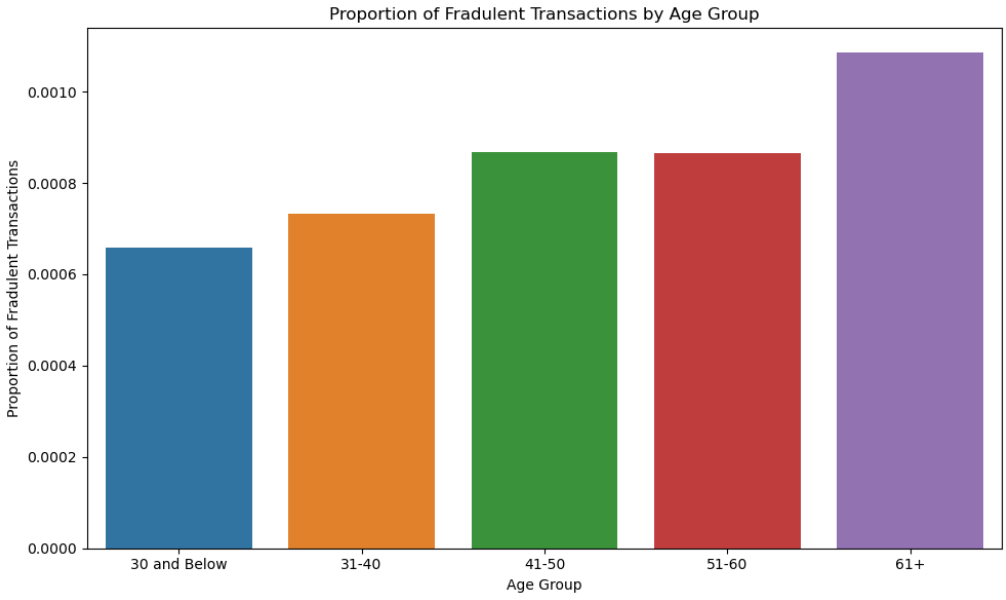


Figure 25: Bar plot comparing the proportion of fraudulent transactions in each age group.

In addition, the financial demographics variables such as the user’s income bracket also show an obvious difference in the proportion of fraudulent transactions. This can be seen in figure 26, where the user group with incomes above 200k had the greatest incidence of fraud.

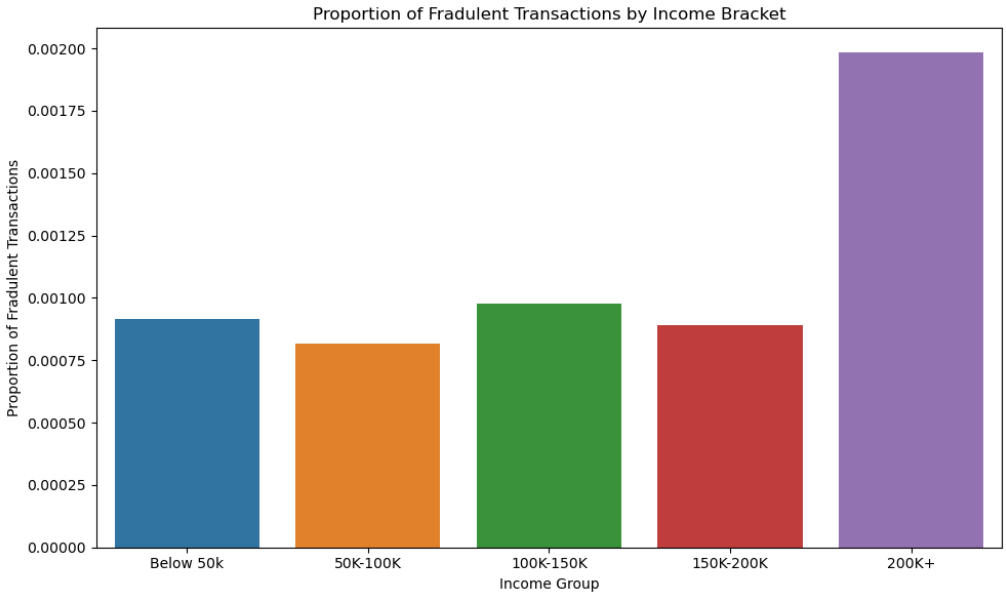


Figure 26: Bar plot comparing the proportion of fraudulent transactions in each income bracket.

Which merchant categories have the highest proportion of fraudulent transactions?

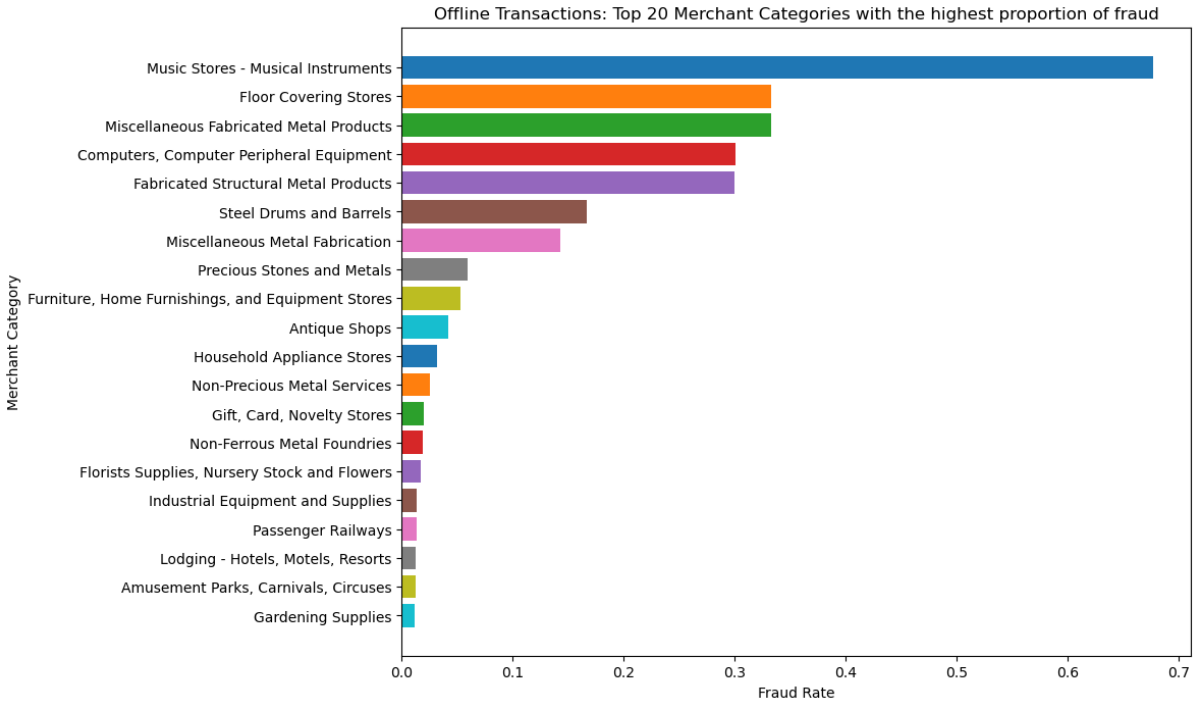


Figure 27: Bar plot of top 20 merchant categories with the highest proportion of fraudulent transactions.

