Repository Name:

Hybrid Machine Learning Models for Churn Prediction and Customer Behaviour Analytics Using Big Data

GitHub:

@aimancsv

Table of Contents

1.0 Introduction

Customer churn, the phenomenon where customers discontinue their relationship with a bank, is a significant challenge in the financial industry. In the competitive world of banking, customer retention is not just important; it is crucial to financial stability and growth. When customers leave, it directly impacts a bank's revenue and can lead to substantial financial losses (Peng, Peng, & Li, 2023). Therefore, understanding the factors that lead to customer churn and being able to predict it accurately is essential. By identifying which customers are likely to leave, banks can take proactive measures to address their concerns, improve their experience, and ultimately retain their business (Ahmad, Jafar, & Aljoumaa, 2019). Churn prediction is about analysing patterns in customer behaviour that signal an increased likelihood of attrition, allowing banks to intervene before it's too late.

In today's banking landscape, the challenge of retaining customers has never been greater. Factors such as market saturation, fluctuating economic conditions, and increasingly high customer expectations make it difficult for banks to maintain their customer base (Ennew, Binks, & Chiplin, 2015). While acquiring new customers is important, it is often much more cost-effective to retain existing ones, as the costs associated with attracting new customers are significantly higher. This makes the ability to predict and prevent churn a critical component of a bank's strategy. By accurately predicting which customers are likely to churn, banks can focus their efforts on retention strategies that are both targeted and cost-effective, ensuring they keep their most valuable customers.

Data analysis and machine learning techniques can help address customer churn and improve service quality in the banking industry (Jing, 2023). By leveraging vast amounts of customer data, these technologies can identify patterns and trends that are not immediately apparent. Predictive models can analyse variables such as transaction frequency, account balances, and customer demographics to determine the likelihood of churn. These insights enable banks to implement personalized retention strategies, such as targeted offers or improved customer service, tailored to the specific needs and behaviours of at-risk customers. As a result, banks can strengthen customer loyalty and enhance their competitive edge in the market.

However, the use of customer data in predictive modeling also comes with significant ethical considerations. It is crucial that the analysis respects customer privacy and complies with all relevant regulations. The data must be handled with the utmost care to ensure that customer information is protected and used responsibly. This project is committed to maintaining high standards of data ethics and integrity, focusing on providing actionable insights that can enhance customer retention strategies without compromising customer trust. By balancing the need for effective churn prediction with a strong commitment to ethical data practices, the project aims to deliver valuable insights while safeguarding the privacy and rights of customers.

1.1 Aim & Objectives

Aim

The aim of this project is to develop accurate and reliable predictive models for forecasting customer churn in the banking sector, enabling effective retention strategies.

Objectives

- To analyze the impact of demographics and customer tenure on churn rates.
- To assess financial activity patterns and their role in predicting customer churn.
- To examine the correlation between temporal transaction patterns and churn.
- To evaluate the influence of customer net worth and geographic factors on churn.

These objectives aim to identify the key factors that drive customer churn in the banking sector. By looking at demographics and how long customers have been with the bank, the project seeks to understand how characteristics like age, gender, and customer tenure affect the likelihood of customers leaving. Additionally, analyzing financial activity patterns will provide insights into how customers' spending and saving behaviors might signal their intention to churn.

The project also explores the timing of customer transactions, investigating how the frequency and timing of these activities might correlate with churn. Finally, it evaluates how factors like customer net worth and geographic location influence churn, helping to understand

the role of wealth and where customers live in their decision to stay or leave. Altogether, these objectives guide the creation of predictive models that will help banks identify which customers are at risk of leaving and enable them to take proactive steps to keep those customers engaged.

2.0 Big Data Analytics Lifecycle & Methodology

The CRISP-DM (Cross-Industry Standard Process for Data Mining) methodology is a widely recognized and structured framework used across various industries for data mining and analytics projects. Its adaptability and thoroughness make it a valuable tool for addressing complex data-related challenges. CRISP-DM offers a clear, step-by-step process that helps practitioners navigate the complexities of data analysis with greater ease and efficiency. This methodology is composed of six key phases: business understanding, data understanding, data preparation, modeling, evaluation, and deployment. Each phase is carefully designed to ensure a systematic and consistent approach, which is especially crucial when working with large and intricate datasets commonly encountered in data mining projects. Due to its comprehensive and methodical nature, the CRISP-DM methodology has been chosen for this project, as it provides a well-organized structure that enhances the effectiveness of data mining efforts, ensuring that all aspects of the project are approached with precision and clarity.

2.1 Business Understanding

The Business Understanding phase is the cornerstone of the CRISP-DM methodology. It is where the project's objectives are aligned with the strategic goals of the organization. In the context of predicting customer churn, this phase involves deeply understanding the factors that influence customer behaviour and identifying the business problem that the model aims to solve. The first step is to clarify and define the business objectives, such as reducing churn rates and improving customer retention strategies. Once these objectives are clearly understood, they are translated into specific data mining goals, such as identifying key predictors of churn and developing a model that can accurately forecast which customers are at risk. The final step in this phase is creating a preliminary project plan, which outlines the overall approach, including the phases, tasks, timelines, and responsibilities. This plan also considers the tools and techniques that will be used in the analysis, ensuring that the project is well-aligned with the business's strategic needs and is set up for success.

2.2 Data Understanding

Following the business understanding, the data understanding phase involves getting familiar with the data that will be used in the analysis. The churn prediction dataset used in this study was obtained from GitHub and comprises raw data from 28,382 customers, including personal information, transaction histories, and bank-specific details crucial for understanding customer behaviour. This phase is crucial as it sets the foundation for all data-related activities that follow. The process begins with the initial collection of this data. The next step is to thoroughly profile the data, examining its structure, identifying any data quality issues, and gaining initial insights that might inform the modeling process. Understanding the dataset involves looking at the types of variables available, the distribution of data across these variables, and identifying any anomalies or patterns that could be important for predicting customer churn. This phase is essential for ensuring that the data is well understood and that any potential issues are addressed early on, paving the way for a smooth data preparation process.

2.3 Data Preparation

The data preparation phase is often the most time-consuming and detailed part of the data mining process. This phase involves transforming the raw data into a format that is suitable for analysis, requiring careful attention to detail. The first task is data cleaning, which is critical for ensuring the accuracy and reliability of the model. Using advanced tools like Python and its libraries such as pandas and NumPy, the project team addresses issues like missing values, data inconsistencies, and outliers. These libraries provide powerful functions for managing large datasets, enabling the team to clean the data effectively and efficiently. After cleaning, the data is further processed through steps such as normalization, scaling, and feature engineering. These steps are crucial for enhancing the data's analytical value and ensuring that the model can accurately predict customer churn. Throughout this phase, it is important to maintain a focus on the project's objectives, as the decisions made here will directly impact the model's performance.

In summary, the CRISP-DM methodology provides a robust framework for tackling the complex task of data mining in the banking sector. Each phase, from business understanding to data preparation, is designed to ensure that the project is aligned with the business's goals, that the data is well understood and prepared, and that the model is built on a solid foundation. This structured approach is essential for developing reliable models that can deliver actionable insights and drive strategic decision-making.

2.4 Modelling

Once the data is cleaned and well-prepared, the next crucial phase is to develop a robust model that can accurately predict customer churn. This stage begins with selecting the most appropriate modeling techniques and tools, considering the specific characteristics of the data and the goals of the analysis.

The first step involves choosing from a range of modeling methods, each offering distinct advantages depending on the nature of the data and the desired outcomes. Factors such as the type of data, the complexity of the relationships within the data, and the interpretability of the model are critical in making this decision.

After selecting the suitable algorithm, the next step is to create a detailed plan for model development and testing. This involves splitting the data into training and testing sets, which allows for an unbiased assessment of the model’s performance. The training set is used to build the model, enabling it to learn from the underlying patterns and relationships within the data. During this process, hyperparameters are tuned, and various configurations are tested to optimize the model’s performance.

Once the model is trained, the focus shifts to evaluating its accuracy and reliability. The model is tested on the testing set, and key metrics such as accuracy, precision, recall, and F1-score are calculated. These metrics provide valuable feedback, guiding further refinements to the model to ensure it can reliably predict customer churn and generalize well to new, unseen data.

2.5 Evaluation

As the model is evaluated, its effectiveness in predicting customer churn becomes evident, highlighting its potential impact on strategic business decisions. The model's ability to provide accurate forecasts in real-time makes it an essential tool for improving customer retention strategies.

One significant application of the model is in identifying customers who are at a high risk of churning. By integrating the model into customer relationship management systems, businesses can proactively engage with these customers through targeted marketing efforts or personalized offers. This predictive capability helps in reducing churn rates and maintaining a stable customer base.

Furthermore, the model's continuous monitoring feature allows businesses to track its performance over time. By regularly evaluating how well the model's predictions align with actual outcomes, companies can refine their customer retention strategies and make informed adjustments to their approach. This ongoing evaluation is crucial for maintaining the model's accuracy and relevance in a dynamic business environment.

Beyond just predicting churn, the model can also be used to assess the effectiveness of various customer engagement strategies. By analysing the impact of different marketing campaigns or customer service initiatives on churn rates, businesses can determine which approaches are most successful in retaining customers. This data-driven insight allows for more strategic allocation of resources towards the most effective retention efforts.

Additionally, the model provides valuable insights into customer behaviour patterns, which can inform broader business strategies. Understanding the factors that contribute to customer churn enables businesses to identify areas for improvement in their products or services, ultimately leading to enhanced customer satisfaction and loyalty.

3.0 Data Understanding

To achieve the objectives of predicting customer churn, a thorough understanding of the dataset is essential. Data understanding involves not only gathering and processing the data but also interpreting it to inform the subsequent stages of analysis. The dataset used in this project is sourced from GitHub and contains 28,382 rows, providing a rich set of features for analysis. This dataset includes a diverse array of variables such as personal demographics, transaction histories, and various financial metrics, each of which plays a critical role in understanding customer behaviour and predicting churn.

The dataset includes multiple data types, including integers, floats, and categorical variables, which reflect the broad range of information available for analysis. Understanding these data types and their distributions is key to developing effective predictive models. It also helps in identifying any potential issues such as missing values or outliers that may need to be addressed during the data preparation phase.

In this phase, initial data exploration is conducted to gain insights into the structure and quality of the data. This includes profiling the data to understand the distribution of values across different variables and identifying any anomalies or patterns that could influence the predictive modeling process. A thorough understanding of the data ensures that the subsequent steps in the analysis are built on a solid foundation, leading to more accurate and reliable predictions.

```
dataset.head()
```

	customer_id	vintage	age	gender	dependents	occupation	city	customer_nw_category	branch_code	current_balance	...	average_monthly_balance_prevQ
0	1	2101	66	Male	0.0	self_employed	187.0	2	755	1458.71	...	1458.71
1	2	2348	35	Male	0.0	self_employed	NaN	2	3214	5390.37	...	7799.26
2	4	2194	31	Male	0.0	salaried	146.0	2	41	3913.16	...	4910.17
3	5	2329	90	NaN	NaN	self_employed	1020.0	2	582	2291.91	...	2084.54
4	6	1579	42	Male	2.0	self_employed	1494.0	3	388	927.72	...	1643.31

Figure 1 Preview of the First Five Rows of the Dataset

The image above provides a snapshot of the first five rows from the dataset using the `.head()` function. This initial glimpse helps to understand the dataset's structure and the nature of the data, offering a quick overview before deeper analysis begins.

```
dataset.tail()
```

	customer_id	vintage	age	gender	dependents	occupation	city	customer_nw_category	branch_code	current_balance	...	average_monthly_balance_prevQ
28377	30297	2325	10	Female	0.0	student	1020.0	2	1207	1076.43	...	2282.19
28378	30298	1537	34	Female	0.0	self_employed	1046.0	2	223	3844.10	...	3668.83
28379	30299	2376	47	Male	0.0	salaried	1096.0	2	588	65511.97	...	53444.81
28380	30300	1745	50	Male	3.0	self_employed	1219.0	3	274	1625.55	...	1683.20
28381	30301	1175	18	Male	0.0	student	1232.0	2	474	2107.05	...	3213.44

Figure 2 Preview of the Last Five Rows of the Dataset

The image displays the last five rows of the dataset, generated using the `.tail()` function. This preview allows for a quick review of the dataset's ending portion, helping to confirm the dataset's structure and completeness before proceeding with further analysis.

```
dataset.shape
```

```
(28382, 21)
```

Figure 3 Dataset Dimensions

The image above shows the output of the `.shape` function, which indicates that the dataset contains 28,382 rows and 21 columns. This summary provides a quick overview of the dataset's size, giving insight into the volume of data available for analysis.

```
dataset.columns
```

```
Index(['customer_id', 'vintage', 'age', 'gender', 'dependents', 'occupation',
       'city', 'customer_nw_category', 'branch_code', 'current_balance',
       'previous_month_end_balance', 'average_monthly_balance_prevQ',
       'average_monthly_balance_prevQ2', 'current_month_credit',
       'previous_month_credit', 'current_month_debit', 'previous_month_debit',
       'current_month_balance', 'previous_month_balance', 'churn',
       'last_transaction'],
      dtype='object')
```

Figure 4 Dataset Columns

The image above shows the output of the `.columns` function, which lists all the column names in the dataset. This output provides a complete overview of the data features available for analysis, detailing each variable and its data type. Understanding these columns is essential for the subsequent steps of data exploration and modeling.

```
dataset.dtypes
```

```
customer_id                    int64
vintage                        int64
age                            int64
gender                        object
dependents                   float64
occupation                    object
city                         float64
customer_nw_category           int64
branch_code                    int64
current_balance              float64
previous_month_end_balance   float64
average_monthly_balance_prevQ  float64
average_monthly_balance_prevQ2 float64
current_month_credit         float64
previous_month_credit        float64
current_month_debit          float64
previous_month_debit         float64
current_month_balance        float64
previous_month_balance       float64
churn                          int64
last_transaction              object
dtype: object
```

Figure 5 Dataset Data Types

The image above displays the output of the `.dtypes` function, which lists the data types of each column in the dataset. This information is crucial for understanding how each variable is stored and processed within the dataset. For instance, numerical data like customer balances and ages are stored as `int64` or `float64`, while categorical data such as gender and occupation are stored as `object`. Knowing these data types is essential for proper data manipulation, analysis, and modeling, ensuring that the right methods and techniques are applied to each variable.

```
dataset.isnull().sum().sort_values(ascending=False)
```

```
last_transaction                3223
dependents                      2463
city                             803
gender                           525
occupation                        80
current_month_credit               0
churn                              0
previous_month_balance             0
current_month_balance              0
previous_month_debit               0
current_month_debit                0
previous_month_credit              0
customer_id                        0
average_monthly_balance_prevQ2     0
average_monthly_balance_prevQ      0
vintage                            0
current_balance                    0
branch_code                        0
customer_nw_category               0
age                                0
previous_month_end_balance         0
dtype: int64
```

Figure 6 Missing Values in the Dataset

The image above displays the output of a command used to identify and count missing values in each column of the dataset. The command sorts the columns in descending order of missing values, making it easy to identify which columns have the most missing data. For instance, the `last_transaction` and `dependents` columns have the highest number of missing values, with 3,223 and 2,463 missing entries, respectively. This insight is crucial for data cleaning and preparation, as handling missing data is essential for ensuring the accuracy and reliability of any subsequent analysis or modeling.

3.1 Data Visualization (Pre-Cleaning)

Before diving into data cleaning, visualizing the raw data can reveal valuable insights that guide the analysis process. For this project, data visualization was performed using Python in VS Code, leveraging the powerful libraries Matplotlib and Seaborn. These tools were employed to create various plots that highlight key characteristics of the dataset. This initial visualization provides a foundational understanding of the data's structure and potential trends, which will be further refined through subsequent cleaning and analysis.

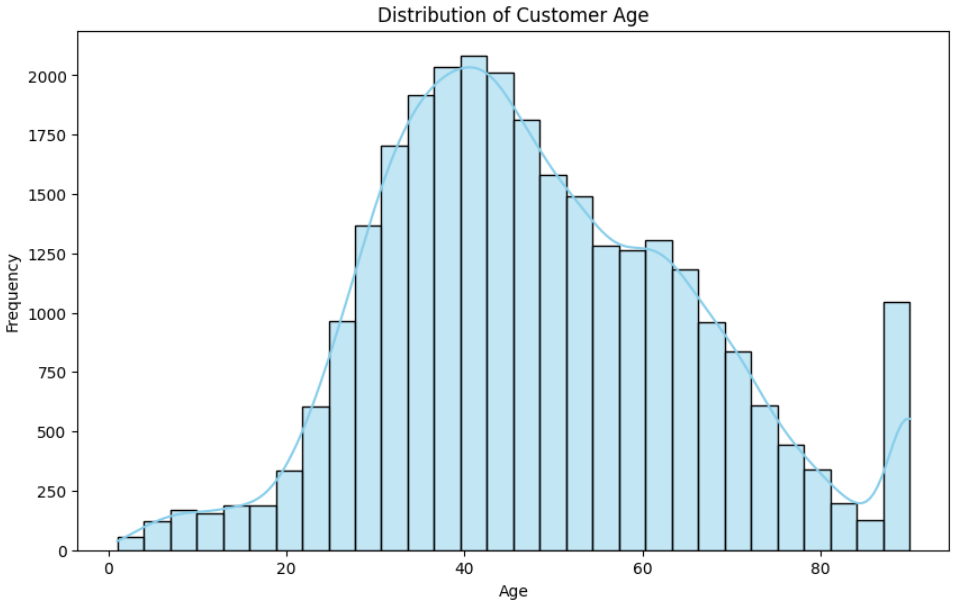

Figure 7 Distribution of Customer Age

The histogram above illustrates the distribution of customer ages in the dataset. The plot reveals a relatively symmetrical distribution centered around the ages of 40 to 45, indicating that a significant portion of the customer base falls within this age range. As age increases beyond this range, the number of customers gradually declines. However, there is a slight uptick in the number of customers in their late 80s, suggesting the presence of a small but notable segment of older customers. This distribution provides valuable insight into the age demographics of the customer base, which is crucial for understanding the diverse needs and behaviours of different age groups within the bank's clientele.