The merchant category shows an interesting pattern whereby rare categories such as musical instruments, floor covering stores, metal products and computer equipment had a much higher fraud rate than other categories. As seen in figure 27, the top 20 merchant categories with high fraud rates consist of many rare categories that most users would not be buying on a typical day. This could indicate suspicious activity and potentially help to identify fraud.

Creating time-based features from the timestamp generated additional features that could be analysed for insights. Unsurprisingly for the offline transactions dataset, most of the fraudulent transactions took place during daytime hours, from 8 am to 8 pm, where physical stores would be open for business. In terms of the day of week feature, most fraud occurs on weekdays around Thursdays or Fridays and occur less frequently during the weekends. As for the month variable, the number of fraud cases fluctuate, with no apparent pattern. Line plots for each time-based feature can be seen on the following page (Figure 28).

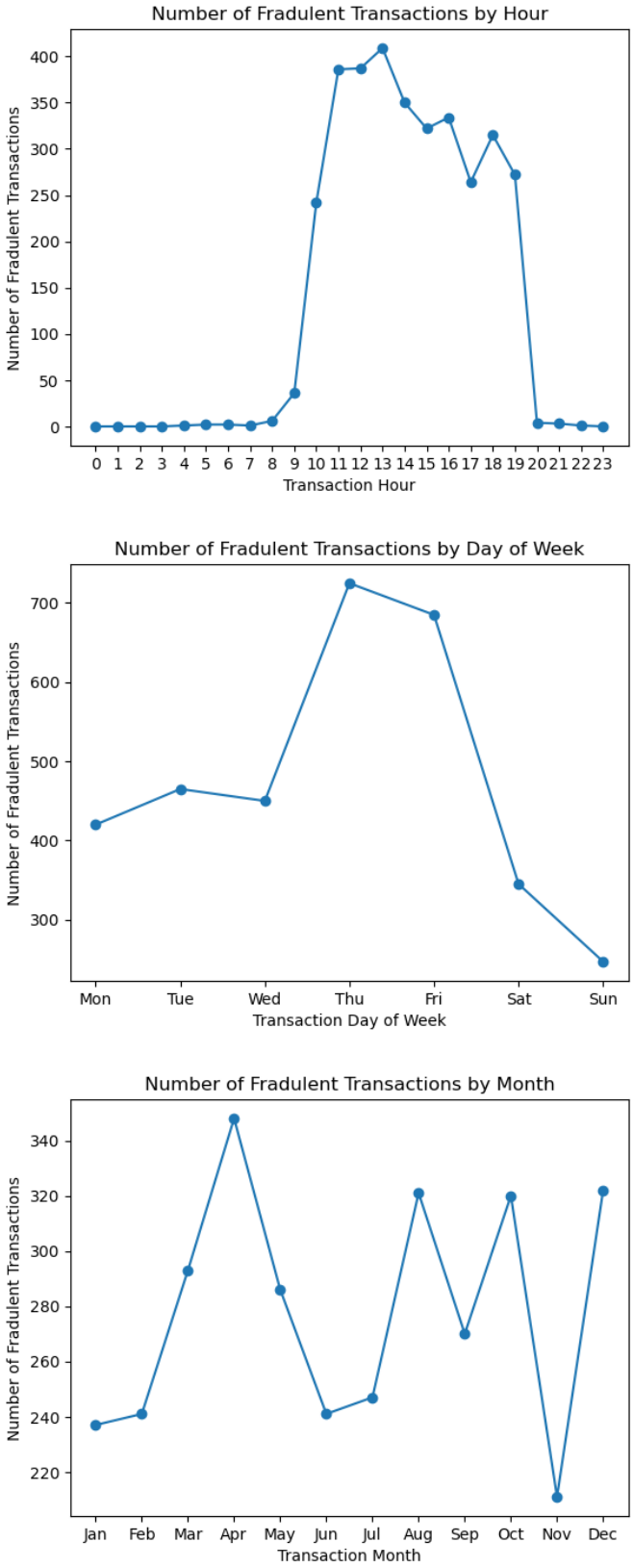


Figure 28: Line plots of the time-based features created from the datetime variable.

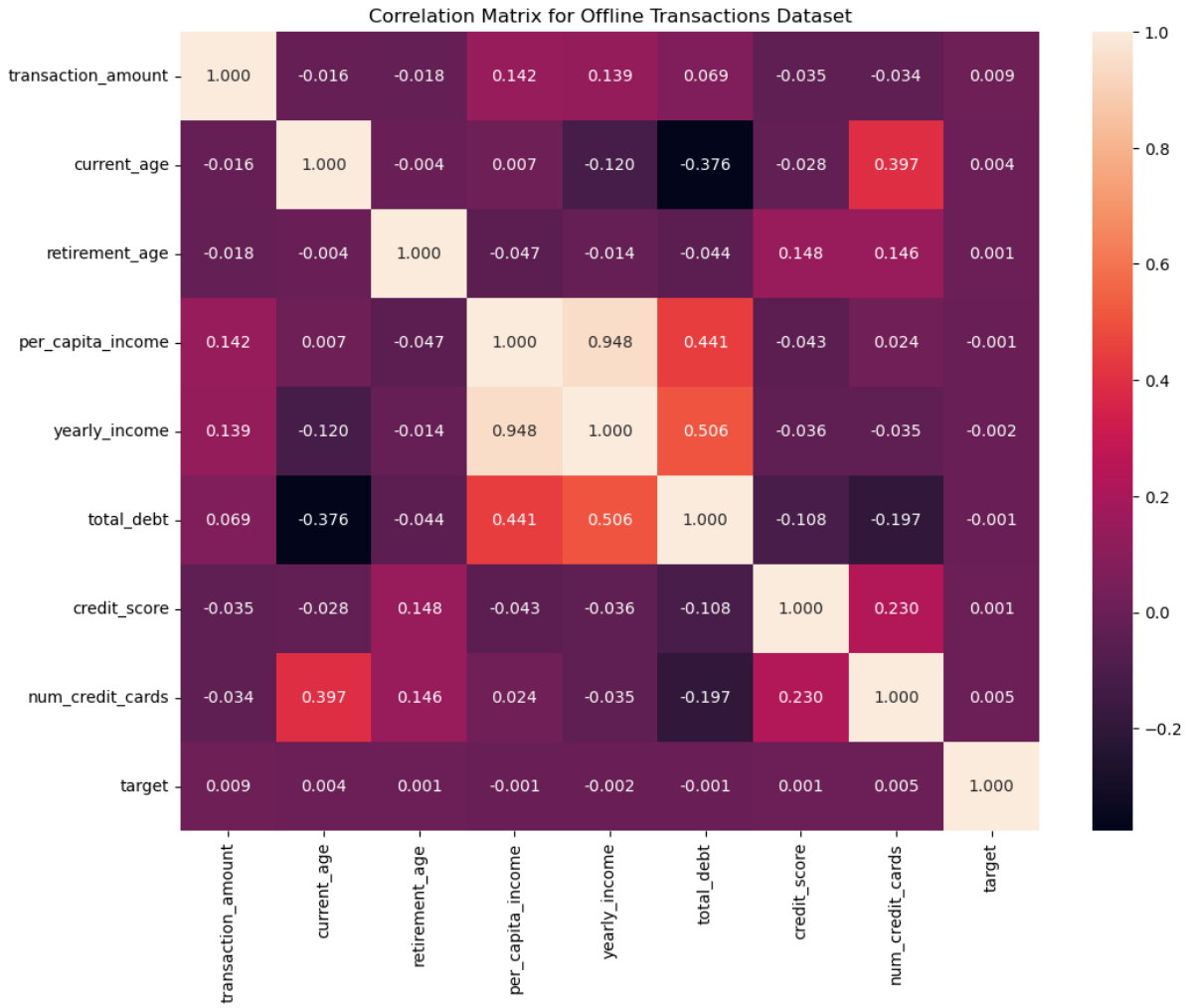


Figure 29: Correlation matrix showcasing the correlation between each of the numeric variables, as well as the target in the offline transactions dataset.