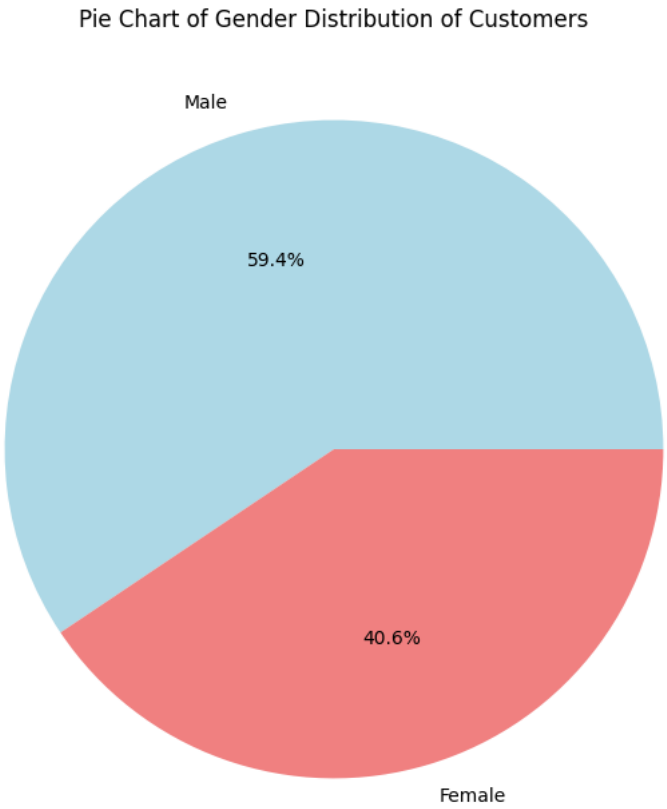

Figure 8 Pie Chart of Gender Distribution of Customers

The pie chart above displays the gender distribution of the bank's customer base. The chart shows that 59.4% of the customers are male, while 40.6% are female. This indicates that the male customer segment is significantly larger than the female segment. Understanding this gender distribution is crucial for tailoring marketing strategies, products, and services to better meet the needs and preferences of the bank's diverse clientele.

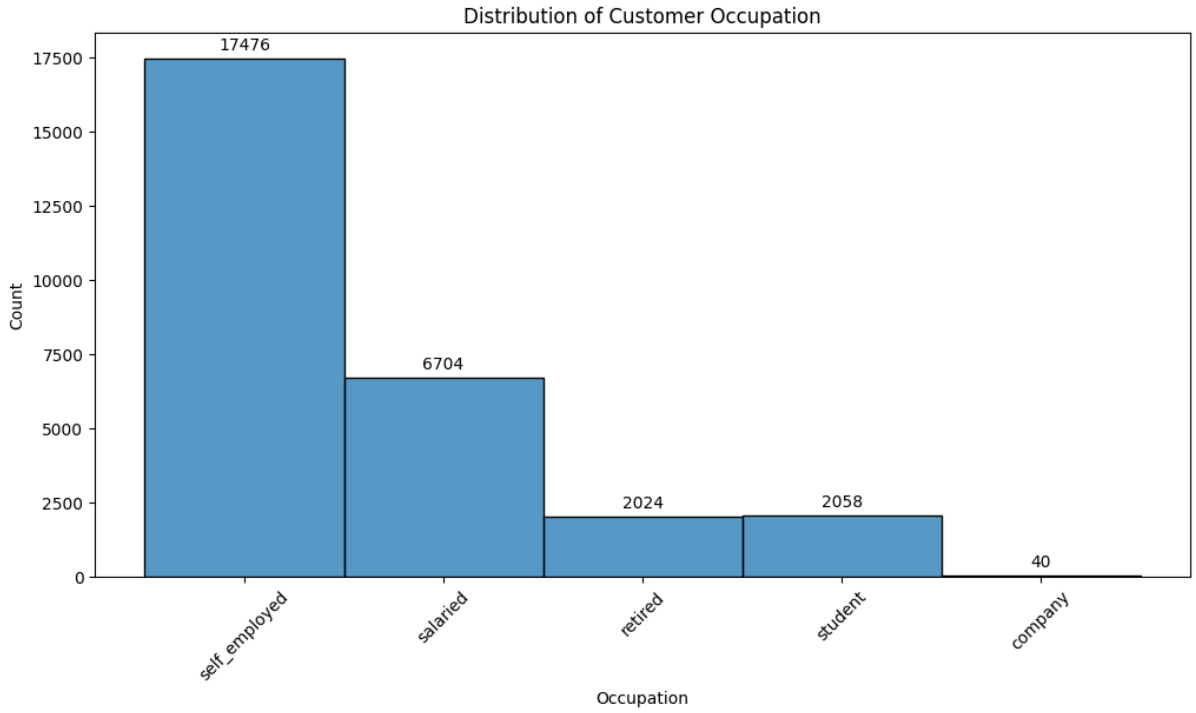

Figure 9 Distribution of Customer Occupation

The histogram above illustrates the distribution of customer occupations within the dataset. The majority of customers, 17,476 in total, are self-employed, making this the largest occupation group. The next significant group comprises salaried individuals, numbering 6,704. Smaller segments include retired customers and students, with 2,024 and 2,058 individuals respectively. The smallest group, with only 40 customers, is associated with companies. This distribution highlights the dominance of self-employed and salaried customers in the dataset, which could influence the financial behaviours and churn patterns observed. Understanding these occupational segments is essential for tailoring financial products and services to meet the distinct needs of each group.

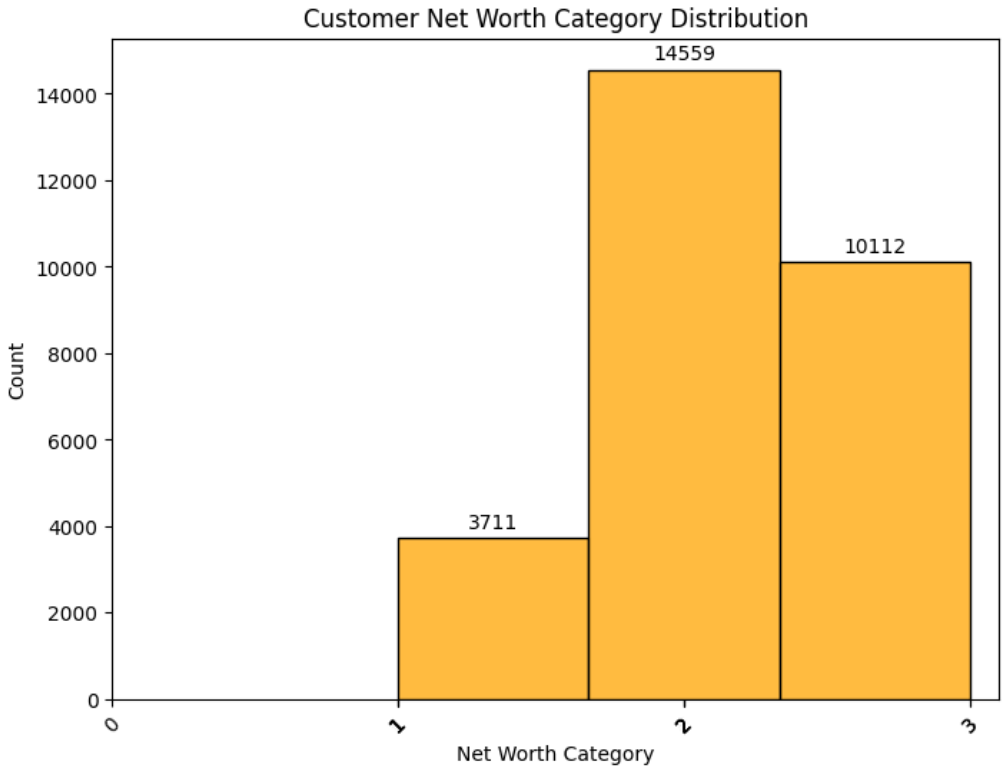

Figure 10 Customer Net Worth Category Distribution

The histogram above presents the distribution of customers across different net worth categories. The largest group falls within category 2, comprising 14,559 customers, indicating that a significant portion of the customer base has moderate net worth. This is followed by category 3 with 10,112 customers, representing the second most prevalent group. Category 1, with only 3,711 customers, represents the smallest segment, indicating fewer customers with lower net worth. Understanding the distribution of customers by net worth is crucial for tailoring financial products and services that meet the specific needs and expectations of each group, thereby enhancing customer satisfaction and retention.

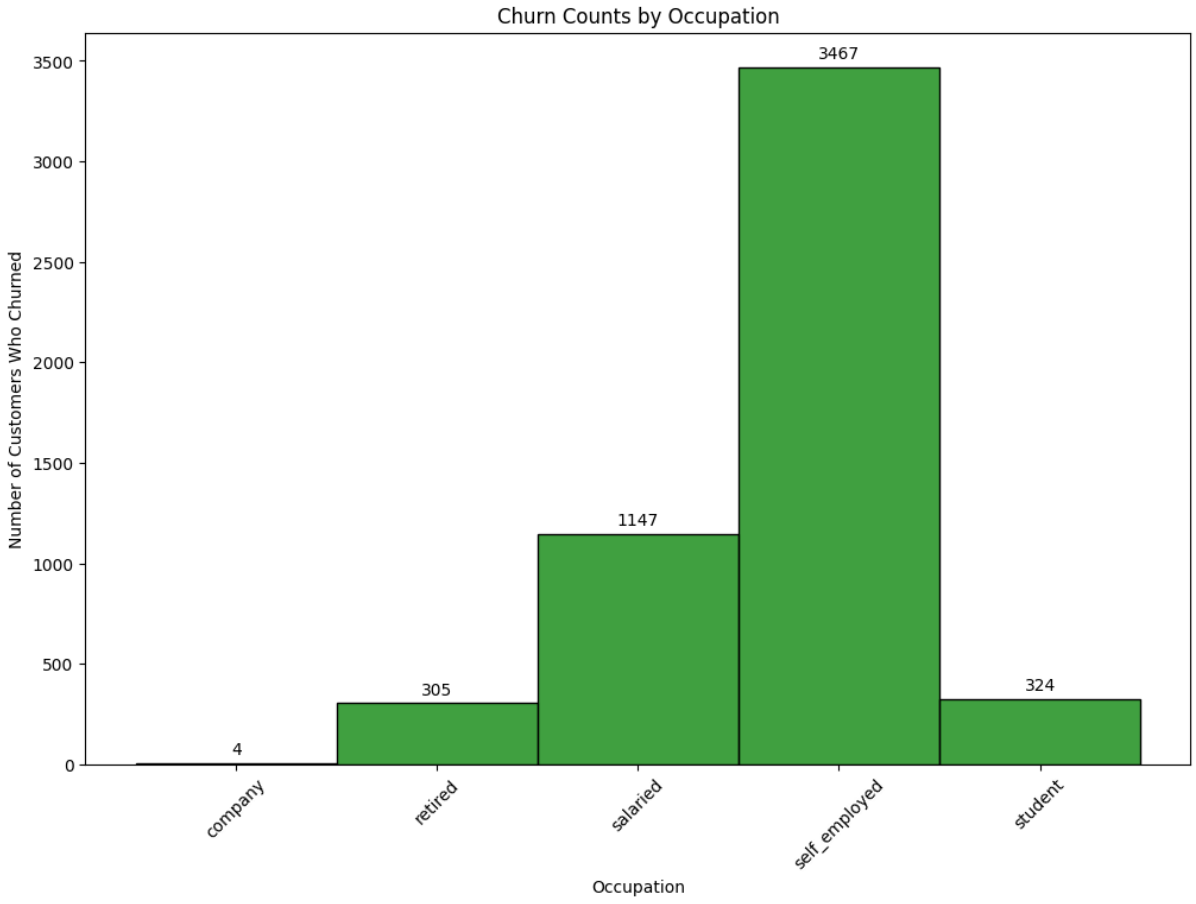

Figure 11 Churn Counts by Occupation

The histogram above illustrates the number of customers who churned, segmented by their occupation. The chart reveals that self-employed customers have the highest churn rate, with 3,467 customers leaving the bank, making them the most significant contributor to churn. Salaried customers are the next highest group with 1,147 customers who churned, followed by retired and student customers with 305 and 324 churns respectively. The 'company' category shows minimal churn, with only 4 customers leaving. This distribution underscores the varying churn tendencies across different occupational groups, suggesting that tailored retention strategies may be needed to address the specific concerns and behaviours of each group.

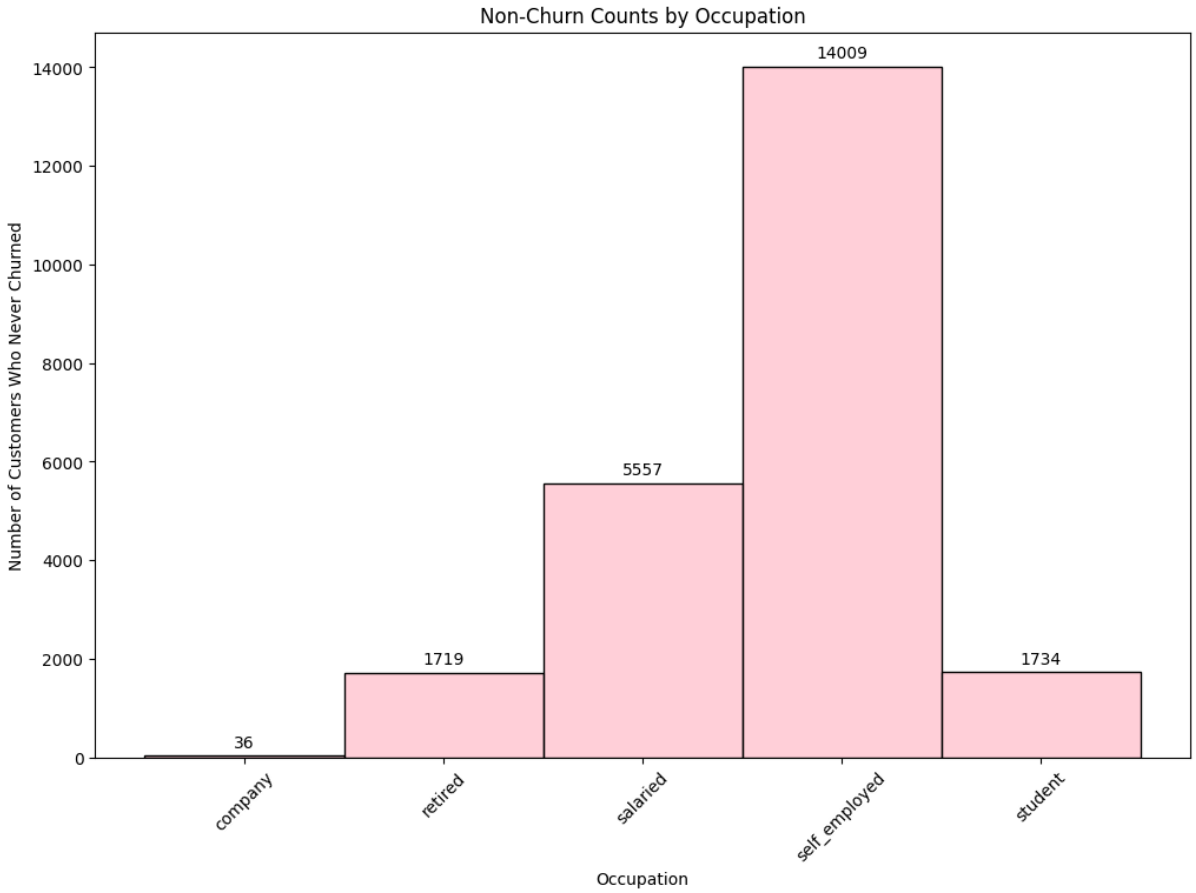

Figure 12 Non-Churn Counts by Occupation

The histogram above displays the number of customers who did not churn, categorized by their occupation. The chart reveals that the majority of non-churning customers are self-employed, with 14,009 individuals remaining loyal to the bank. Salaried customers follow with 5,557 non-churns, while retired and student customers account for 1,719 and 1,734 non-churns respectively. The 'company' category shows the lowest number of non-churns with only 36 individuals. This distribution highlights the varying levels of customer loyalty across different occupational groups, suggesting that certain occupations may have a higher tendency to remain with the bank. Understanding these trends is crucial for developing targeted strategies to maintain customer loyalty across different segments.

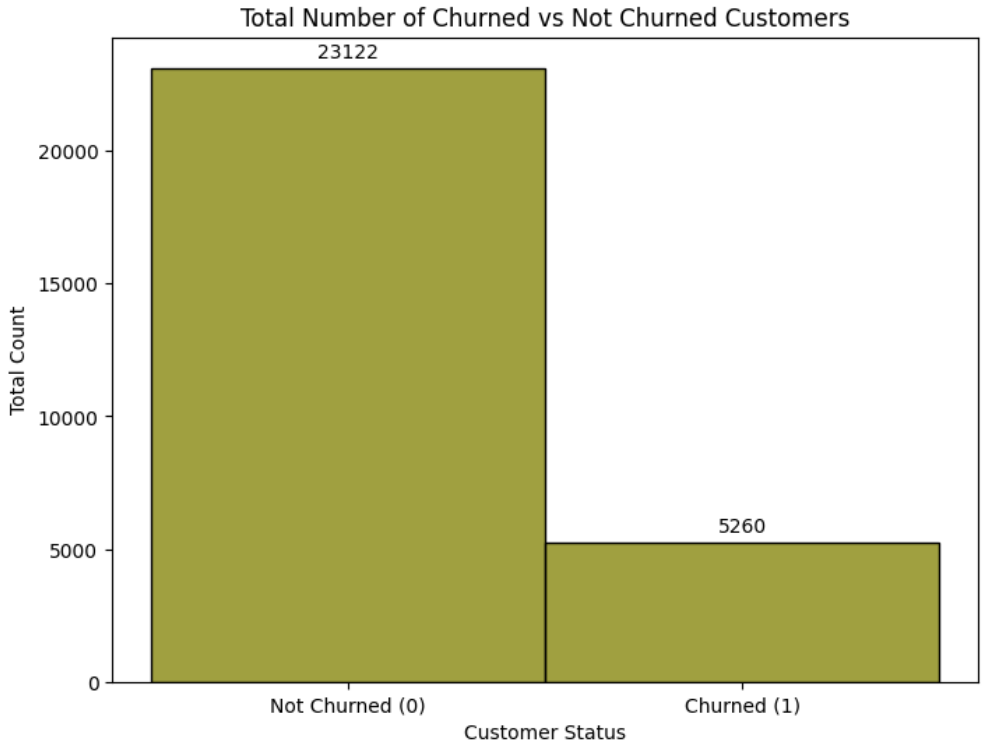

Figure 13 Total Number of Churned vs. Not Churned Customers

The histogram above compares the total number of customers who have churned (left the bank) versus those who have not. The chart shows a significant difference between the two groups, with 23,122 customers remaining loyal to the bank and 5,260 customers who have churned. This stark contrast highlights the proportion of the customer base that the bank has retained compared to those who have left, underscoring the importance of focusing on churn prevention strategies to reduce further attrition.

4.0 Data Preparation

4.1 Data Cleaning

Data cleaning is a crucial part of the data analysis process, particularly when working with large datasets like the one used for predicting customer churn in the banking sector. This step is all about ensuring that the data is accurate, reliable, and ready for analysis. By addressing issues such as missing values, inconsistencies, and outliers, the data becomes more dependable for building predictive models. The data cleaning techniques demonstrated in the code snippets are designed to refine the dataset, making it more suitable for analysis and ultimately leading to more accurate predictions. This careful preparation of data is essential for drawing meaningful insights and making informed decisions that can improve customer retention strategies.

```
dataset = dataset.drop(columns=['customer_id'])
```

Figure 14 Dropping the `customer_id` Column

The code snippet above demonstrates a data cleaning step where the `customer_id` column is removed from the dataset. The `customer_id` is typically a unique identifier that does not contribute to predictive modeling, as it doesn't contain any intrinsic value relevant to customer behaviour or churn prediction. By dropping this column, the dataset is streamlined, focusing on the features that are more likely to provide meaningful insights for analysis and model building. This step is crucial for ensuring that the model is not influenced by unnecessary or non-informative data, which could skew the results and reduce the overall accuracy of predictions.

```python
# Convert 'last_transaction' to datetime
dataset['last_transaction'] = pd.to_datetime(dataset['last_transaction'], errors='coerce')

# Split into separate columns for day, month, and year
dataset['last_transaction_day'] = dataset['last_transaction'].dt.day.astype('Int64')
dataset['last_transaction_month'] = dataset['last_transaction'].dt.month.astype('Int64')
dataset['last_transaction_year'] = dataset['last_transaction'].dt.year.astype('Int64')

# Display the first few rows to check the new columns
print(dataset[['last_transaction_day', 'last_transaction_month', 'last_transaction_year']].head())
```

```
   last_transaction_day  last_transaction_month  last_transaction_year
0                    21                       5                   2019
1                     1                      11                   2019
2                  <NA>                    <NA>                   <NA>
3                     6                       8                   2019
4                     3                      11                   2019
```

Figure 15 Converting and Splitting the `last_transaction` Column

The code snippet above illustrates the process of converting the `last_transaction` column into a datetime format using `pd.to_datetime()`, which is essential for analyzing transaction timing. After conversion, the datetime values are split into three separate columns: `last_transaction_day`, `last_transaction_month`, and `last_transaction_year`. These new columns enable a more detailed analysis of transaction data by allowing examination based on specific days, months, and years. The snippet concludes by displaying the first few rows of the newly created columns, confirming the success of the transformation and preparing the data for further analysis.

```python
dataset = dataset.drop(columns=['last_transaction'])
```

Figure 16 Dropping the Original `last_transaction` Column

The code snippet above demonstrates the removal of the `last_transaction` column from the dataset. After splitting the `last_transaction` data into day, month, and year columns, the original column becomes redundant. Dropping it helps to streamline the dataset, eliminating unnecessary data and reducing potential confusion during subsequent analysis. This step ensures the dataset remains clean and focused on the most relevant information for predictive modeling.

```
dataset.isnull().sum().sort_values(ascending=False)
```

```
last_transaction_year            3223
last_transaction_month           3223
last_transaction_day             3223
dependents                       2463
city                              803
gender                            525
occupation                         80
previous_month_credit               0
churn                               0
previous_month_balance              0
current_month_balance               0
previous_month_debit                0
current_month_debit                 0
vintage                             0
current_month_credit                0
age                                 0
average_monthly_balance_prevQ       0
previous_month_end_balance          0
current_balance                     0
branch_code                         0
customer_nw_category                0
average_monthly_balance_prevQ2      0
dtype: int64
```

Figure 17 Identifying Missing Values in the Dataset

The code snippet above is used to identify and sort the missing values within the dataset, with the results displayed in descending order. The output reveals that certain columns, such as `last_transaction_year`, `last_transaction_month`, and `last_transaction_day`, contain a significant number of missing entries. Other columns, including `dependents`, `city`, and `gender`, also have some missing data, though to a lesser extent. Recognizing and quantifying these missing values is an important step in the data cleaning process, as it helps in deciding how to address these gaps, whether through imputation, removal, or other data preprocessing techniques. Proper handling of missing data is essential to ensure the integrity and reliability of the analyses and predictive models that follow.

```
dataset['last_transaction_year'].fillna(dataset['last_transaction_year'].mode()[0], inplace=True)
dataset['last_transaction_month'].fillna(dataset['last_transaction_month'].mode()[0], inplace=True)
dataset['last_transaction_day'].fillna(dataset['last_transaction_day'].mode()[0], inplace=True)
```

Figure 18 Filling Missing Values in 'last_transaction' Columns with Mode

```
dataset['dependents'].fillna(dataset['dependents'].mode()[0], inplace=True)
```

Figure 19 Filling Missing Values in 'dependents' with Mode

```
dataset['city'].fillna(dataset['city'].mode()[0], inplace=True)
```

Figure 20 Filling Missing Values in 'city' with Mode

```
dataset['gender'].fillna(dataset['gender'].mode()[0], inplace=True)
```

Figure 21 Filling Missing Values in 'gender' with Mode

```
dataset['occupation'].fillna(dataset['occupation'].mode()[0], inplace=True)
```

Figure 22 Filling Missing Values in 'occupation' with Mode

These code snippets above demonstrate a systematic approach to handling missing data in the dataset. The first snippet fills in the missing values in the 'last_transaction_year', 'last_transaction_month', and 'last_transaction_day' columns using the most frequently occurring value (mode) within each column. This method is effective for categorical or ordinal data, ensuring that the majority trend is followed. The subsequent snippets apply the same technique to other columns, including 'dependents', 'city', 'gender', and 'occupation'. By replacing missing values with the mode, the dataset is prepared for analysis without introducing bias that could occur with random or average imputation, ensuring that the most representative values are used for further analysis.

```
dataset.isnull().sum().sort_values(ascending=False)
```

```
vintage                          0
age                              0
last_transaction_month           0
last_transaction_day             0
churn                            0
previous_month_balance           0
current_month_balance            0
previous_month_debit             0
current_month_debit              0
previous_month_credit            0
current_month_credit             0
average_monthly_balance_prevQ2   0
average_monthly_balance_prevQ    0
previous_month_end_balance       0
current_balance                  0
branch_code                      0
customer_nw_category             0
city                             0
occupation                       0
dependents                       0
gender                           0
last_transaction_year            0
dtype: int64
```

Figure 23 Final Check for Missing Values in the Dataset

The code snippet above performs a final check for missing values in the dataset, confirming that all missing values have been addressed. The output shows that every column now has zero missing values, indicating that the dataset is fully prepared for the next steps in the analysis.

```
# Splitting the 'gender' column into two separate columns: 'is_male' and 'is_female'
dataset['is_male'] = (dataset['gender'] == 'Male').astype(int)
dataset['is_female'] = (dataset['gender'] == 'Female').astype(int)
```

Figure 24 Splitting the Gender Column into Binary Indicators

The code snippet above splits the original 'gender' column into two separate binary columns: 'is_male' and 'is_female'. Each new column contains binary values (0 or 1), where 1 indicates the presence of the corresponding gender and 0 indicates its absence. This

transformation is useful for simplifying gender data, making it easier to use in machine learning models.

```
# Displaying the first few rows to check the new columns
print(dataset[['gender', 'is_male', 'is_female']].head(20))
```

```
    gender  is_male  is_female
0     Male        1          0
1     Male        1          0
2     Male        1          0
3     Male        1          0
4     Male        1          0
5   Female        0          1
6     Male        1          0
7     Male        1          0
8     Male        1          0
9     Male        1          0
10    Male        1          0
11    Male        1          0
12    Male        1          0
13    Male        1          0
14  Female        0          1
15  Female        0          1
16    Male        1          0
17  Female        0          1
18  Female        0          1
19  Female        0          1
```

Figure 25 Verifying Binary Gender Columns

The code snippet above displays the first 20 rows of the newly created binary columns, 'is_male' and 'is_female', alongside the original 'gender' column. The output confirms that the transformation correctly assigns a 1 to the appropriate binary column based on the customer's gender, ensuring the data is accurately prepared for further analysis or modeling.

```
dataset = dataset.drop(columns=['gender'])
```

Figure 26 Dropping the Original Gender Column

The code snippet above removes the original 'gender' column from the dataset after successfully splitting it into binary columns 'is_male' and 'is_female'. This step ensures that the dataset is clean and only contains the necessary features for subsequent analysis and modeling.

```
dataset = pd.get_dummies(dataset, columns=['occupation'], prefix='occupation')
```

Figure 27 One-Hot Encoding the Occupation Column

The code snippet above uses the get_dummies function to convert the 'occupation' column into multiple binary (0 or 1) columns, each representing a different occupation category. This process, known as one-hot encoding, is essential for preparing categorical data for machine learning models, which typically require numerical input.

```
dataset.columns
```

```
Index(['vintage', 'age', 'dependents', 'city', 'customer_nw_category',
       'branch_code', 'current_balance', 'previous_month_end_balance',
       'average_monthly_balance_prevQ', 'average_monthly_balance_prevQ2',
       'current_month_credit', 'previous_month_credit', 'current_month_debit',
       'previous_month_debit', 'current_month_balance',
       'previous_month_balance', 'churn', 'last_transaction_day',
       'last_transaction_month', 'last_transaction_year', 'is_male',
       'is_female', 'occupation_company', 'occupation_retired',
       'occupation_salaried', 'occupation_self_employed',
       'occupation_student'],
      dtype='object')
```

Figure 28 Displaying Updated Dataset Columns After Feature Engineering

```
dataset.dtypes
```

```
vintage                          int64
age                              int64
dependents                     float64
city                           float64
customer_nw_category             int64
branch_code                      int64
current_balance                float64
previous_month_end_balance     float64
average_monthly_balance_prevQ  float64
average_monthly_balance_prevQ2 float64
current_month_credit           float64
previous_month_credit          float64
current_month_debit            float64
previous_month_debit           float64
current_month_balance          float64
previous_month_balance         float64
churn                            int64
last_transaction_day             Int64
last_transaction_month           Int64
last_transaction_year            Int64
is_male                          int32
is_female                        int32
occupation_company                bool
occupation_retired                bool
occupation_salaried               bool
occupation_self_employed          bool
occupation_student                bool
dtype: object
```

Figure 29 Updated Data Types of Features After Data Cleaning and Feature Engineering

```
# Convert boolean columns to integers
bool_columns = ['occupation_company', 'occupation_retired', 'occupation_salaried',
                'occupation_self_employed', 'occupation_student']
dataset[bool_columns] = dataset[bool_columns].astype(int)

# Convert Int64 columns to int64
int_columns = ['last_transaction_day', 'last_transaction_month', 'last_transaction_year']
dataset[int_columns] = dataset[int_columns].astype('int64')
```

Figure 30 Converting Boolean and Integer Columns to Appropriate Data Types

```
dataset.dtypes
```

```
vintage                          int64
age                              int64
dependents                     float64
city                           float64
customer_nw_category             int64
branch_code                      int64
current_balance                float64
previous_month_end_balance     float64
average_monthly_balance_prevQ  float64
average_monthly_balance_prevQ2 float64
current_month_credit           float64
previous_month_credit          float64
current_month_debit            float64
previous_month_debit           float64
current_month_balance          float64
previous_month_balance         float64
churn                            int64
last_transaction_day             int64
last_transaction_month           int64
last_transaction_year            int64
is_male                          int32
is_female                        int32
occupation_company               int32
occupation_retired               int32
occupation_salaried              int32
occupation_self_employed         int32
occupation_student               int32
dtype: object
```

Figure 31 Final Data Types of All Columns After Data Cleaning and Transformation

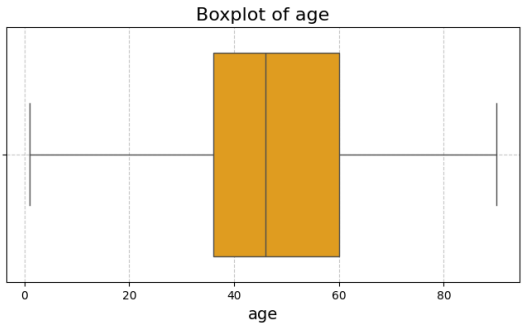

Figure 32 Boxplot of age before outlier removal

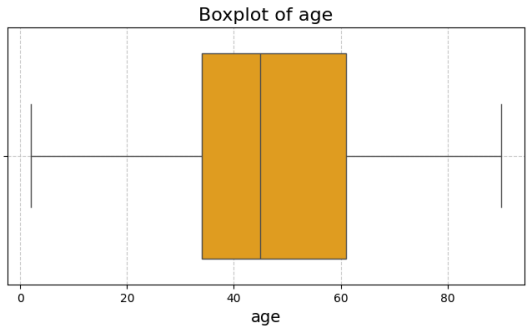

Figure 33 Boxplot of age after outlier removal

The first image displays a boxplot of the "age" variable before any outliers were removed. In this plot, the whiskers extend further out, indicating the presence of some extreme values at both ends of the age range. These outliers represent customers who are either very young or quite old compared to the majority of the dataset. The second image shows the boxplot after these outliers were removed. Here, the whiskers are shorter, reflecting a more concentrated age range and the exclusion of those extreme values. This adjustment helps to narrow the focus to the main body of data, reducing the potential skew and allowing for a more accurate analysis of the central age distribution.

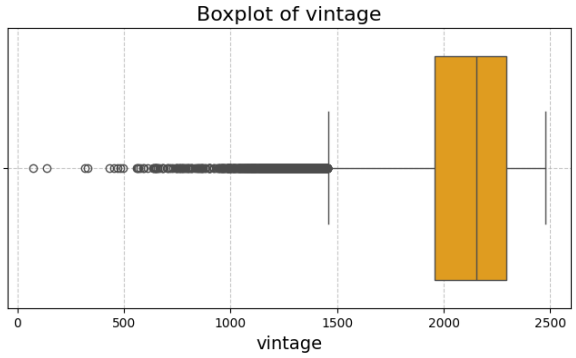

Figure 34 Boxplot of vintage before outlier removal

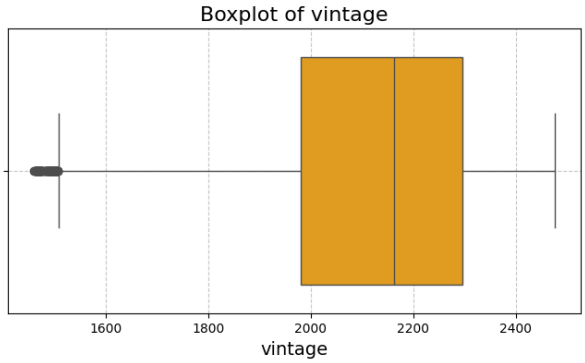

Figure 35 Boxplot of vintage after outlier removal

The first boxplot of the "vintage" variable reveals the presence of numerous outliers on the lower end of the scale, with many values significantly below the median, extending as far down as close to zero. This indicates that there is a substantial subset of customers with a relatively short relationship duration with the bank. These outliers can skew the analysis and affect the model's performance. After applying outlier removal techniques, as shown in the

second boxplot, the data is much more condensed, and the outliers have been effectively removed. The focus is now on a more consistent range of values, with the central tendency of vintage data between 2000 and 2200 days, allowing for a more accurate and reliable analysis.

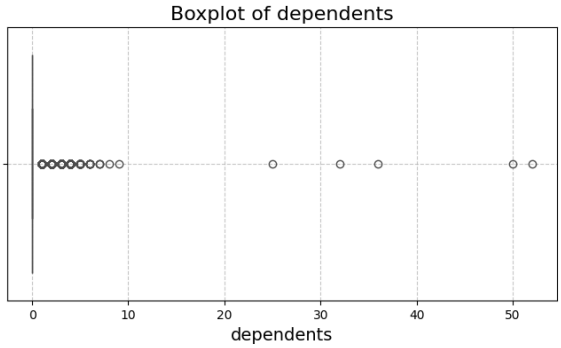

Figure 36 Boxplot of dependents before outlier removal

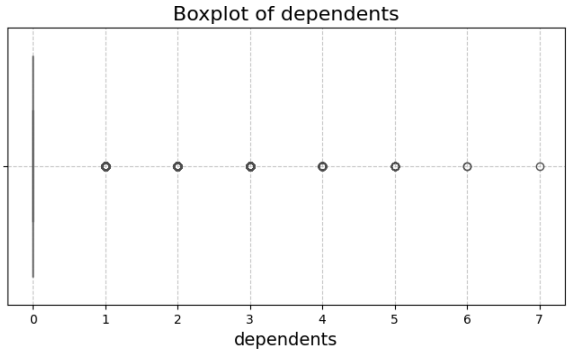

Figure 37 Boxplot of dependents after outlier removal

The first boxplot shows the distribution of the 'dependents' variable before outlier removal, where several extreme values, particularly above 10 dependents, are evident. These values are likely data entry errors or represent atypical cases, leading to a highly skewed visualization. After removing the outliers, as illustrated in the second boxplot, the distribution becomes more normalized and centered around the lower range, providing a more accurate reflection of the typical number of dependents in the dataset. This cleaning step is essential for ensuring that the analysis is not disproportionately influenced by anomalous data points.

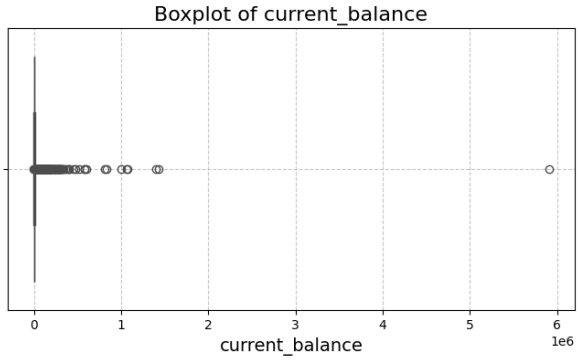

Figure 38 Boxplot of current_balance before outlier removal

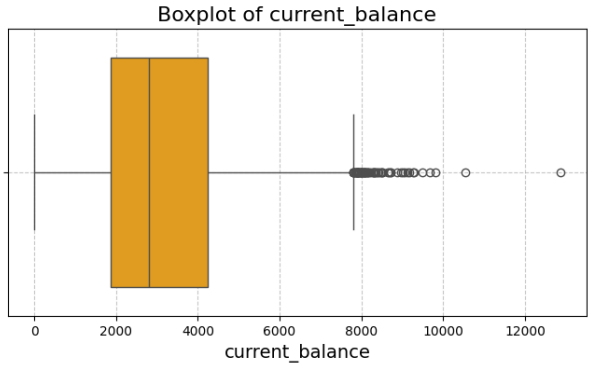

Figure 39 Boxplot of current_balance after outlier removal

In Figure 38, the boxplot shows a significant number of extreme outliers in the current balance, with values ranging up to 6 million. These outliers are widely dispersed, indicating a highly skewed distribution. After removing these extreme outliers, as shown in Figure 39, the boxplot reflects a more compact range, with the bulk of the data centered between approximately 1,000 and 7,000. The presence of fewer outliers in the modified boxplot suggests a more accurate and representative understanding of the typical customer balance within the dataset.