All the numeric variables in figure 29 show almost no correlation with the target variable, with the exception of "transaction_amount" showing a very weak positive correlation of 0.09. Some interesting correlations between age and the variables that indicate financial status can be derived. For instance, age is positively correlated to the number of credit cards the customer owns and yet negatively correlated with the amount of debt they incur. This suggests that older customers may be more prudent with spending and avoid taking on more debt, despite having more cards.

Total debt was observed to be positively correlated with the customer's annual income, with high income earners also taking on more debt. The annual income also has a very strong positive correlation (0.948) with per capita income, indicating that only one of these variables need to be used as features in modelling process to avoid multicollinearity.

3.5.2 Online Transactions Dataset

EDA was carried out separately for the online transactions dataset using the same visualisation techniques as the previous section. Like the offline transactions dataset, the seniors in the "61+" group were more likely to be impacted by fraudulent transactions, as there is a higher proportion of transactions being fraud in this group. The income bracket with the highest proportion of fraud was also different for the online transactions, with the "150K to 200k" group being the most susceptible to fraud (Figure 30).

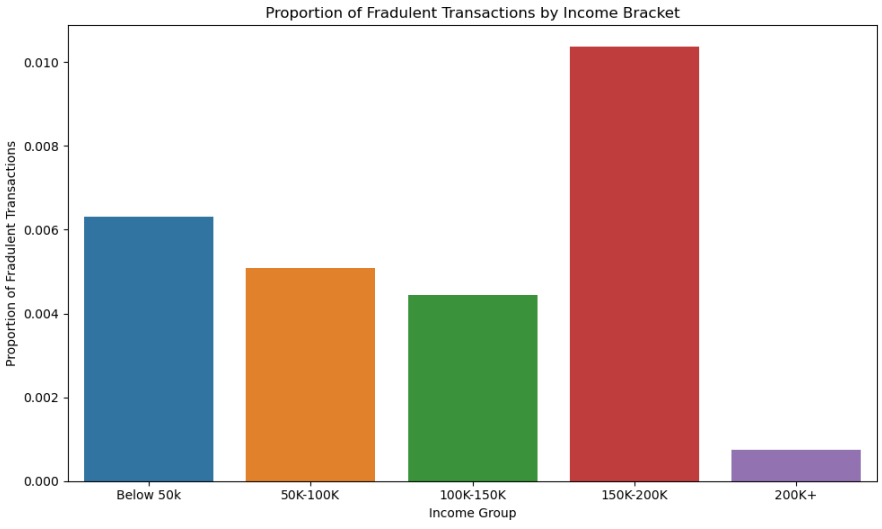


Figure 30: Bar plots comparing the proportion of fraudulent transactions in each income bracket for the online transactions dataset.

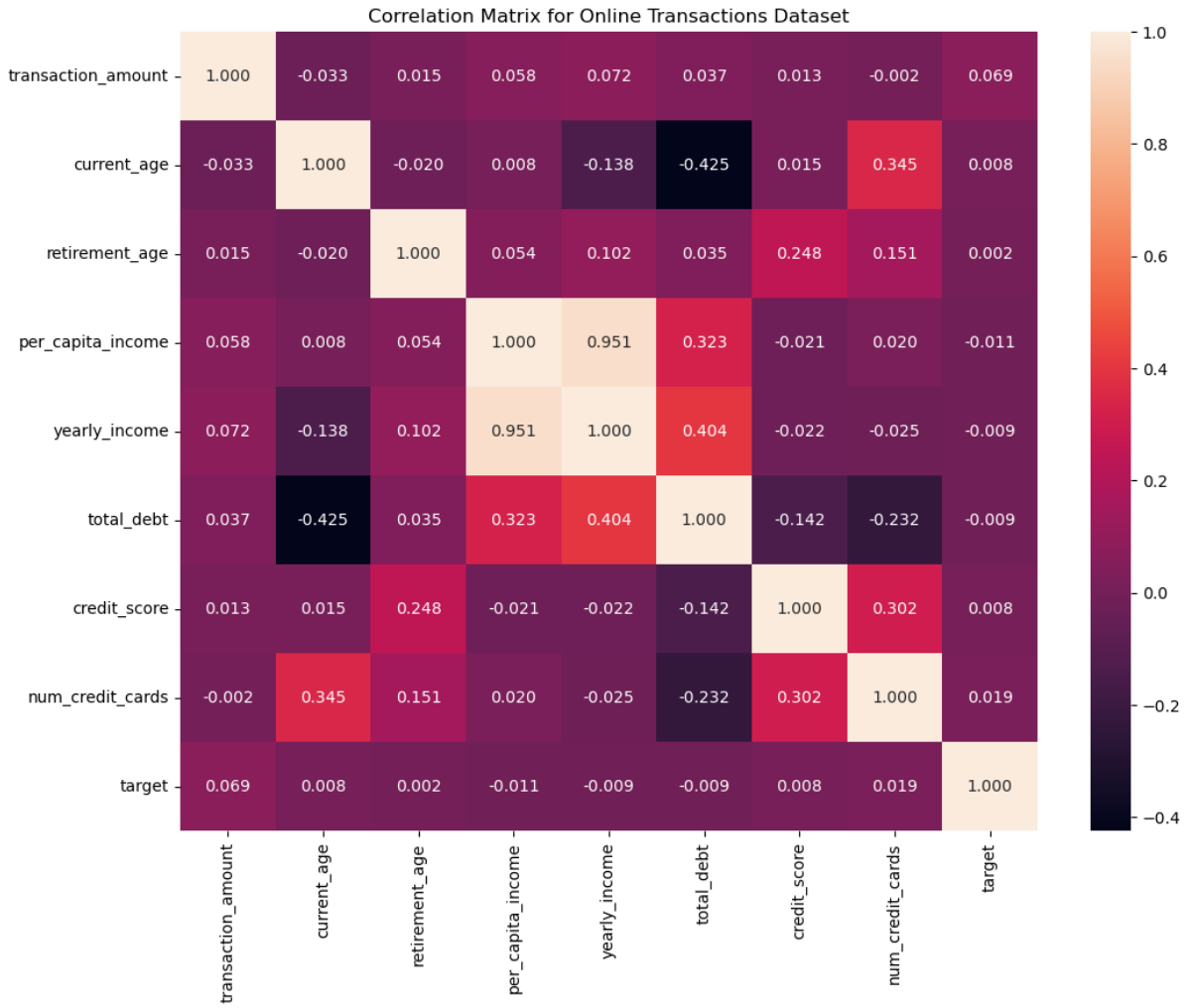


Figure 31: Correlation matrix showcasing the correlation between each of the numeric variables, as well as the target in the offline transactions dataset.