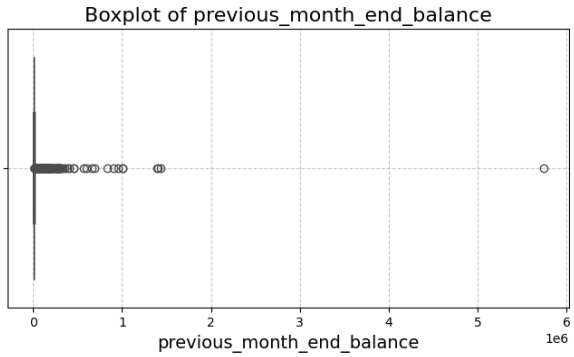

Figure 40 Boxplot of previous_month_end_balance before outlier removal

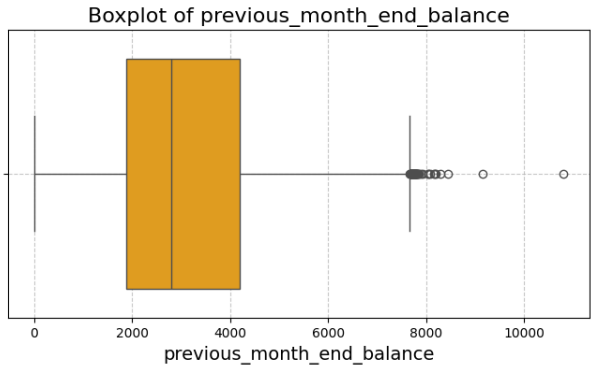

Figure 41 Boxplot of previous_month_end_balance after outlier removal

The two boxplots above show the distribution of the "previous_month_end_balance" variable in the dataset. The first boxplot, which includes outliers, shows a significant number of outliers on the lower end, with extreme values skewing the data. After removing the outliers, the second boxplot reveals a more balanced distribution, with most of the data now

concentrated within a narrower range. The removal of outliers has led to a more accurate representation of the typical "previous_month_end_balance" values, making it easier to detect meaningful patterns and trends in the dataset.

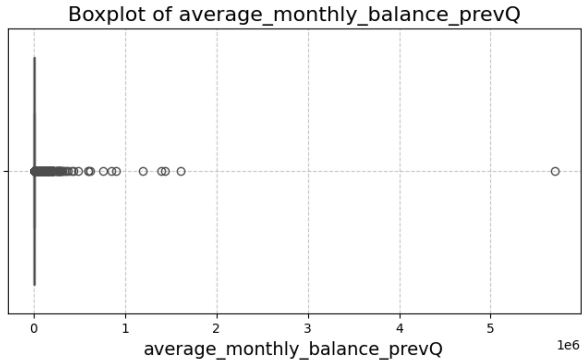

Figure 42 Boxplot of average_monthly_balance_prevQ before outlier removal

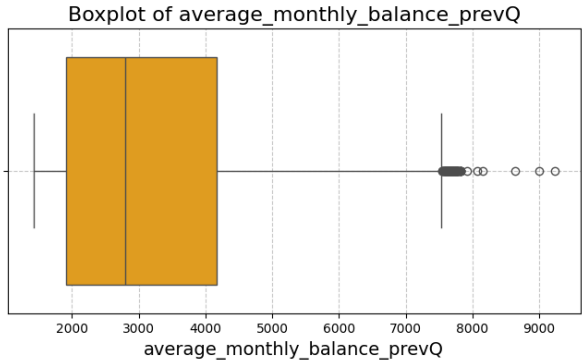

Figure 43 Boxplot of average_monthly_balance_prevQ after outlier removal

The boxplots illustrate the distribution of the average monthly balance from the previous quarter before and after outlier removal. Initially, the data shows a significant presence of outliers, with values extending far beyond the main cluster, indicating extreme deviations. After outlier removal, the distribution becomes more centralized and compact, which enhances the dataset's overall quality and reliability, allowing for more accurate analysis in the subsequent steps.

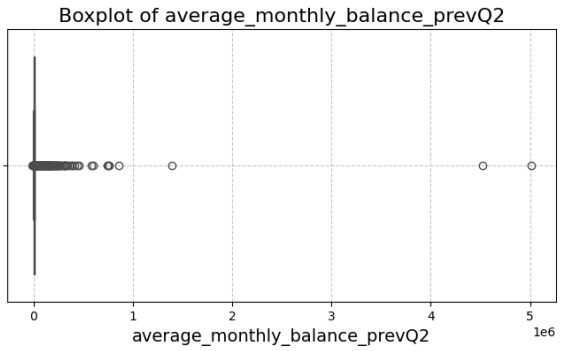

Figure 44 Boxplot of average_monthly_balance_prevQ2 before outlier removal

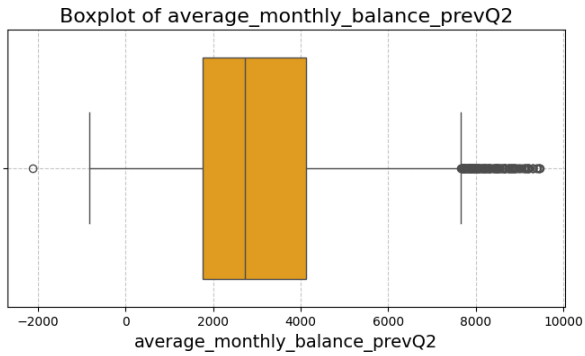

Figure 45 Boxplot of average_monthly_balance_prevQ2 after outlier removal

In the initial boxplot of the "average_monthly_balance_prevQ2" variable, there are several extreme outliers extending far beyond the typical range, indicating a skewed distribution. After addressing these outliers, the subsequent boxplot reveals a more compact and centered distribution with the majority of data points falling within a narrower range. The adjustments result in a cleaner visualization of the data, making the central tendency and variability of "average_monthly_balance_prevQ2" more apparent and meaningful for analysis.

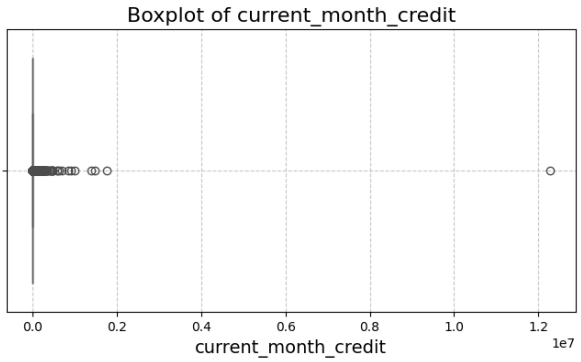

Figure 46 Boxplot of current_month_credit before outlier removal

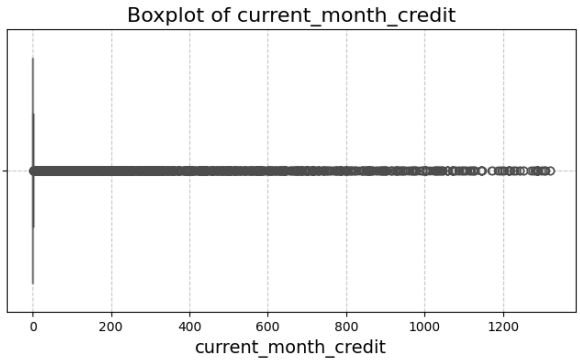

Figure 47 Boxplot of current_month_credit after outlier removal

The first image shows the boxplot of the "current_month_credit" variable before outlier removal, where a few extreme values significantly distort the scale of the plot, making the majority of the data appear compressed towards the lower end of the x-axis. The second image displays the same variable after outlier removal, where the distribution is more spread out, with the majority of the data now clearly visible and without the influence of extreme outliers. This

adjustment provides a clearer view of the central tendency and variability of the data, making the plot more informative for analysis.

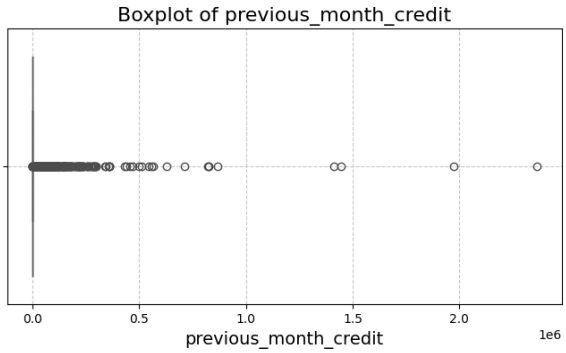

Figure 48 Boxplot of previous_month_credit before outlier removal

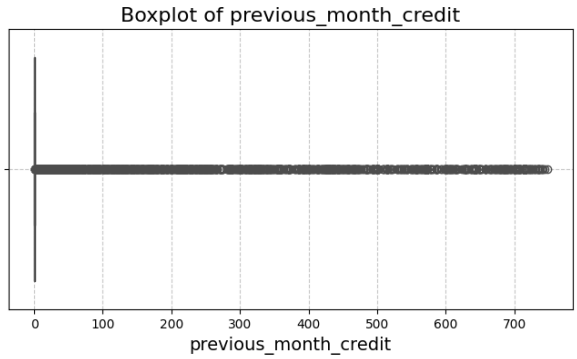

Figure 49 Boxplot of previous_month_credit after outlier removal

The boxplots illustrate the distribution of "previous_month_credit" before and after outlier removal. In the first image, the presence of extreme values is evident, with a significant number of outliers extending far from the main data cluster. These outliers suggest a wide variation in credit amounts for the previous month. After removing the outliers, as seen in the second image, the distribution becomes more condensed, providing a clearer view of the typical credit values without the influence of extreme cases. This refinement in the data helps in achieving more accurate and reliable analysis outcomes.

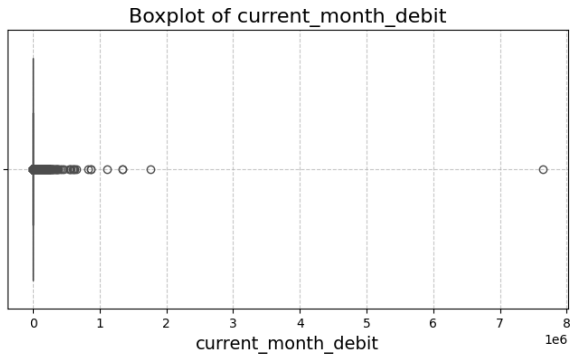

Figure 50 Boxplot of current_month_debit before outlier removal

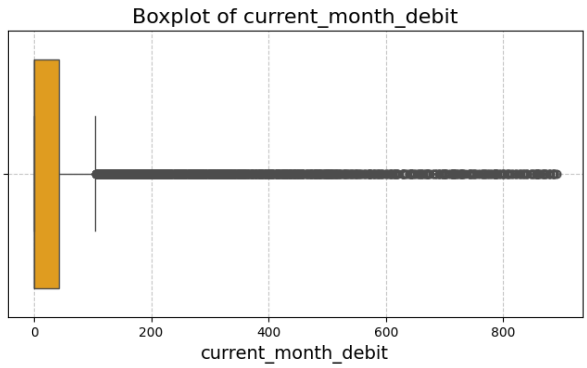

Figure 51 Boxplot of current_month_debit after outlier removal

In these boxplots of "current_month_debit," the first image shows the data before outlier removal, where a significant number of extreme values distort the overall distribution. The second image, post-outlier removal, demonstrates a more focused view of the majority of data points, making the central tendency and spread of the "current_month_debit" values clearer. This refinement is essential for more accurate analysis, as it reduces the skewness caused by outliers, providing a better understanding of typical debit amounts.

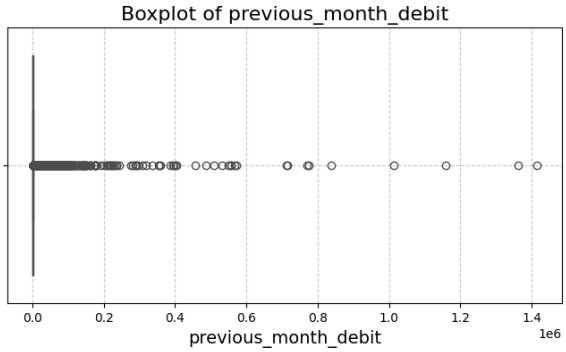

Figure 52 Boxplot of previous_month_debit before outlier removal

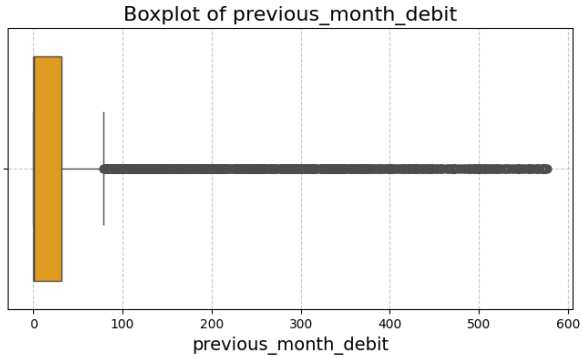

Figure 53 Boxplot of previous_month_debit after outlier removal

The first boxplot for "previous_month_debit" displays significant outliers with values reaching up to 1.4 million, indicating a wide range of debit amounts in the previous month. The second boxplot, post outlier removal, shows a more concentrated distribution around lower debit values, with a much narrower range. This suggests that the majority of customers have a more consistent and lower debit amount, and the removal of extreme outliers provides a clearer view of the common debit behaviour.

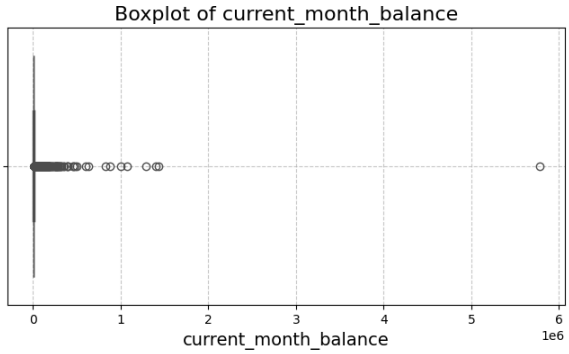

Figure 54 Boxplot of current_month_balance before outlier removal

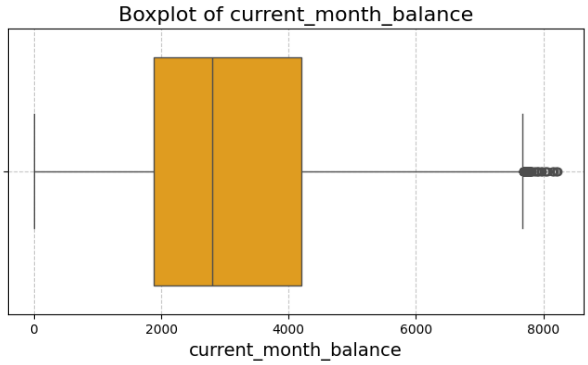

Figure 55 Boxplot of current_month_balance after outlier removal

The provided boxplots of the current month's balance offer a clear visualization of how the data changes before and after outlier removal. In the first plot, before outlier removal, the data is heavily skewed with many outliers extending far beyond the upper quartile, indicating a significant presence of unusually high balances. After outlier removal, as shown in the second plot, the distribution appears more compact, with most data points concentrated around the median, providing a more accurate reflection of the central tendency and reducing the distortion caused by extreme values.

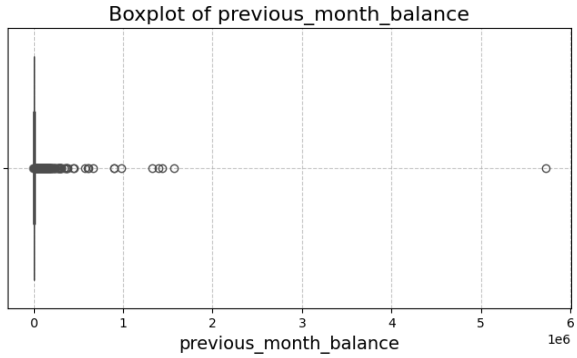

Figure 56 Boxplot of previous_month_balance before outlier removal

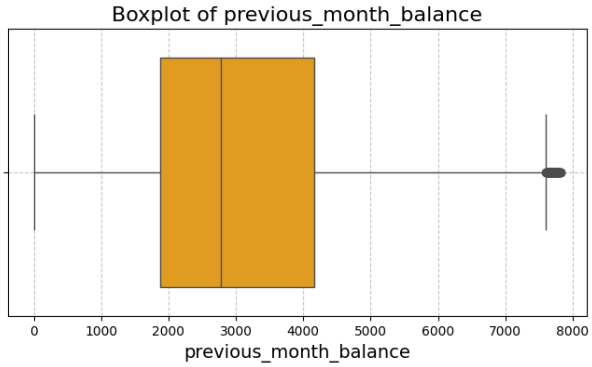

Figure 57 Boxplot of previous_month_balance after outlier removal

The two boxplots compare the distribution of the "previous_month_balance" variable before and after outlier removal. In the first boxplot, a significant number of outliers are evident, particularly on the higher end of the scale, indicating extremely high previous month balances for a few customers. After removing these outliers, the second boxplot presents a much more condensed and symmetric distribution, with the interquartile range (IQR) now being more centered, providing a clearer view of the typical previous month balances within the dataset. This process highlights the importance of outlier detection and removal in achieving a more accurate and interpretable data analysis.

5.0 Modeling & Evaluation

5.1 Assess financial activity patterns and their role in predicting customer churn

The method of evaluation aims to examine how various financial activity patterns relate to customer churn, providing insights into the behaviours that might indicate a higher risk of customers leaving the bank. This analysis is conducted using a Random Forest model, a robust and flexible machine learning technique. The Random Forest model leverages key financial data, including account balances, transaction histories, and average monthly balances over different periods, to predict the likelihood of customer churn. By analysing these financial metrics, the model helps identify customers who are at a greater risk of churn, enabling the bank to implement targeted retention strategies that address specific financial behaviours.

5.1.1 Descriptive Analysis

a. Data Distribution

The histograms below illustrate the distributions of various financial metrics critical to understanding customer behaviour and predicting churn. Each histogram provides insights into key factors like customer balances, transaction amounts, and historical financial trends. The descriptive analysis of these figures helps to identify patterns and anomalies, forming the basis for predicting customer churn and guiding retention strategies.

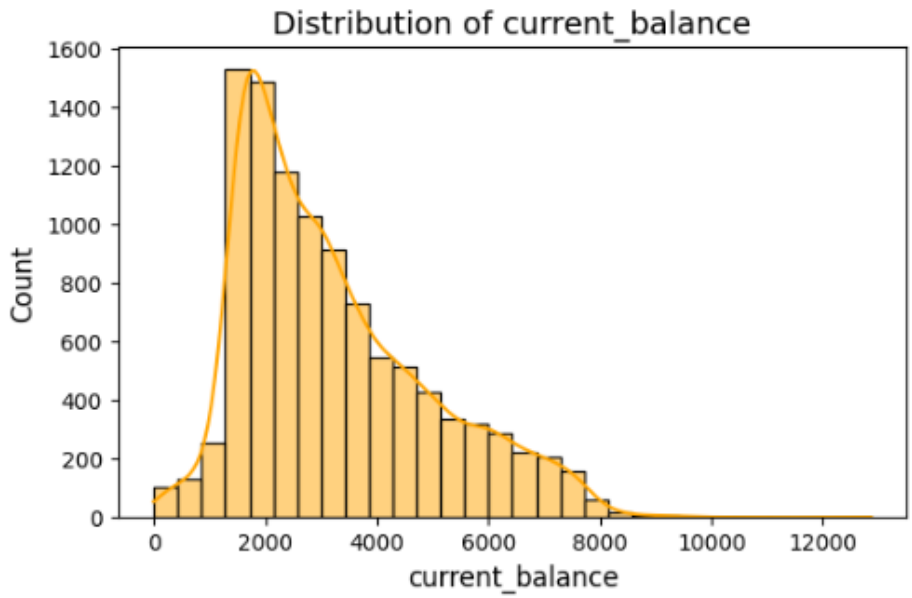

Figure 58 Distribution of current_balance

The distribution of current balance is right-skewed, with most customers having balances concentrated between 0 and 4000 units. The peak of the distribution occurs around the 1000-2000 range, where the highest frequency of balances is observed. There is a gradual decline in the number of customers as the balance increases beyond 4000, with very few customers having balances exceeding 8000.

This distribution suggests that many customers maintain relatively low to moderate account balances, potentially reflecting typical saving and spending habits. The right-skew indicates that while a few customers have very high balances, the majority keep lower amounts in their accounts. This pattern may highlight a common financial behaviour among the customer base, where only a small portion of the population holds significant wealth within their current accounts.

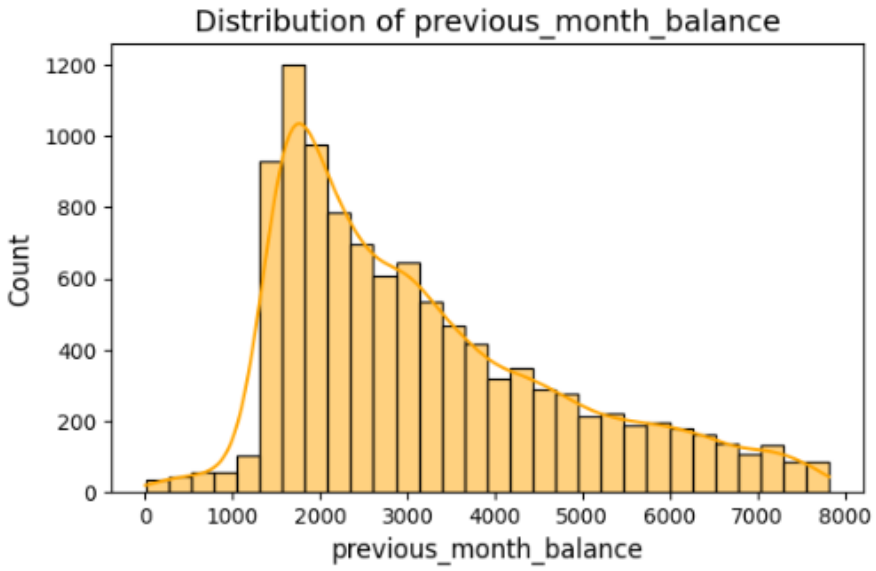

Figure 59 Distribution of previous_month_balance

The distribution of the previous month balance is right-skewed, with most customer balances concentrated between 1000 and 4000 units. The highest frequency occurs around the 2000-3000 range, indicating where the majority of balances were held. The frequency gradually decreases as the balance increases, with fewer customers holding balances above 4000 and even fewer exceeding 7000.

This distribution suggests that, similar to the current month balance, most customers maintain moderate balances in their accounts. The peak around the 2000-3000 range indicates a common threshold for account balances among the customer base. The right-skew indicates that while a few customers have significantly higher balances, the majority tend to keep their balances within a more modest range. This consistency in balance behaviour over time could be indicative of stable financial habits or income patterns among the customer base.

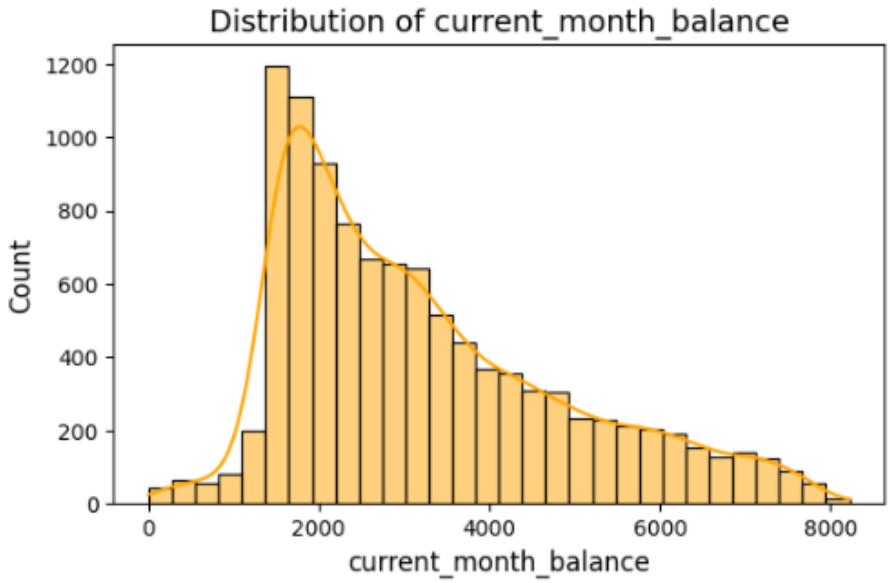

Figure 60 Distribution of current_month_balance

The distribution of the current month balance is right-skewed, with the majority of customer balances concentrated between 1000 and 4000 units. The peak of the distribution occurs around the 2000-3000 range, indicating that most customers have balances within this interval. As the balance amount increases beyond 4000, the frequency of customers holding these balances decreases gradually, with fewer customers having balances close to 8000.

This distribution suggests that most customers maintain moderate account balances, with a concentration around the 2000-3000 range. The right-skewness of the distribution indicates that while there are some customers with higher balances, they are less common. The similarity between the current and previous month balance distributions may imply consistent financial behaviour over time, with customers generally maintaining similar balance levels from month to month. This could be indicative of stable income patterns or spending habits among the customer base.

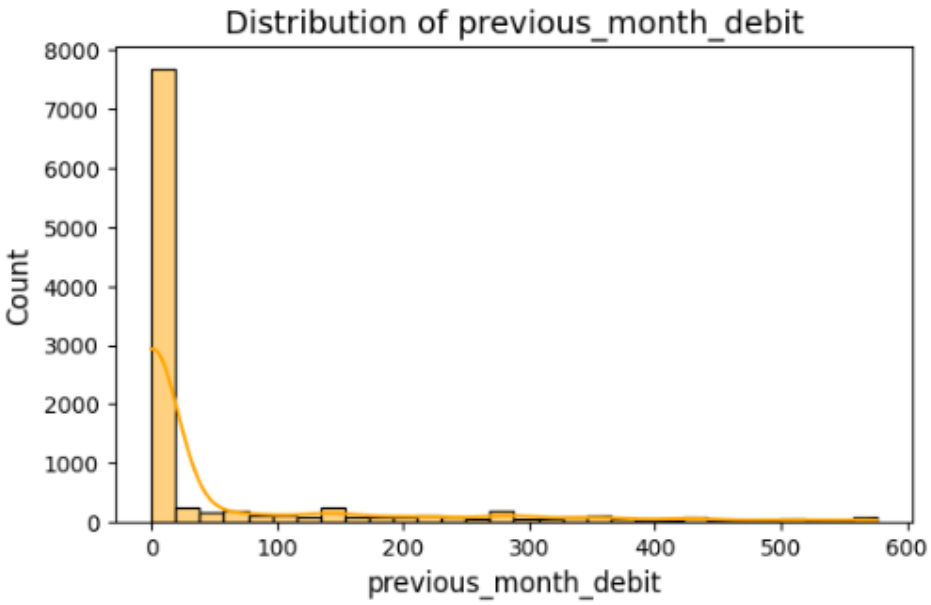

Figure 61 Distribution of previous_month_debit

The distribution of previous month debit transactions is highly right-skewed, with the majority of transactions concentrated at very low values, predominantly below 50. The frequency of transactions decreases sharply as the amount increases, with very few transactions exceeding 100, and only a handful reaching up to 600.

This distribution indicates that most customers engage in small-value debit transactions, likely for everyday expenses or minor payments. The sharp decline in transaction frequency for higher amounts suggests that large debit transactions are uncommon. The pattern is consistent with routine financial behaviour, where customers make frequent small transactions, with only occasional larger debits.

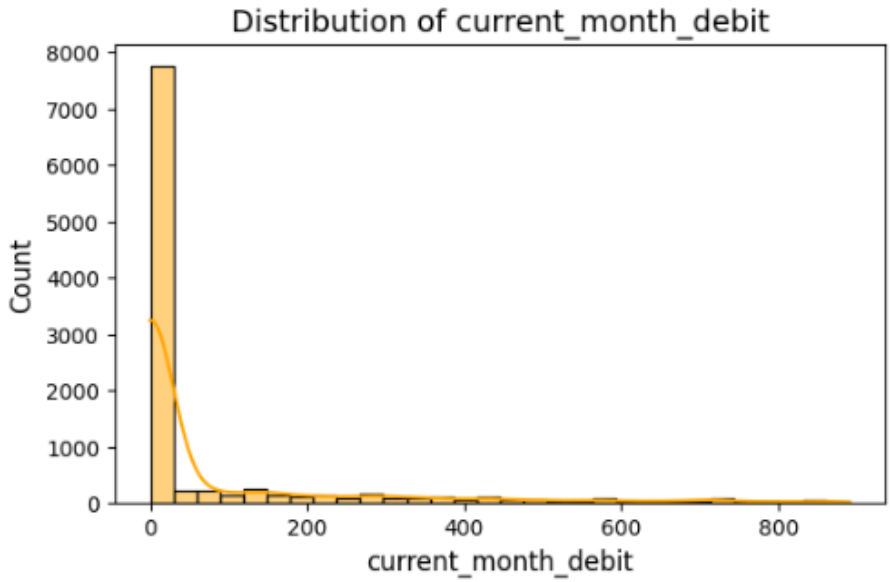

Figure 62 Distribution of current_month_debit

The distribution of current month debit transactions is highly right-skewed, with the majority of transactions concentrated at very low values, predominantly below 50. There is a steep decline in the frequency of transactions as the amount increases, with very few transactions exceeding 100.

This pattern indicates that most customers engage in small-value debit transactions during the current month, possibly for routine expenses or minor purchases. The sharp drop-off in transaction frequency as the amount increases suggests that high-value debits are relatively uncommon. The similarity between the distributions of current and previous month debit transactions implies stable spending behaviour among customers over time, with consistent patterns of small-value transactions.

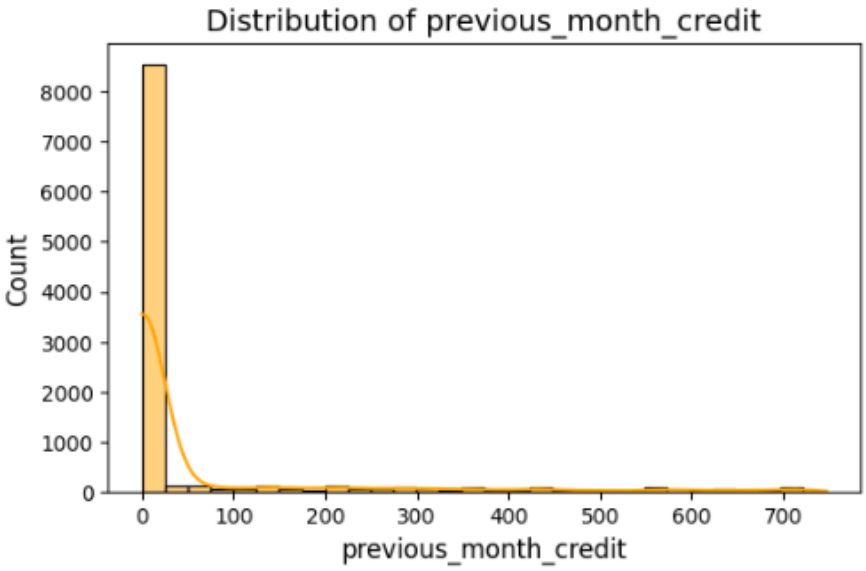

Figure 63 Distribution of previous_month_credit

The distribution of previous month credit transactions is highly right-skewed, with the majority of transactions clustered around very low values, predominantly below 20. Only a very small number of transactions exceed this range, with very few reaching or surpassing 100. The data shows a significant drop-off in frequency as transaction amounts increase.

This distribution suggests that most customers receive very small credits, possibly from minor income sources or routine small deposits. The sharp drop-off in frequency for higher amounts indicates that substantial credits are rare among the customer base. The extreme skewness highlights the concentration of credit transactions at the lower end, suggesting that large credit inflows are uncommon for many customers.

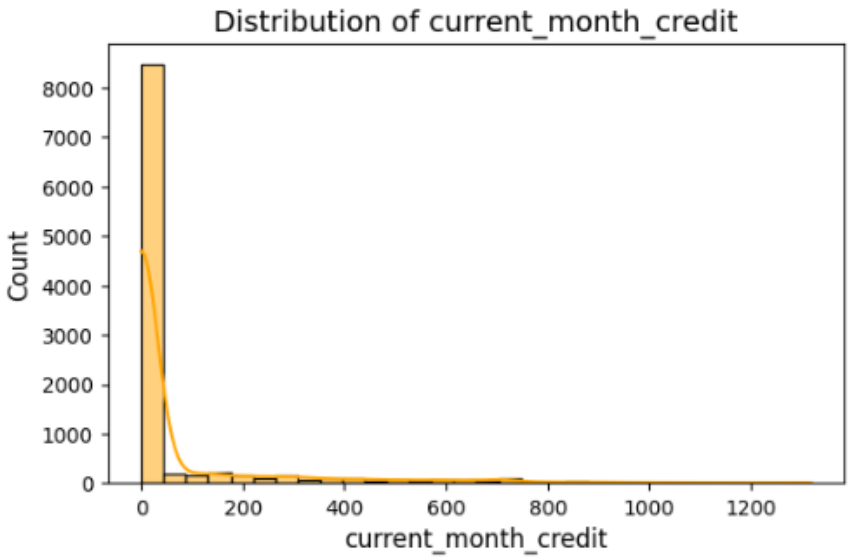

Figure 64 Distribution of current_month_credit

The distribution of current month credit transactions is highly right-skewed, with many transactions clustered around very low values, primarily below 50. There is a sharp decline in the frequency of transactions as the credit amount increases, with very few transactions exceeding 100 and almost none surpassing 200 units.

This distribution suggests that most customers receive small credit amounts during the current month, possibly from routine income sources or regular small deposits. The sharp drop-off in frequency for higher credit amounts indicates that substantial credits are rare among the customer base. This pattern of distribution is consistent with typical financial behaviour, where most customers experience small, regular inflows of money, with only a few receiving significant credits.

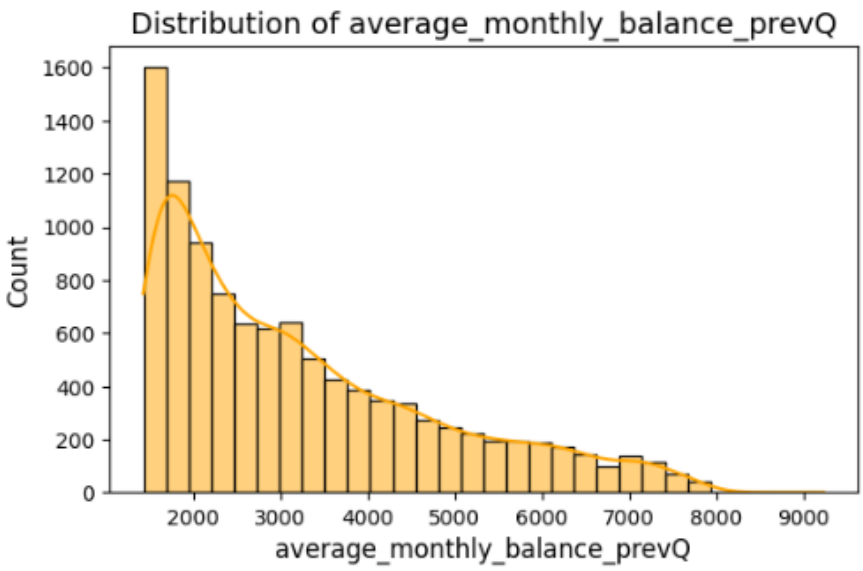

Figure 65 Distribution of average_monthly_balance_prevQ

The distribution of the average monthly balance from the previous quarter is right-skewed, with the majority of customers having average balances concentrated between 2000 and 4000 units. The peak of the distribution is around the 2000-3000 range, indicating the most common average balance. As the average balance increases beyond 4000, the frequency of customers decreases steadily, with fewer customers maintaining higher average balances up to 9000 units.

This distribution suggests that most customers maintain moderate average monthly balances over the previous quarter, with a concentration around the 2000-3000 range. The right-skew indicates that while some customers have higher average balances, they are less common. This pattern is indicative of consistent saving or spending behaviours, where the majority of the customer base manages to maintain a stable average balance over time, while only a smaller segment holds significantly higher balances.

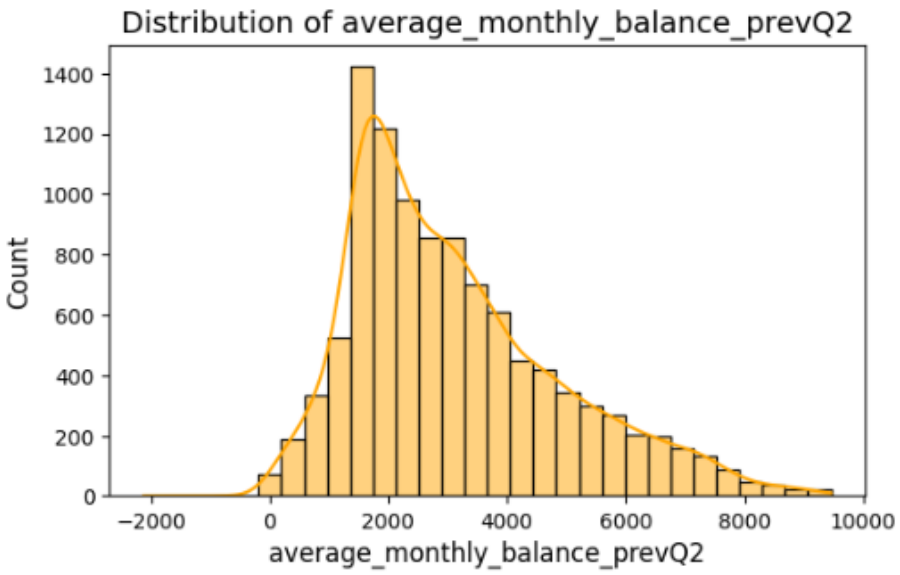

Figure 66 Distribution of average_monthly_balance_prevQ2

The distribution of the average monthly balance from the second previous quarter is right-skewed, with many customer balances concentrated between 1000 and 4000 units. The peak of the distribution occurs around the 2000-3000 range, which is where the highest frequency of average balances is observed. The frequency gradually decreases as the balance increases, with fewer customers maintaining average balances beyond 5000 units. Interestingly, there are also negative values in this distribution, indicating some accounts had average balances below zero during the second previous quarter.