The correlation matric for the online transactions dataset show almost the same correlations between variables as the offline transactions dataset. The correlations between variables like total debt and age, income and per capita income, as well as income and debt all remain similar to figure 29, with slight differences in magnitude.

3.6 Data Preprocessing for Modelling

In the preprocessing step, all numeric variables were first scaled to standardise their values, ensuring that features with varying magnitudes do not disproportionately influence the model results. This is particularly important for models like K-nearest neighbours (KNN) and neural networks.

High cardinality categorical variables with at least ten different categories were then encoded using target encoding. This was done using the target encoder from the category_encoders python library, which replaces categories with the mean of the target variable for that category. This corresponds to the proportion of fraudulent transactions in each category, which was found to be highly informative for various features through EDA conducted in the previous section. The variables "merchant_category", "merchant_city", "merchant_state", all the identifier columns, as well as transaction_month" and "transaction_hour" were encoded with the target encoding technique.

Low cardinality categorical variables were also encoded separately using one hot encoding, which creates a new binary column for each unique category and marks the category that is present with 1. The variables “use_chip”, “gender” and “transaction_dayofweek” were encoded using the one hot encoding technique.

Chapter Four – Modelling Results and Discussion

4.1 Methodology

The approach for classifying transactions as fraud or legitimate can be defined as a binary classification problem. Commonly used models for binary classification problems include

logistic regression, decision tree, random forest, neural networks, gradient boosting, support vector classifier (SVC) and KNN. These machine learning models were then constructed and evaluated using 5-fold cross validation to determine each model's generalisation ability on unseen data. The best models out of the 7 models compared was then chosen based on the F1-score, which is the harmonic mean of precision and recall. This scoring criteria considers both precision and recall, ensuring that the chosen model can capture most of the positive class (fraudulent transactions) without also flagging many false positives. The precision, recall , accuracy, fitting time and scoring time of each model were also provided for comparison. The models were trained separately for the offline transactions and offline transactions datasets and one champion model was chosen for each of the datasets.

```
models = [
    ('RandomForest', RandomForestClassifier(random_state=42, n_jobs = -1)),
    ('LogisticRegression', LogisticRegression(max_iter=2000, random_state=42)),
    ('DecisionTree', DecisionTreeClassifier(random_state=42)),
    ('K_NearestNeighbour',KNeighborsClassifier(n_neighbors=3)),
    ('SVC', SVC(random_state=42)),
    ('NeuralNetwork', MLPClassifier(max_iter=2000, random_state=42)),
    ('GradientBoosting', GradientBoostingClassifier(random_state=42))
]
```

Figure 32 : Model parameter settings for each of the machine learning models used in the 5-fold cross validation.

4.2 Model Evaluation using Cross validation

4.2.1 Offline Transactions Dataset

Using the dataset consisting of only offline transactions, the F1-score, precision, recall, accuracy, as well as the model fitting and scoring time was computed for each candidate model. The results are presented in table 2 below.

Table 2 : Offline transactions dataset results for each candidate model using the 5-fold cross validation technique, ordered from the best model to the worse in terms of F1-score. The average time taken for fitting and scoring of each model is also given.

Model Name	Accuracy	Precision	Recall	F1-score	Fit time (Seconds)	Score time (Seconds)
Random Forest	0.9997	0.8911	0.8034	0.8447	17.54	3.35
SVC	0.9996	0.7883	0.8150	0.8010	227.90	208.97
Gradient Boosting	0.9996	0.7386	0.8026	0.7686	68.91	1.69
Logistic Regression	0.9995	0.7025	0.8191	0.7557	2.90	1.69
Neural Network	0.9994	0.6413	0.8236	0.7200	20.69	1.62
Decision Tree	0.9994	0.6474	0.7974	0.7134	3.62	1.74
K-Nearest Neighbours	0.9995	0.7932	0.5936	0.6788	2.59	668.47

Evaluating the results, random forest was found to have the best overall performance at classifying fraudulent transactions, with a F1-score of 0.8447. Additionally, random forest also has the highest precision at 0.8911 compared to other models. The SVC model also perform very similarly to random forest in terms of recall, but fall short in terms of precision, leading to a lower F1-score.

The average fitting time taken for each of the models also varied greatly, with SVC having the longest training time compared to the others. This highlights that training an SVC model is computationally demanding and may not be ideal for large datasets such as this transaction dataset. In terms of scoring time, the KNN model took a significantly longer time for scoring the test fold compared to all other models. Since there is a need to process large volumes of transactions, this slow scoring time would also make the KNN model less suitable as a candidate model for fraud classification applications.

Therefore, considering the model performance and scoring time, the random forest model was selected as the champion model for the offline dataset due to its excellent performance and lower computational requirements.

4.2.2 Online Transactions Dataset

The same model evaluation and selection process was conducted separately for the online transactions dataset, which has the same features present in the offline transactions dataset. Overall, the results from the 5-fold cross validation were better compared to the offline transactions dataset across all models. For instance, the random forest model for the online transactions dataset achieved an F1-score of 0.8815, which is better compared to the same random forest model which only achieved the F1-score of 0.8447 when trained and evaluated on the offline transactions dataset. This suggests that fraud patterns in online transactions are less complex and more distinct from legitimate transactions, leading to better model performance compared to offline transactions.

Table 3: Online transactions dataset results for each candidate model using the 5-fold cross validation technique, ordered from the best model to the worse in terms of F1-score. The average time taken for fitting and scoring of each model is also given.

Model Name	Accuracy	Precision	Recall	f1 score	Fit time (Seconds)	Score time (Seconds)
Random Forest	0.9987	0.9851	0.7977	0.8815	18.84	0.49
Neural Network	0.9984	0.9371	0.7980	0.8618	38.45	0.42
Logistic Regression	0.9983	0.9731	0.7491	0.8465	2.75	0.34
Gradient Boosting	0.9983	0.9601	0.7536	0.8442	60.71	0.21
SVC	0.9980	0.9940	0.6861	0.8118	109.02	28.45
Decision Tree	0.9972	0.7880	0.7577	0.7721	4.00	0.34
K-Nearest Neighbour	0.9948	0.8283	0.2023	0.3248	2.11	146.94

From the results, the random forest model was also the best performing model for the online transactions dataset with the highest F1-score of 0.8815 amongst the models tested. The neural network was also similar in performance, with comparable recall and slightly lower precision than the random forest model. As the online transactions dataset was smaller than the offline

transactions dataset, the fitting time was less important, and most models were able to complete training under one minute except for SVC and gradient boosting. The scoring time was also relatively quick and the top two models, namely random forest and neural network, were able to completely score the test fold in under one second.