This distribution suggests that most customers maintain moderate average monthly balances over the second previous quarter, with a significant concentration around the 2000-3000 range. The presence of negative balances might indicate instances of overdrafts or negative account balances, suggesting financial instability for a portion of the customer base during that period. The right-skewness of the distribution, combined with the gradual decline in frequency for higher balances, highlights that while some customers maintain higher average balances, they are less common. This pattern indicates that many customers manage to keep their average balances in a moderate range, with only a few experiencing more extreme balances, either high or low.

Customer Balance Patterns:

Most customers tend to maintain their account balances at a moderate level. The most common balance range falls between 2000 and 4000 units, and this pattern is consistent across current balances, previous month balances, and average balances over several months. This stability suggests that many customers manage to keep their balances within a certain range, avoiding significant fluctuations.

Transaction Behaviour:

A noticeable trend is observed towards smaller transaction amounts. Both debit and credit transactions predominantly fall under 50 units, indicating regular, everyday spending and income, such as paying for groceries or receiving paychecks. This pattern remains steady from month to month, reflecting consistent spending habits without significant changes in transaction amounts.

Historical Financial Trends:

Historical data shows that current balance trends have been consistent over time. The average monthly balances from previous quarters exhibit a similar right-skewed distribution, with most customers maintaining balances in the 2000-4000 unit range. While there are some outliers, with a few customers having much higher or, in rare cases, negative balances, these are exceptions. This indicates that while many customers manage their finances steadily, some may occasionally experience financial challenges.

b. Potential Outliers

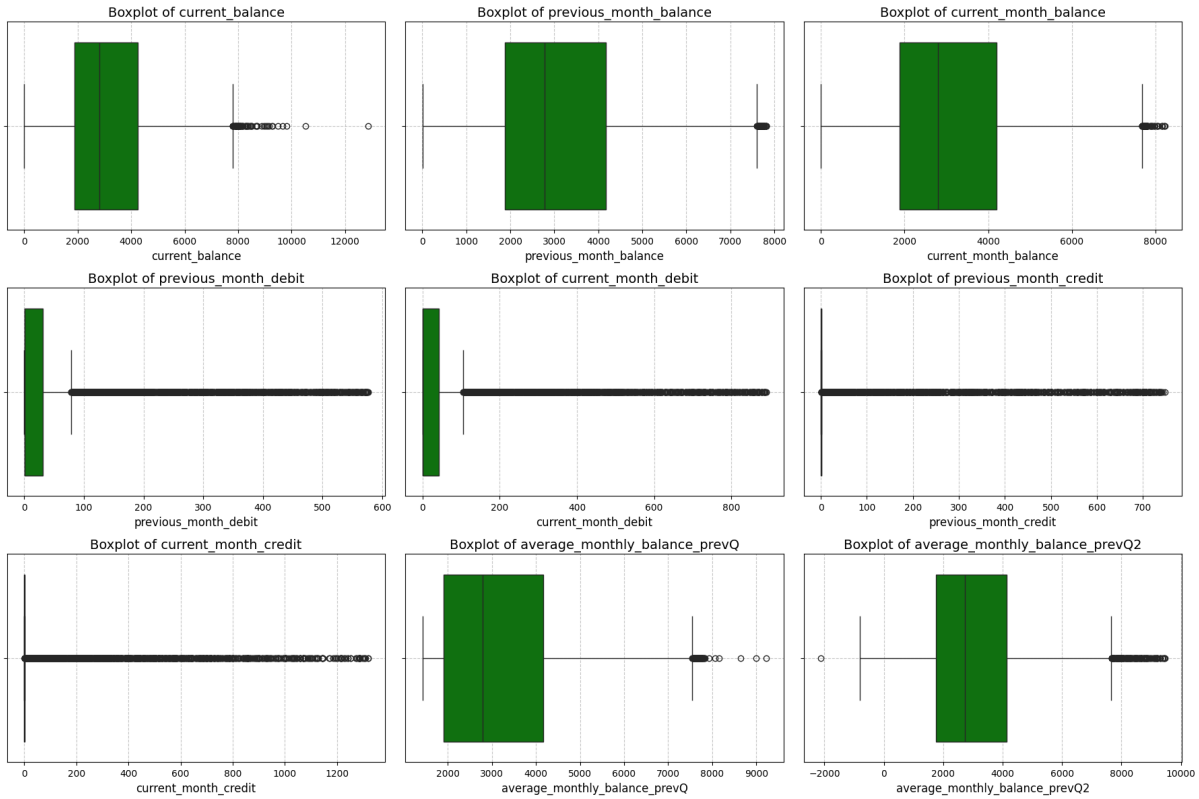

Figure 67 Boxplots illustrating customer financial activity metrics, highlighting central tendency, dispersion, and outliers across balance and transaction data

The boxplots in the figure above illustrate key insights into customer financial behaviour by highlighting the central tendencies, dispersion, and presence of outliers across various financial metrics.

Starting with the balance-related columns (current_balance, previous_month_balance, current_month_balance), most customers maintain moderate balances, typically ranging between 2000 and 4000 units. The consistency of median values across these columns suggests stable financial habits, indicating a core group of customers who use their accounts regularly and predictably. However, there are notable outliers, particularly at the higher end, with balances exceeding 6000 and even reaching up to 12000 units. These outliers likely represent high-value customers with potentially more complex financial needs, who may either be more loyal due to their higher engagement or at higher risk of churn if their needs are not met.

For debit and credit transactions (both current and previous months), the data shows a concentration of small transactions under 100 units, typical of everyday spending or regular income. However, there are also outliers where transaction amounts are significantly higher, indicating that a small number of customers engage in large transactions, possibly reflecting business activities or major expenses. These customers could have different churn dynamics compared to the general customer base.

The average monthly balance columns (average_monthly_balance_prevQ and average_monthly_balance_prevQ2) reflect a similar trend to the current balance data, with most customers falling within the same moderate balance range. However, the variability in these balances and the presence of high-value outliers suggest that some customers maintain significantly higher average balances, possibly due to more stable or higher income sources, which could reduce their likelihood of churn.

Overall, the boxplots provide a clear visual summary of customer financial behaviour, identifying consistent patterns among most customers and significant outliers that could require targeted retention strategies. This analysis underscores the importance of segmenting customers based on their financial behaviour to better predict churn and develop effective customer retention strategies.

c. Correlation Heatmap of Financial Activity Metrics in Relation to Churn Prediction

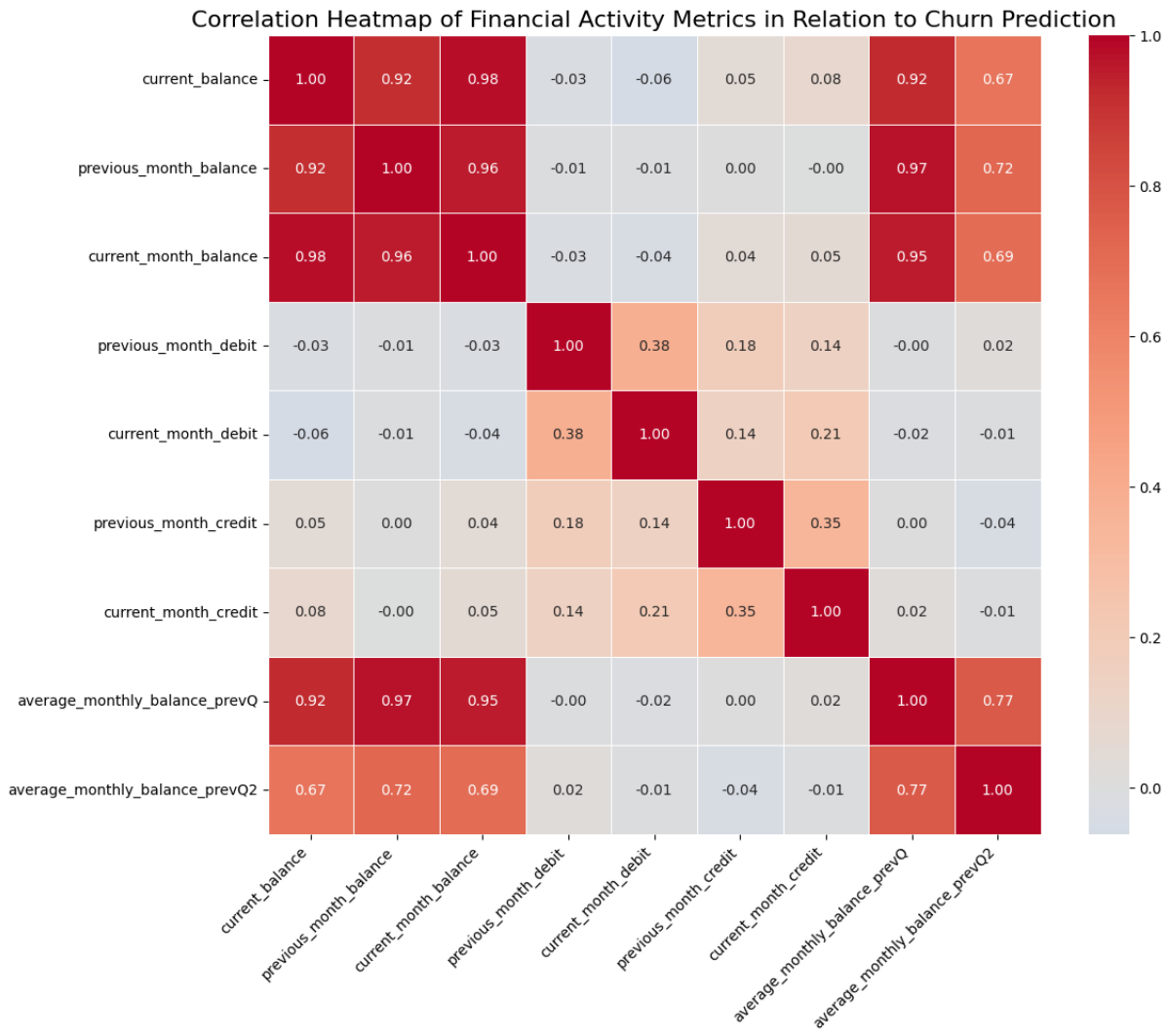

Figure 68 Correlation heatmap of financial activity metrics in relation to churn prediction

The correlation heatmap above provides a detailed view of how various financial activity metrics relate not only to each other but also to the likelihood of customer churn. The heatmap uses a range of correlation coefficients from -1 to 1, where values closer to 1 indicate a strong positive correlation, values closer to -1 indicate a strong negative correlation, and values near 0 suggest little to no linear relationship.

One of the key insights from the heatmap is the relationship between churn and various financial activity metrics. The churn variable shows a weak negative correlation with most balance-related metrics such as current balance, previous month's balance, and current month's

balance. This indicates that customers with higher balances are slightly less likely to churn, although the relationship is not particularly strong. The correlation between churn and transaction-related metrics like previous month's debit and current month's debit is positive but still weak, suggesting that higher debit transactions could slightly increase the likelihood of churn, but the effect is minimal.

The balance-related variables are highly correlated with each other, with correlations often exceeding 0.9. This strong positive correlation suggests that customers who maintain high balances tend to do so consistently across different time periods. However, when considering churn, these balance metrics show only weak correlations, indicating that high balances alone may not be a strong predictor of whether a customer will churn.

In terms of transaction-related variables, the previous month’s debit and current month’s debit show a moderate correlation with each other, implying that customers who make high debit transactions in one month are somewhat likely to continue this behaviour the following month. However, these transaction metrics do not show strong correlations with the churn variable, which suggests that debit and credit amounts may not have a significant direct influence on churn. This could imply that other factors, not captured by these specific financial transactions, are influencing churn.

The weak correlations between churn and the other variables suggest that while financial activity metrics provide some insight, they may not be sufficient on their own to predict churn with high accuracy. The negative correlations with balance metrics suggest that customers who maintain higher balances are somewhat more stable and less likely to churn, but the effect is weak. This might indicate that these customers are generally more satisfied or engaged with the bank, but other factors, potentially outside the scope of the financial metrics analysed, play a crucial role in determining whether they ultimately churn.

In summary, the inclusion of the churn variable in the heatmap has provided valuable insights. It reveals that while there are some correlations between financial activity metrics and churn, these relationships are not strong enough to rely on alone for predictive purposes. The high intercorrelation among balance-related metrics suggests stability in customer financial behaviour, but the weak correlation with churn highlights the need to consider a broader range of features and more advanced analytical techniques when building predictive models for churn.

d. Multivariate Regression Analysis of Financial Activity Metrics for Churn Prediction

OLS Regression Results

Dep. Variable:	churn	R-squared:	0.078
Model:	OLS	Adj. R-squared:	0.077
Method:	Least Squares	F-statistic:	97.94
Date:	Thu, 15 Aug 2024	Prob (F-statistic):	2.03e-176
Time:	23:26:38	Log-Likelihood:	-434.94
No. Observations:	10463	AIC:	889.9
Df Residuals:	10453	BIC:	962.4
Df Model:	9		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	0.0889	0.006	15.143	0.000	0.077	0.100
current_balance	-5.196e-05	7.59e-06	-6.849	0.000	-6.68e-05	-3.71e-05
previous_month_balance	-6.913e-05	7.25e-06	-9.536	0.000	-8.33e-05	-5.49e-05
current_month_balance	-4.516e-05	1.05e-05	-4.302	0.000	-6.57e-05	-2.46e-05
previous_month_debit	2.953e-05	2.12e-05	1.396	0.163	-1.19e-05	7.1e-05
current_month_debit	0.0001	1.53e-05	6.888	0.000	7.54e-05	0.000
previous_month_credit	-1.829e-05	1.77e-05	-1.036	0.300	-5.29e-05	1.63e-05
current_month_credit	-7.98e-07	1.37e-05	-0.058	0.954	-2.76e-05	2.6e-05
average_monthly_balance_prevQ	0.0002	7.63e-06	22.575	0.000	0.000	0.000
average_monthly_balance_prevQ2	-1.474e-05	2.22e-06	-6.634	0.000	-1.91e-05	-1.04e-05

Figure 69 OLS regression results for predicting customer churn based on financial activity metrics

In this analysis, multivariate regression was chosen due to the complexity of the relationships between the dependent variable, churn, and multiple independent financial activity variables. Unlike simple regression, which examines the relationship between a single independent variable and the dependent variable, multivariate regression allows for the assessment of the simultaneous impact of several predictors on churn. This method is particularly useful for understanding how different aspects of financial behaviour, such as balances and transaction amounts, collectively influence the likelihood of customer churn. By considering multiple variables at once, the analysis can identify which factors are most influential and how they interact with each other, providing a more comprehensive understanding of the overall situation.

The output of the regression analysis reveals several key insights. The coefficients represent the estimated change in the churn rate for a one-unit change in each predictor variable, holding all other variables constant. For example, the coefficient for current_balance is -5.196e-05, indicating a slight negative relationship with churn; higher current balances are associated with a lower likelihood of churn. Similarly, previous_month_balance and current_month_balance also show negative coefficients, reinforcing the idea that customers with higher balances are less likely to churn. These coefficients are statistically significant, as indicated by their low p-values (all less than 0.05), suggesting that the relationships between these balance metrics and churn are not due to random chance.

The t-statistic for each coefficient indicates the ratio of the estimated coefficient to its standard error, providing a measure of how many standard deviations the coefficient is away from zero. A high absolute value of the t-statistic, such as -6.849 for current_balance, suggests that the coefficient is significantly different from zero, and therefore, the variable is a significant predictor of churn. The same applies to previous_month_balance with a t-statistic of -9.536 and current_month_balance with -4.302, indicating strong evidence that these variables are significant predictors of churn. On the other hand, variables like previous_month_debit with a t-statistic of 1.396 and a p-value of 0.163 do not provide strong evidence of a significant effect on churn.

The R-squared value of 0.078 indicates that approximately 7.8% of the variance in churn can be explained by the independent variables included in the model. While this is a modest proportion, it suggests that other factors not included in this analysis may play a more significant role in predicting churn. The F-statistic of 97.94 and its associated p-value (2.03e-176) indicate that the model is statistically significant, meaning that at least some of the predictors are related to churn.

In summary, the multivariate regression analysis highlights the relationships between various financial activity metrics and customer churn, with balance-related variables showing the strongest negative association with churn. While the model provides some valuable insights, the relatively low R-squared value suggests that additional variables or more complex models may be needed to better predict customer churn. The significance of the coefficients, their associated t-statistics, and p-values underscores the importance of certain financial behaviours in understanding customer retention, although the impact of these variables is relatively small.

e. OLAP Dashboard for Financial Metrics Distribution

The OLAP Dashboard for Financial Metrics Distribution was developed using the Dash framework in Python, with Plotly used for creating detailed histograms. This interactive dashboard allows for the analysis of various financial metrics, including current balances, monthly debits and credits, and average monthly balances. A dropdown menu provides flexibility in selecting different financial metrics, automatically updating the displayed histogram. The dashboard also includes dynamic interactivity where hovering over any data point in the histogram reveals specific details, enhancing the ability to identify trends, outliers, and patterns within the data.