Therefore, based on the F1-score, the random forest model was also selected as the champion model for the online transactions dataset.

4.3 Hyperparameter tuning

4.3.1 Offline Transactions Dataset

To optimise the performance of the chosen random forest model, hyperparameter tuning was carried out using the GridSearchCV module from Sklearn, which trains and evaluates model performance for a chosen grid of parameter values. The grid used for tuning the random forest model is shown in figure 33 and consists of parameters such as the number of trees, the number of features to consider when looking for the best split, the maximum depth of the trees and the criterion to measure the quality of a split. These were chosen to control the growth of the tree in random forest to avoid overfitting to the training data.

```
param_grid = {
    'classifier__n_estimators': [100, 200, 300],
    'classifier__max_features': ['sqrt', 'log2'],
    'classifier__max_depth': [5, 10, 15, 20],
    'classifier__criterion': ['gini', 'entropy']
}
```

Figure 33 : Parameter grid used for hyper parameter tuning of the random forest model for the offline transactions dataset.

Table 4: Comparison of the model performance for the offline transactions dataset before and after hyperparameter tuning.

Model	Accuracy	Precision	Recall	F1-score
Random Forest (Default settings)	0.9997	0.8911	0.8034	0.8447
Random Forest (After hyperparameter tuning)	0.9997	0.9320	0.7811	0.8499

After the best hyperparameters were identified by GridSearchCV, the tuned model was evaluated on the unseen test set to determine the improvement over the model with default hyperparameters. The tuned model was able to achieve a F1-score of 0.8499, which is higher than the F1-score of 0.8447 before hyperparameter tuning. While the recall fell slightly, the precision increased from 0.8911 to 0.9320 in the tuned model. Hence, the tuning successfully improved the model performance, and the tuned model was selected as the champion model to be used for the offline dataset.

The hyperparameters of the tuned random forest model for the offline transactions dataset are as follows:

- criterion: entropy
- max_depth: 10
- max_features: sqrt
- n_estimators: 200
- random_state = 42

4.3.2 Online Transactions Dataset

Similar to the offline transactions dataset, the hyperparameters for the random forest model was also tuned using GridSearchCV to optimise the model's performance. The parameter grid consists of the same hyperparameters but with slightly different values based on trial and error, to maximize the F1-score obtained.

```
param_grid = {
    'classifier__n_estimators': [200, 300, 400],
    'classifier__max_features': ['sqrt', 'log2'],
    'classifier__max_depth': [20, 25, 30],
    'classifier__criterion': ['gini', 'entropy']
}
```

Figure 34 : Parameter grid used for hyper parameter tuning of the random forest model for the online transactions dataset.

Table 5: Comparison of the model performance for the online transactions dataset before and after hyperparameter tuning.

Model	Accuracy	Precision	Recall	F1-score
Random Forest (Default settings)	0.9987	0.9851	0.7977	0.8815
Random Forest (After hyperparameter tuning)	0.9988	0.9926	0.8072	0.8903

Evaluating the tuned model on the test set, the F1 score was found to have slightly increased from 0.8815 to 0.8903. Both recall and precision improved, with recall increasing from 0.7977 to 0.8072 and the precision increasing from 0.9987 to 0.9988. Hence, the tuned model was adopted as the champion model to be used for the online transactions dataset.

The hyperparameters of the tuned random forest model are as follows:

- criterion: gini
- max_depth: 20
- max_features: log2
- n_estimators: 300
- random_state = 42

4.4 Decision threshold tuning

The decision threshold is the probability threshold at which the prediction is classified to belong to a certain class. By default, this is set at 0.5 for binary classification, and probabilities above 0.5 are classified as the positive class (1 in this case) while those below 0.5 are classified as the negative class (0 in this case). Tuning this decision threshold may potentially improve the model performance by adjusting the balance between precision and recall depending on the business requirements.

4.4.1 Offline Transactions Dataset

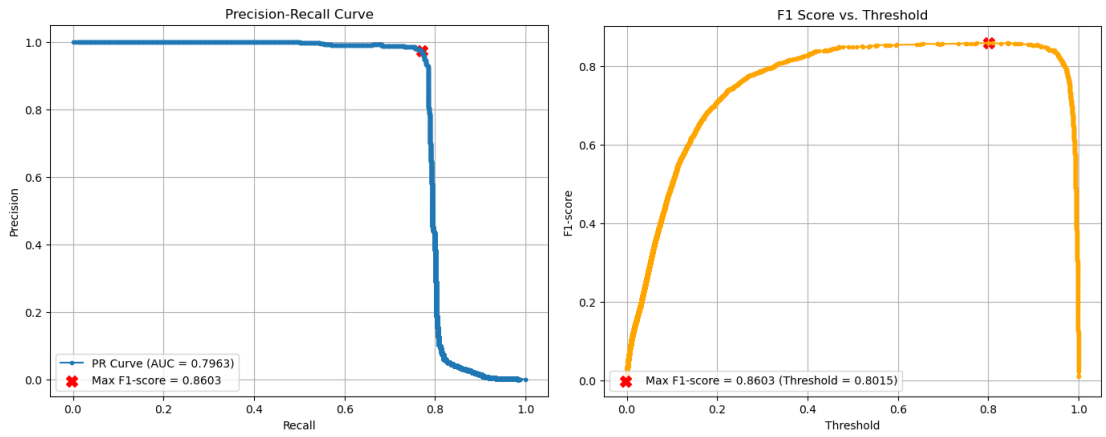


Figure 35 : Precision-recall curve and F1-Score vs. threshold for the offline transactions dataset.

The precision-recall curve above displays the precision and recall across different decision thresholds. Generally, as the decision threshold decreases, the model becomes less strict. This leads to an increase in recall at the expense of precision, as the model also makes more false positive predictions. For the tuned random forest model, the precision plateaus at recall values below 0.8 and only falls sharply right before the recall reaches 0.80. This means that it would

be unwise to choose to aim for recall above 0.8, as the corresponding precision would be very low. To optimise the decision threshold, the point at which the maximum F1-score occurs was identified to be 0.8603 at the threshold value of 0.8015. This point occurs at the turning point of the precision-recall curve right before the sharp drop as seen in figure 35 above.

Table 6: Comparison of the model for the offline transactions dataset before and after decision threshold tuning.

Model	Accuracy	Precision	Recall	F1-score
Random Forest (After hyperparameter tuning)	0.9997	0.9320	0.7811	0.8499
Random Forest (After adjusting decision threshold)	0.9998	0.9735	0.7706	0.8603

Selecting this new threshold value improves the F1-score to 0.8603 from 0.8499, and increases the precision to 0.9735, but slightly decreases the recall to 0.7706. While this means that overall performance of the model has improved, recall is still more important for fraud detection and therefore, the model with the default threshold was selected as the champion model for the offline transactions dataset.