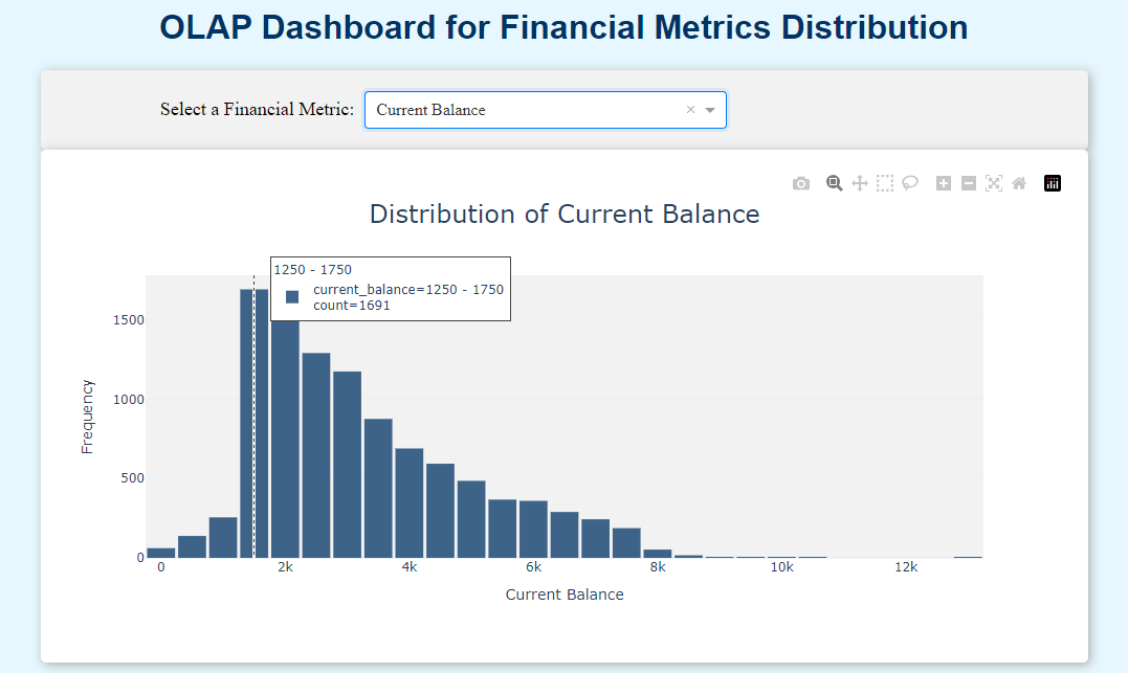

Figure 70 Screenshot of the OLAP dashboard displaying the histogram for the distribution of "Current Balance"

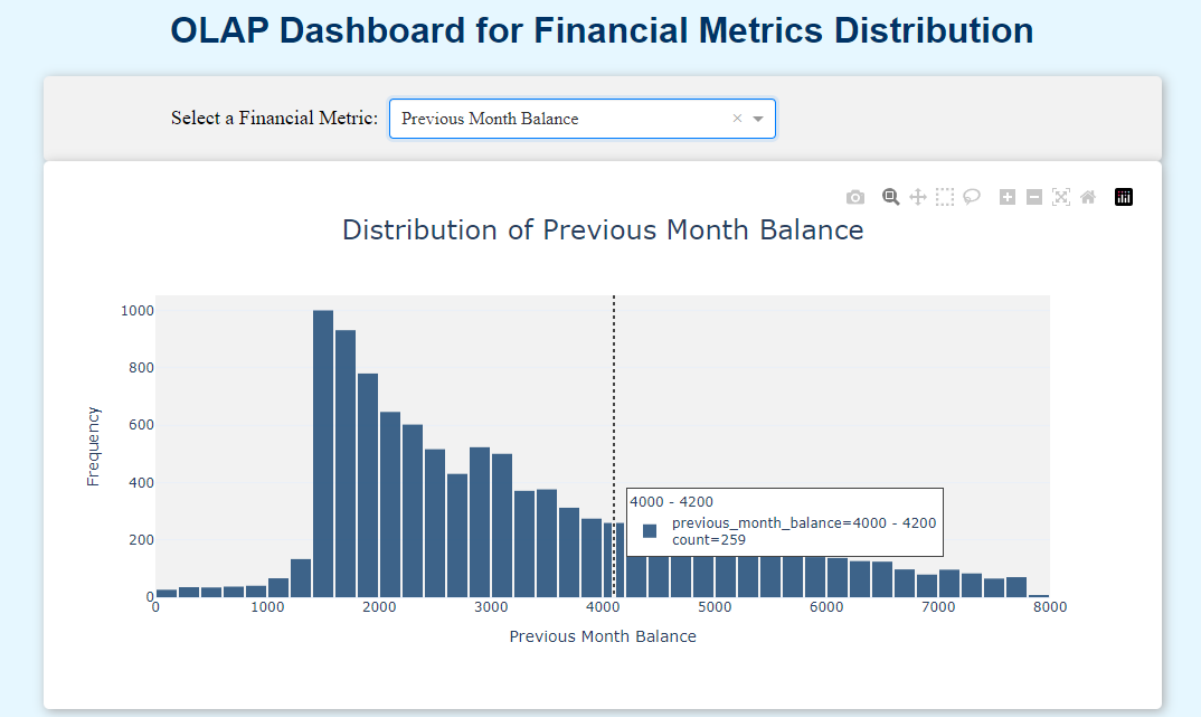

Figure 71 Screenshot of the OLAP dashboard showing the histogram for the distribution of "Previous Month Balance"

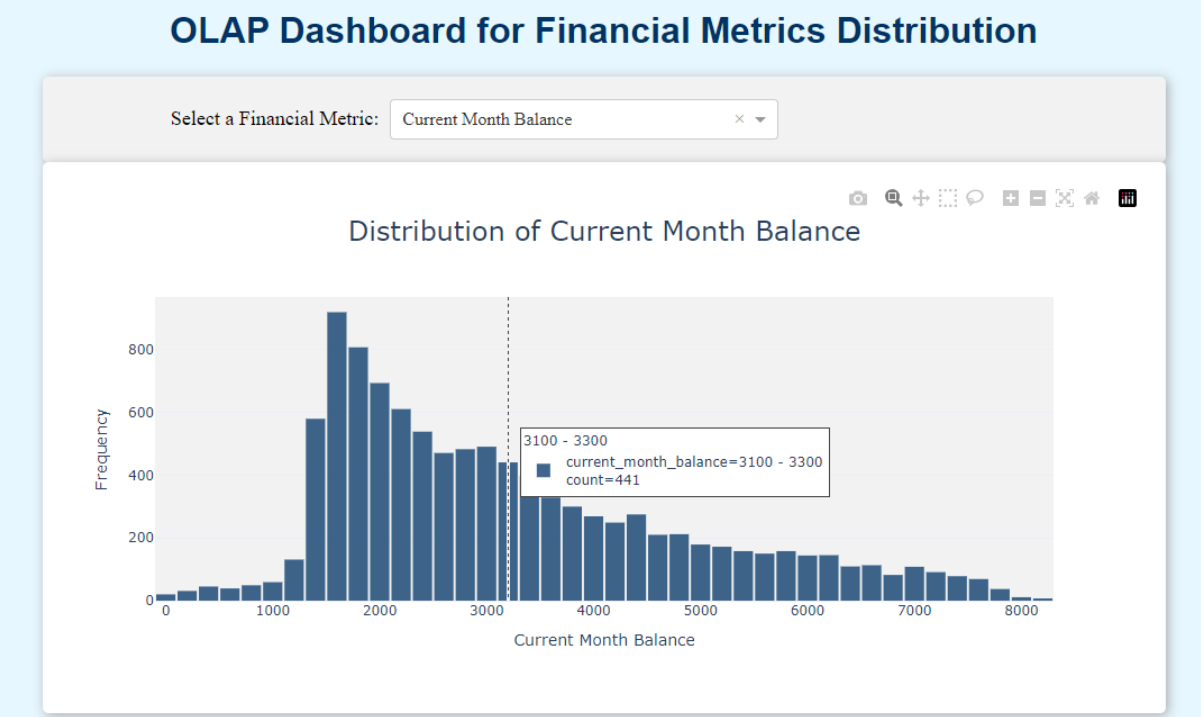

Figure 72 Screenshot of the OLAP dashboard illustrating the histogram for the distribution of "Current Month Balance"

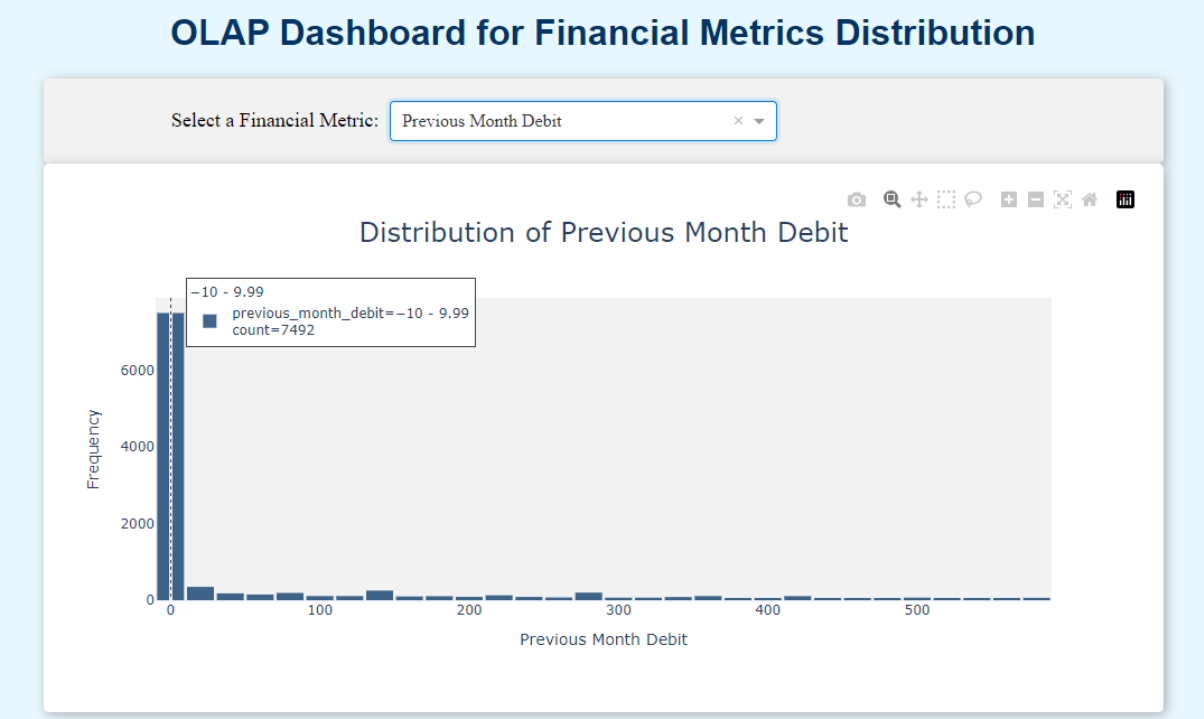

Figure 73 Screenshot of the OLAP dashboard presenting the histogram for the distribution of "Previous Month Debit"

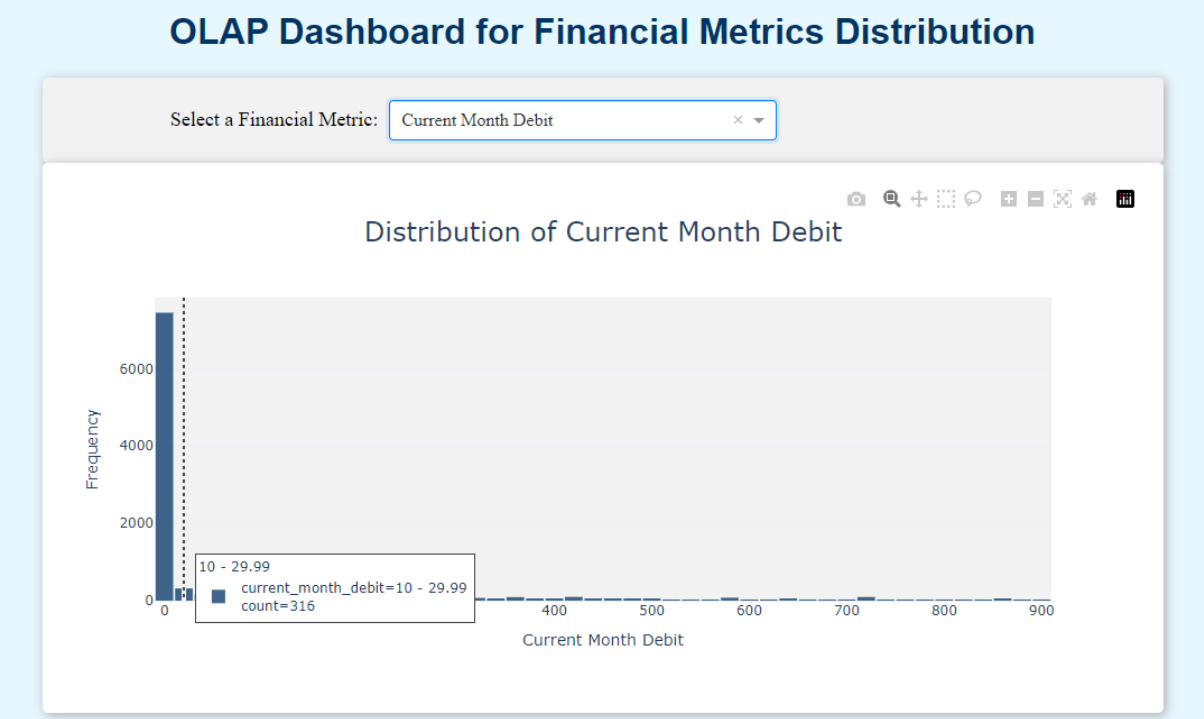

Figure 74 Screenshot of the OLAP dashboard displaying the histogram for the distribution of "Current Month Debit"

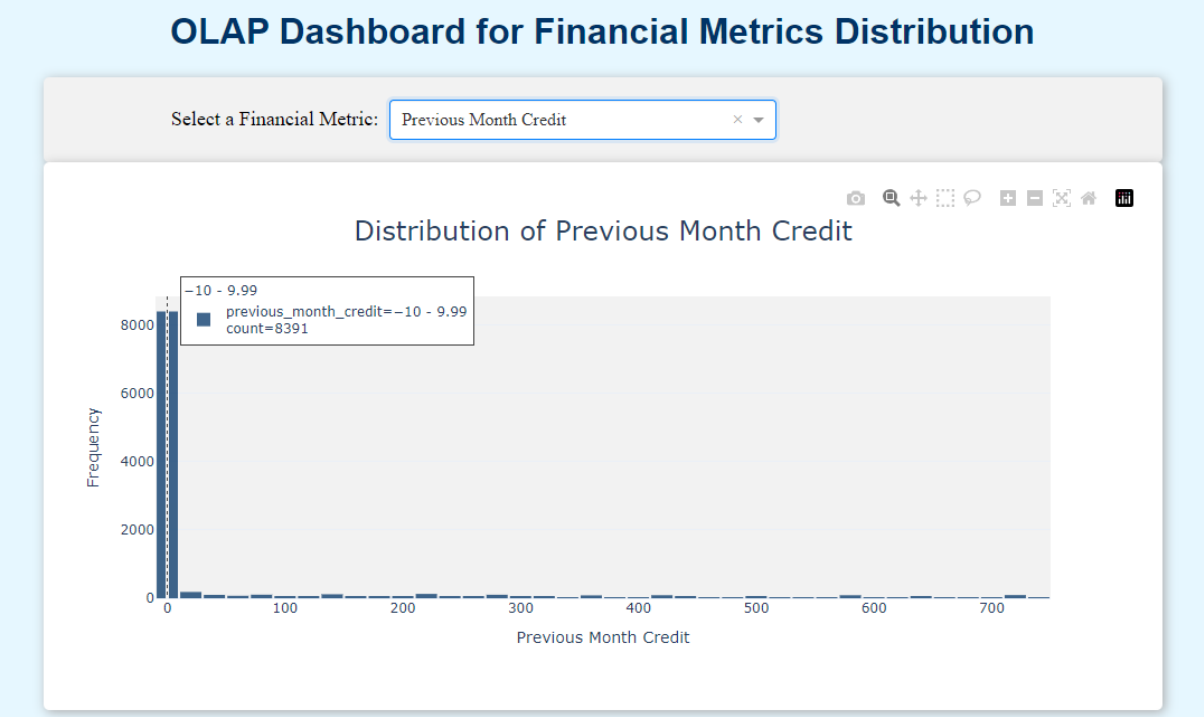

Figure 75 Screenshot of the OLAP dashboard showing the histogram for the distribution of "Previous Month Credit"

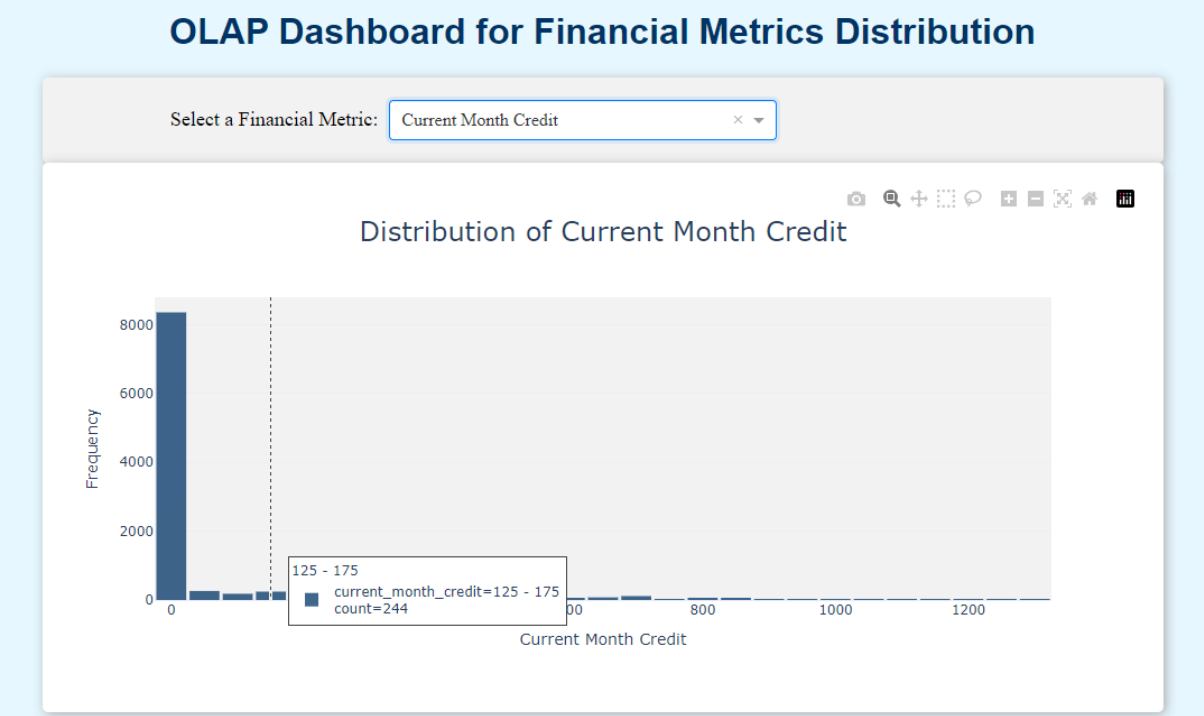

Figure 76 Screenshot of the OLAP dashboard illustrating the histogram for the distribution of "Current Month Credit"