4.4.2 Online Transactions Dataset

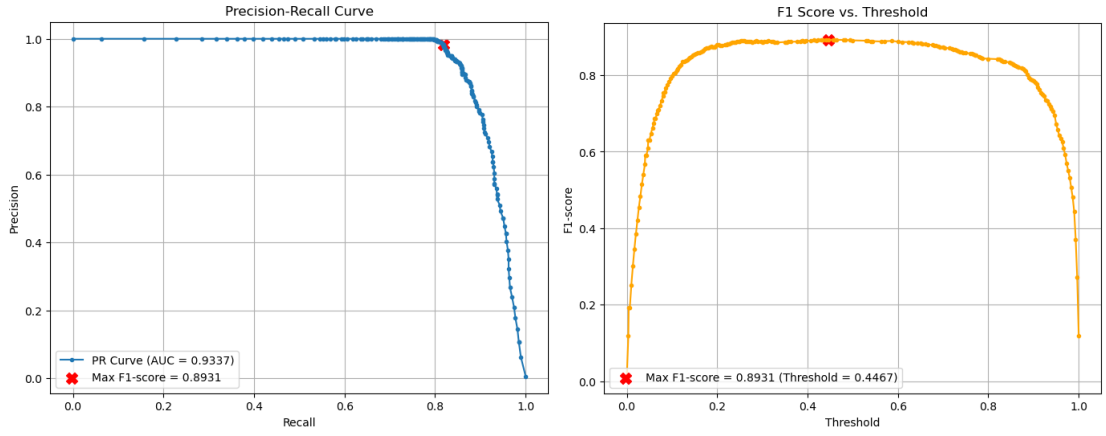


Figure 36 : Precision-recall curve and F1-Score vs. threshold for the online transactions dataset.

Table 7: Comparison of the model for the online transactions dataset before and after decision threshold tuning.

Model	Accuracy	Precision	Recall	F1-score
Random Forest (After hyperparameter tuning)	0.9988	0.9926	0.8072	0.8903
Random Forest (After adjusting decision threshold)	0.9988	0.9837	0.8178	0.8931

The precision-recall curve of the tuned model for the online transactions dataset also displays a plateau for all recall values below 0.8 but only begins to fall off after the recall exceeds 0.80. From the higher Area Under the Curve (AUC) which is 0.93, the model also performs extremely well in balancing the precision and recall. To tune the decision threshold for the online transactions dataset without sacrificing too much precision, the point at which the maximum value of the F1-score occurred was determined to be 0.8931, and the corresponding threshold was found to be 0.4467. This new F1-score is an improvement over the score obtained from the hyperparameter tuned model. The adjusted model also had a higher recall of 0.8178 compared to 0.8072 for the tuned model. This comes at the expense of precision, which fell to 0.9837. Since precision is less important for fraud detection applications, the model with the adjusted threshold was chosen for the champion model for the online transactions dataset.

4.5 Results and Discussion

Table 8: Comparison of the champion model for each type of transaction dataset.

Dataset	Model	Accuracy	Precision	Recall	F1-score
Offline	Random Forest (With hyperparameter tuning)	0.9997	0.9320	0.7811	0.8499
Online	Random Forest (With hyperparameter tuning and adjusted decision threshold)	0.9988	0.9837	0.8178	0.8931

After the model selection, hyperparameter tuning and threshold adjustments, the champion model chosen for each dataset is given in table 8. Both models exceeded the target F1-score of 0.80 set in the project objectives and achieved high precision and recall. Achieving a recall of

0.7811 means that for the offline transactions dataset, the model can catch 78.11% of fraudulent transactions successfully, while a precision of 0.9320 means that out of the predicted fraudulent transactions, 93.20% of them are actually fraud. Similarly for the online transactions, the model performs even better, being able to detect 81.78% of fraudulent transactions, with 98.37% of predicted fraudulent transactions being actual fraud. This high precision suggests that the models can be used to quickly identify fraudulent transactions and notify fraud investigators or users for prompt action, without overwhelming them with false positives.

4.5.1 Offline Transactions Dataset – Interpreting the results

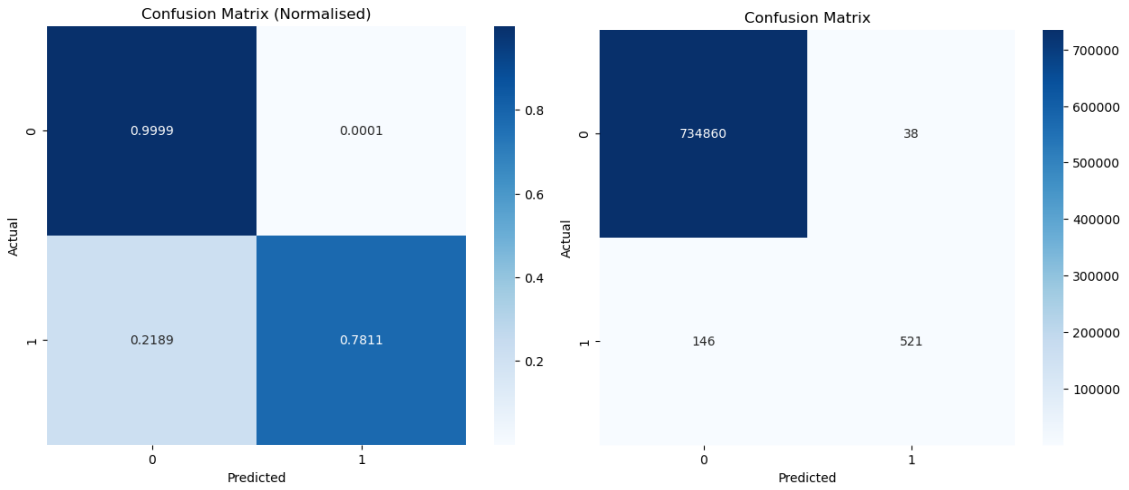


Figure 37: Confusion matrices for the final random forest model scored on the test set.

The confusion matrices above show how well the predicted labels match the actual labels. In the top left quadrant, the model is shown to have predicted almost all legitimate transactions correctly, with only 38 transactions being wrongly classified as fraud. On the other hand, the lower left quadrant contains 146 fraudulent transactions wrongly classified as legitimate, highlighting the lower recall of this model.

The feature importances illustrates the features that are the most important when classifying fraudulent transactions. For the offline transactions dataset, the location of the merchant was found to be the most important, with features like “merchant_city” and “merchant_state” accounting for most of the variance in the target variable. This is followed by the merchant and card identifier features, which was encoded with the mean of the target variable. This suggests that there are certain merchants and locations that may have a higher incidence rate of fraud.

The “merchant_category” feature places fifth on the list of the most important features. This corresponds to the patterns observed during the EDA, where there are certain merchant categories that have a higher proportion of fraud.

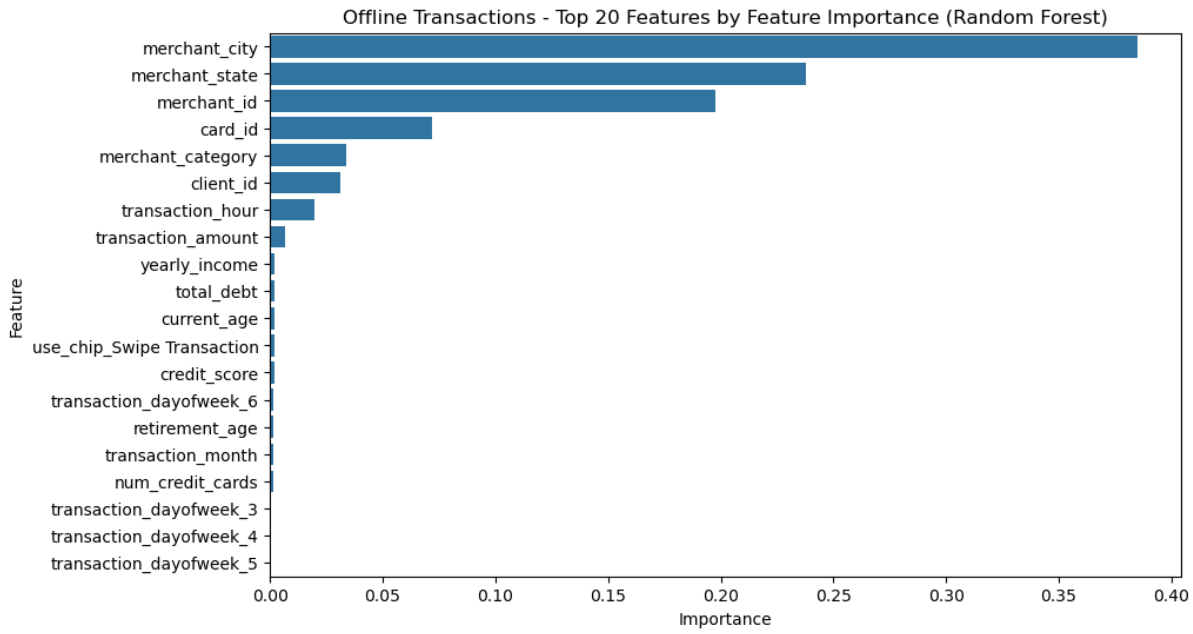


Figure 38 : Top 20 most important features extracted from the final random forest model.

4.5.2 Online Transactions Dataset – Interpreting the results

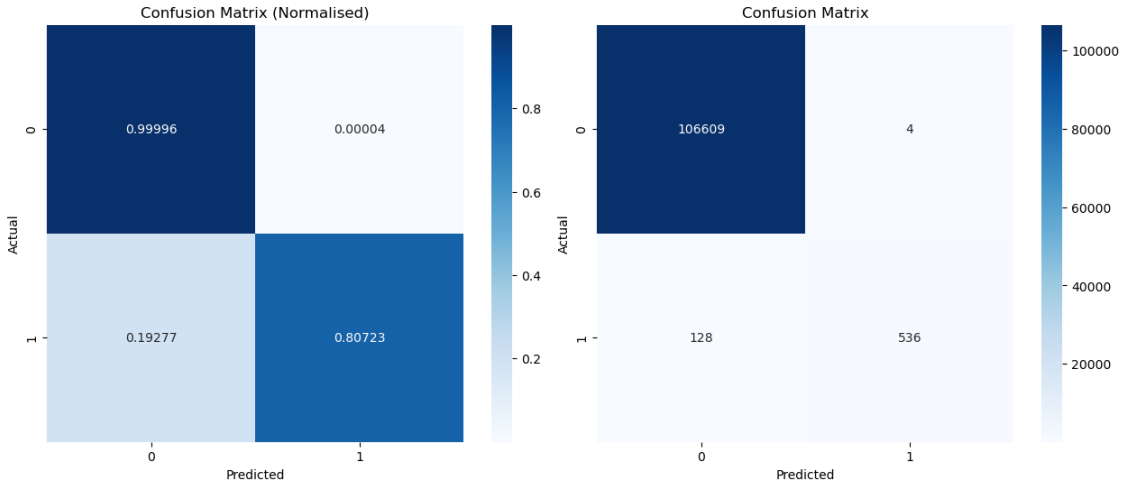


Figure 39: Confusion matrices for the final random forest model scored on the test set.

For the online transactions dataset, the model also correctly classified almost all the legitimate transactions. The result for the lower left quadrant was also better, with only 128 fraudulent transactions being wrongly classified as legitimate.

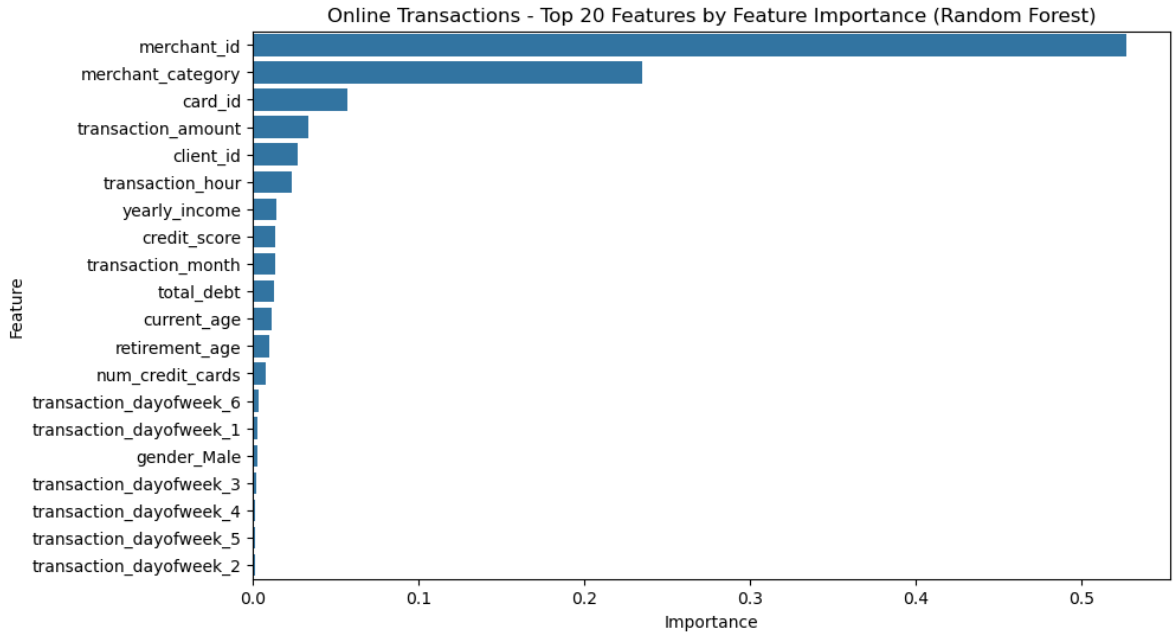


Figure 40 : Top 20 most important features extracted from the final random forest model.

For the online transactions dataset using the tuned random forest model, the merchant id was the most important, explaining over 40% of the variance. This could be due to some specific online merchants having insecure payment gateways or are less strict with user verification, leading to more fraud. The merchant category and card id were the next most important factors, symbolising the importance of identifying high risk merchant categories and cards that may have their details compromised by criminals. Besides merchant details, transaction amount ranks fourth in explaining the variance in the target. This could be due to differences in transaction amount when criminals make fraudulent transactions. Lastly, time based features and user demographic features also play a small part in explaining the model predictions.

4.5.3 Model Implications and Business Impact

The numbers from the confusion matrix only tell us about the number of transactions classified as either fraudulent or legitimate, as well as whether the model was correct. To dive into the business impact of using the model, some calculations were required to estimate the monetary value that could be saved by using the random forest models to detect fraud. Since the test data contains 20% of five years' worth of transaction data for 2000 customers, calculating the total cost incurred by fraud can give an approximation of the monetary value lost to credit card fraud per 2000 customers in a year. The amount of money saved by using the machine learning models built in this project can then be estimated by making a few assumptions.

Assumptions:

- Assuming a complete loss equal to the value of the transaction when fraud occurs undetected and a small loss of 5% of the transaction value when a false positive occurs to account for operational costs.
- Assuming no part of the transaction value is lost when fraud is detected by the model, as prompt action can be taken.

- The sample of 2,000 customers is assumed to be representative of the larger customer population in terms of transaction behaviour and fraud occurrence.

Offline Transactions Dataset	**Online Transactions Dataset**
Cost of fraud without model:	Cost of fraud without model:
$36099.43	$60498.54
Cost of fraud with model:	Cost of fraud with model:
$ 11357.42	$10061.83
Potential cost savings with model:	Potential cost savings with model:
$24742.01	$50436.71
percentage saved:	**percentage saved:**
68.54%	**83.37%**

The potential costs savings of the model was estimated to be $24742.01 a year for offline transactions and $50436.71 a year for online transactions. This is a decrease of 68.54% and 83.37% respectively for offline and online transactions compared to the amount that would have been lost to fraud if no model was used. Therefore, deploying the machine learning models would be highly effective in minimising financial losses due to fraud.

Chapter Five – Recommendations and Conclusion

In this project, several machine learning models were constructed and evaluated based on the F1-score to select the best model for credit card fraud detection. In both the offline and online transactions datasets, the random forest model was the best performing model, outperforming other models such as SVC, logistic regression and neural networks. The hyperparameters and decision threshold of the random forest models were then tuned to further improve their F1-score, precision and recall, with the best model achieving F1-scores of 0.8499 and 0.8931 for the offline and online transactions dataset respectively.

This model optimisation process results in the model being able achieve a recall of 78.11% and 81.78% in the offline and online transactions dataset respectively, meaning that about 80% of

fraudulent transactions can be caught. Additionally, the precision of both models was relatively high, exceeding 90% , signalling the possibility of using the model to notify fraud investigators and user promptly without overwhelming them with false positives.

Both models for the online and offline transactions dataset were also able to exceed the business objective of achieving a F1-score over 80% and reducing the financial losses due to fraud. To assess the business impact, several assumptions were made about the dataset to estimate the monetary value saved by implementing the models. Assuming the worst-case scenario of 100% loss of the transaction value when fraud occurs, and a small loss of 5% of the transaction value when flagging a false positive, deploying the machine learning models for offline transactions and online transactions would substantially reduce the costs due to fraud by up to 68.54% and 83.37% respectively.

Limitations

The project does not consider some features used in traditional fraud detection, such as behavioural analysis of the users and analysis of their transactions patterns. Additionally, due to the under-sampling technique, many of the legitimate transactions were removed to balance the datasets. This could cause some bias in the feature importances, as the models can overfit on the smaller under sampled datasets.

Recommendations for Deployment

Based on the features (user demographics, merchant profiles and transaction details) required and the model performance, the model could be deployed at the bank or card issuer level. Banks would have access to customer demographics and merchant profiles, as well as transaction information to provide the data required by the models for fraud classification.

The model can be incorporated into the backend system of the bank and scheduled to process mini batches of transactions periodically as they occur. This interval can be once a minute to once every few minutes, depending on the volume of transactions. Notifications to both cardholders and the bank's fraud investigation teams can be sent out if a suspected fraudulent transaction is detected. Once deployed, the model performance should be monitored and retrained periodically as new data becomes available to keep up to date with new fraud patterns.

The bank can also consider applying penalties to errant online merchants. Merchant_id was identified the most important feature for the online transactions dataset, this could be due to some merchants intentionally being lax in security. Applying penalties to this group of merchants with high fraud rates could tackle the root cause of the problem and reduce the number of the fraud cases.

Future work

Due to constraints of time and computational resources, hyperparameter tuning of other models such as SVC and neural networks were not carried out. Future work could investigate the performance of these models after hyperparameter tuning and decision threshold tuning to see if they could outperform the random forest models.

The use of machine learning for fraud detection also introduces a potential issue with regulatory compliance as regulators require explanations to justify why certain transactions are classified as fraud (Nayab, Ahmad, Sunday, & Mooale, 2025). While feature importance helps to understand which features are the most influential on the model, it does not provide local explainability for a specific transaction. Future work on the fraud detection system could incorporate the use of explainable AI techniques such as SHapley Additive exPlanations (SHAP) and Local Interpretable Model-Agnostic Explanations (LIME) to improve the explainability of machine learning models like random forest.

References

Afriyie, J., Tawiah, K., Pels, W. A., Addai-Henne, S., Dwamena, H. A., Owiredu, E. O., . . . Eshun, J. (2023). A supervised machine learning algorithm for detecting and predicting fraud in credit card transactions. , 6, 100163. do. *Decision Analytics Journal*, 6, 100163. doi:10.1016/j.dajour.2023.100163.

Capital One Shopping. (19 August, 2024). *U.S. Cashless Statistics*. Retrieved from Capital One Shopping.: https://capitaloneshopping.com/research/cashless-statistics

Frbservices.org. (14 June, 2024). *2024 Diary of Consumer Payment Choice*. Retrieved from Frbservices.org: https://www.frbservices.org/news/research/2024-findings-from-the-diary-of-consumer-payment-choice

Hernandez Aros, L., Bustamante Molano, L., Gutierrez-Portela, F., Moreno Hernandez, J., & Rodríguez Barrero , M. (2024). Financial fraud detection through the application of machine learning techniques: a literature review. *Humanities and Social Sciences Communications*, 11(1), 1130. https://doi.org/10.1057/s41599-024-03606-0.

Lindemulder, G., & Kosinski, M. (29 May, 2024). *What is fraud detection?. "Fraud Detection." Ibm.com, 29 May 2024, www.ibm.com/think/topics/fraud-detection*. Retrieved from Ibm.com: https://www.ibm.com/think/topics/fraud-detection

Madarshahian, R. (4 April, 2024). *Precision & Recall: When Conventional Fraud Metrics Fall Short*. Retrieved from Kount: https://kount.com/blog/precision-recall-when-conventional-fraud-metrics-fall-short

Martínez, P., Forradellas, R., Gallastegui, L., & Alonso, S. (2025). Comparative analysis of machine learning models for the detection of fraudulent banking transactions. *Cogent Business & Management*, 12(1), 2474209. doi:10.1080/23311975.2025.2474209.

Nayab, F., Ahmad, T., Sunday, O., & Mooale, G. (2025). Explainable AI (XAI) for Fraud Detection: Building Trust and Transparency in AI-Driven Financial Security Systems. *ResearchGate*.

Olushola , A., & Mart, J. (2024). Fraud Detection using Machine Learning. *ScienceOpen Preprints.*, DOI: 10.14293/PR2199.000647.v1.