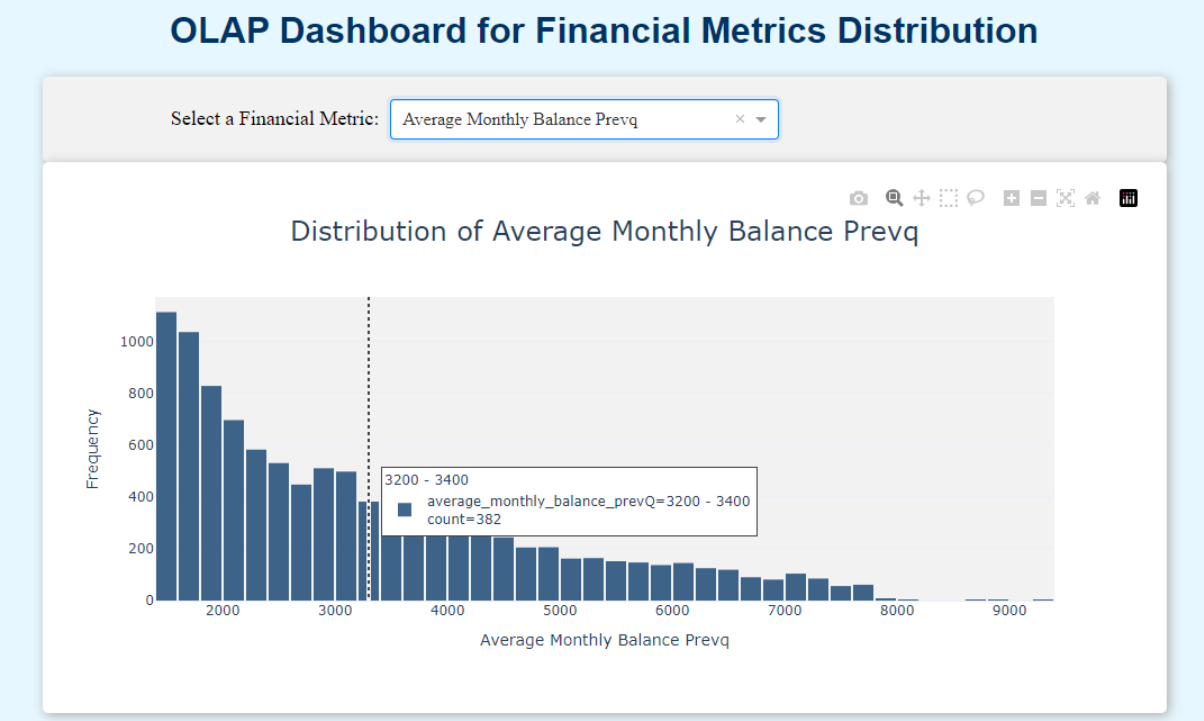

Figure 77 Screenshot of the OLAP dashboard presenting the histogram for the distribution of "Average Monthly Balance in the Previous Quarter (PrevQ)"

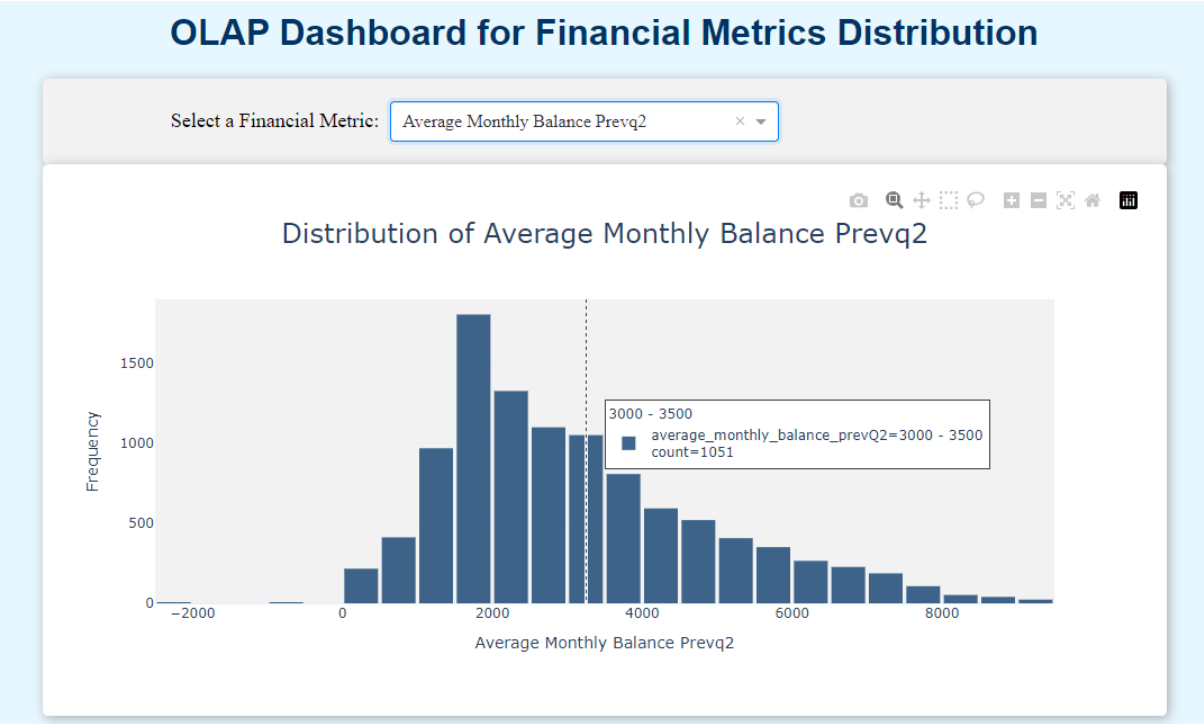

Figure 78 Screenshot of the OLAP dashboard displaying the histogram for the distribution of "Average Monthly Balance in the Previous Quarter 2 (PrevQ2)"

Overall Summary

This analysis focused on understanding the drivers behind customer churn in the financial services sector by employing a combination of data visualization, correlation analysis, multivariate regression, and an interactive OLAP dashboard to explore financial metrics that could predict churn and inform customer retention strategies.

The process began with a detailed exploration of the dataset, utilizing histograms and boxplots to examine the distribution, central tendencies, and variations in various financial metrics, such as account balances, debit and credit transactions, and average monthly balances over time. The visualizations indicated that most customers maintain moderate account balances and engage in smaller, regular transactions, while a few outliers represent higher-value customers. This initial exploration provided key insights into typical customer financial behaviour and highlighted significant outliers.

Subsequently, a correlation heatmap was generated to visualize the relationships between different financial variables. The heatmap revealed strong positive correlations among balance-related metrics, suggesting that customers tend to maintain consistent financial behaviour over time. Conversely, weaker correlations between transactions (such as debits and credits) and balances indicated that routine transactions do not significantly impact overall account balances. These findings guided the selection of variables for further analysis in the regression modeling.

To gain a deeper understanding of the factors influencing customer churn, a multivariate regression analysis was conducted. This method allowed for the simultaneous evaluation of multiple predictors, providing a comprehensive view of how different financial metrics interact to affect churn. The regression model included key variables such as account balances and transactional activities, with the results quantifying the relationship between each predictor and churn. The statistical significance of these relationships was highlighted by t-statistics and p-values, while the R-squared and Adjusted R-squared values provided insights into the model's explanatory power, affirming the importance of the selected variables in predicting customer churn.

Additionally, an interactive OLAP dashboard was developed using Python's Dash framework, allowing for dynamic exploration of the financial metrics. The dashboard featured histograms that visually represented the distribution of selected metrics, with a dropdown menu enabling easy switching between different metrics. Hovering over the data points in the

histograms revealed specific details, facilitating the identification of trends, outliers, and patterns in the data.

In summary, this analysis employed a structured approach to identify and model the financial behaviours that contribute to customer churn. By integrating data visualization, correlation analysis, multivariate regression, and an interactive dashboard, the study provided a comprehensive understanding of the key factors at play. The results suggest that while account balances are relatively stable and closely linked to churn, everyday transactions like debits and credits have less impact. These insights can inform targeted retention strategies, focusing on customers who are at higher risk of churn based on their financial behaviour.

5.1.2 Predictive Analysis – Random Forest Classifier

a. Importing Necessary Libraries

```
import pandas as pd
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.metrics import classification_report, confusion_matrix, accuracy_score
```

Figure 79 Importing key libraries and functions for Random Forest classification

The code snippet above imports necessary libraries and functions to build and evaluate a Random Forest classifier. It starts with `pandas`, a key tool for handling and analyzing data, which helps in working with datasets in a structured way. The `RandomForestClassifier` from `sklearn.ensemble` is included, providing a strong algorithm for classification tasks by using multiple decision trees to improve accuracy. The snippet also brings in `train_test_split` and `GridSearchCV` from `sklearn.model_selection`. `train_test_split` is used to split the data into training and testing sets, which is important for validating the model, while `GridSearchCV` is used for tuning hyperparameters by searching for the best model configuration. Finally, `classification_report`, `confusion_matrix`, and `accuracy_score` from `sklearn.metrics` are used to evaluate how well the model performs, offering detailed metrics like precision, recall, F1-score, and overall accuracy.

b. Model Building

```
# Initialize features and target variable
X = dataset[['current_balance', 'previous_month_balance', 'current_month_balance',
             'previous_month_debit', 'current_month_debit', 'previous_month_credit',
             'current_month_credit', 'average_monthly_balance_prevQ', 'average_monthly_balance_prevQ2']]

y = dataset['churn']
```

Figure 80 Initializing features and target variable for churn prediction

The code snippet above demonstrates the initialization of features (X) and the target variable (y) for a churn prediction model. The X variable is a DataFrame containing the selected features that are believed to be predictive of customer churn. These features include various

balance metrics and transaction amounts, such as current_balance, previous_month_balance, and average_monthly_balance_prevQ. These features capture the financial activities of customers over different periods. The target variable y is set to the churn column from the dataset, which indicates whether a customer has churned (typically a binary value, where 1 indicates churn and 0 indicates no churn). This separation of features and target variable is a crucial step in preparing the data for training a machine learning model.

```
# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

Figure 81 Splitting the dataset into training and testing sets

The code snippet above shows the process of splitting the dataset into training and testing sets using the train_test_split function from sklearn.model_selection. The features (X) and target variable (y) are divided into four subsets: X_train, X_test, y_train, and y_test. The test_size=0.2 parameter indicates that 20% of the data will be allocated to the testing set, while the remaining 80% will be used for training the model. The random_state=42 ensures that the split is reproducible, meaning the same split will occur each time the code is run. This step is critical for validating the model's performance, as it allows for testing the model on data it hasn't seen during training.

```
# Initialize the Random Forest model
model = RandomForestClassifier(random_state=42)
```

Figure 82 Initializing the Random Forest model

The code snippet above shows the initialization of the Random Forest model using `RandomForestClassifier` from the `sklearn.ensemble` module. The `random_state=42` parameter is set to ensure that the results are reproducible, meaning that the model will produce the same output if the code is run multiple times. This initialization step is crucial as it prepares the Random Forest classifier for subsequent training on the dataset, allowing the model to learn patterns in the data by constructing multiple decision trees to enhance predictive accuracy.

d. Hyperparameter Tuning

```
# Define the hyperparameters grid
param_grid = {
    'n_estimators': [100, 200, 300],
    'max_depth': [10, 20, 30, None],
    'min_samples_split': [2, 5, 10],
    'min_samples_leaf': [1, 2, 4],
    'bootstrap': [True, False]
}

# Initialize GridSearchCV
grid_search = GridSearchCV(estimator=model, param_grid=param_grid,
                           cv=5, n_jobs=-1, verbose=2, scoring='accuracy')
```

Figure 83 Hyperparameter tuning using GridSearchCV

The code snippet above sets up hyperparameter tuning for the Random Forest model using `GridSearchCV` from the `sklearn.model_selection` module. The `param_grid` dictionary defines a grid of hyperparameters to be tested, including the number of trees (`n_estimators`), the maximum depth of the trees (`max_depth`), the minimum number of samples required to split an internal node (`min_samples_split`), the minimum number of samples required to be at a leaf node (`min_samples_leaf`), and whether bootstrap samples are used when building trees (`bootstrap`). The `GridSearchCV` is then initialized with the model, parameter grid, and additional settings such as cross-validation (`cv=5`), parallel processing (`n_jobs=-1`), and verbose output (`verbose=2`). The `scoring='accuracy'` argument indicates that the model's performance will be evaluated based on accuracy during the grid search, which helps in selecting the optimal combination of hyperparameters.

```
# Perform Grid Search
grid_search.fit(X_train, y_train)

# Best parameters from grid search
best_params = grid_search.best_params_
print("Best parameters found: ", best_params)

# Use the best parameters to fit the model
best_model = grid_search.best_estimator_

# Make predictions on the test data
y_pred = best_model.predict(X_test)
```

Figure 84 Executing the grid search and using the best model

The code snippet above performs the grid search and applies the best model to make predictions. The `grid_search.fit(X_train, y_train)` line starts the grid search process on the training data, where various combinations of hyperparameters are tested to find the best performing model. Once the grid search is complete, the best hyperparameters are extracted using `grid_search.best_params_`, which are then printed out. The best model, determined by the grid search, is retrieved with `grid_search.best_estimator_`. Finally, the model with the optimal hyperparameters is used to make predictions on the test data (`X_test`) using `best_model.predict(X_test)`, storing the predicted values in `y_pred`.

```
Best parameters found:  {'bootstrap': True, 'max_depth': 10, 'min_samples_leaf': 4, 'min_samples_split': 10, 'n_estimators': 100}
```

Figure 85 Best hyperparameters identified through grid search

```
# Evaluate the model
print(classification_report(y_test, y_pred))
print(f"Accuracy: {accuracy_score(y_test, y_pred):.2f}")
```

Figure 86 Evaluating the model's performance using a classification report and accuracy score

The code snippet above evaluates the performance of the trained Random Forest model. The classification_report function provides detailed metrics, including precision, recall, F1-score, and support for each class in the target variable, offering insights into how well the model is performing across different categories. The accuracy_score function calculates the overall accuracy of the model, representing the proportion of correct predictions out of the total number of predictions, and this score is formatted to two decimal places for clarity.

```
Accuracy: 0.94
```

Figure 87 Overall Model Accuracy on the Test Dataset: 94%

	precision	recall	f1-score	support
0	0.94	0.99	0.97	1938
1	0.67	0.24	0.35	155
accuracy			0.94	2093
macro avg	0.81	0.61	0.66	2093
weighted avg	0.92	0.94	0.92	2093

Figure 88 Classification Report Detailing Model Performance Across Target Classes

The classification report above provides a detailed breakdown of the Random Forest model's performance on the test dataset. The report is divided into two primary classes: class 0 (representing "Not Churn") and class 1 (representing "Churn"). For each class, the report includes key metrics such as precision, recall, F1-score, and support.

Precision measures the accuracy of the positive predictions, with the model achieving a precision of 0.94 for class 0 and 0.67 for class 1. This indicates that when the model predicts "Not Churn," it is correct 94% of the time, while it is correct 67% of the time for "Churn" predictions. Recall, or the true positive rate, is particularly high for class 0 at 0.99, meaning the model successfully identifies 99% of actual "Not Churn" cases. However, the recall for class 1 is significantly lower at 0.24, indicating that the model misses many actual "Churn" cases.

The F1-score, which balances precision and recall, reflects the overall performance for each class, with the model scoring 0.97 for class 0 and 0.35 for class 1. These values suggest that the model performs exceptionally well in predicting "Not Churn" but struggles with accurately predicting "Churn."

The overall accuracy of the model stands at 0.94, indicating that 94% of all predictions are correct. The macro average provides an unweighted average of the F1-scores for both classes, with a value of 0.66, highlighting the disparity in performance between the two classes. The weighted average, which accounts for the support (number of instances) of each class, shows an overall F1-score of 0.92, demonstrating the model's strong performance in handling the majority class (0) but less effectiveness with the minority class (1).

This report reveals that while the model is highly effective in predicting customers who will not churn, it needs improvement in accurately identifying those who are at risk of churning. This imbalance suggests that further tuning or alternative modeling approaches may be necessary to improve the model's sensitivity to the "Churn" class, thereby enhancing its overall predictive power.

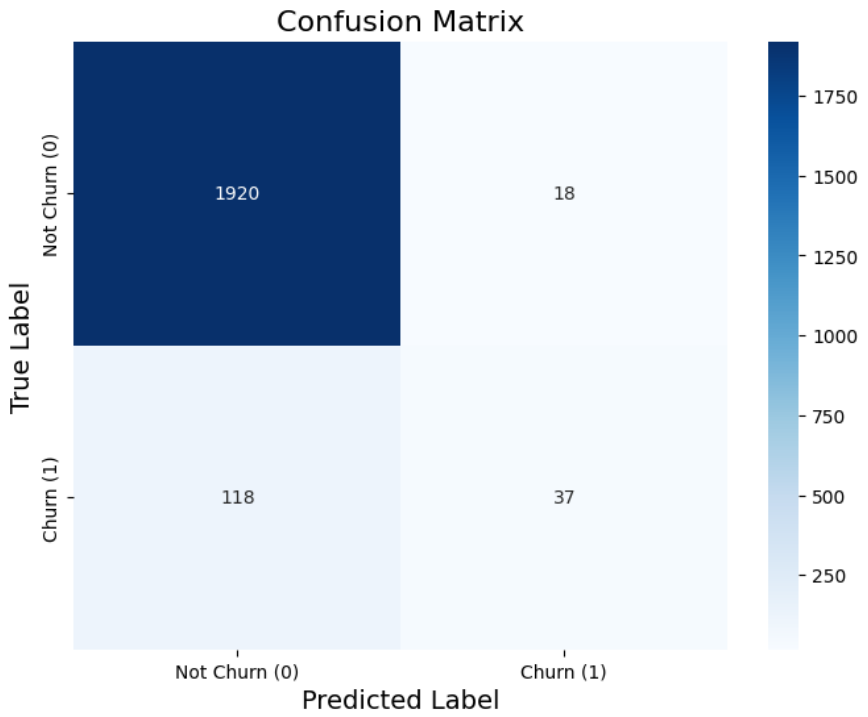

Figure 89 Confusion Matrix Illustrating the Random Forest Classifier's Performance in Predicting Customer Churn

The confusion matrix shown in the figure above provides a detailed breakdown of the Random Forest classifier's performance in predicting whether customers will churn or remain with the service. Each element within the matrix offers valuable insights into the model's predictive accuracy and areas that may require further refinement.

The top-left cell of the matrix, which contains **1920 True Negatives (TN)**, represents the number of customers correctly identified as not churning. This high number indicates that the model is particularly effective at recognizing customers who are likely to stay, which suggests that it has a strong capacity to maintain customer loyalty by accurately identifying those not at risk of leaving.

On the top-right, we see the **18 False Positives (FP)**, which are instances where the model incorrectly predicted that customers would churn when, in fact, they did not. The low number of false positives is encouraging, as it indicates the model's ability to avoid incorrectly categorizing loyal customers as potential churners. Minimizing false positives is crucial because such errors could lead to unnecessary interventions and possibly disrupt customer satisfaction.

The bottom-left cell, showing **118 False Negatives (FN)**, is more concerning. These are the customers who churned but were not identified as churn risks by the model. False negatives are particularly problematic in churn prediction because failing to identify at-risk customers means missing opportunities for proactive retention strategies. This suggests that while the model is strong overall, it may need improvement in sensitivity, particularly in identifying subtle signs that a customer may leave.

Finally, the bottom-right cell presents the **37 True Positives (TP)**, which are the cases where the model correctly identified that the customers would churn. This figure is relatively low compared to the total number of customers who churned (155), indicating that while the model can detect some churn cases, it struggles to capture the full extent of at-risk customers.

The overall accuracy of the model, as indicated by the confusion matrix, is high, but this alone does not capture the full complexity of the model's performance. While the model excels at predicting customers who will not churn (high TN), its ability to correctly identify those who will churn (TP) is limited. This imbalance highlights a trade-off between precision (avoiding false positives) and recall (minimizing false negatives). For a business focused on retaining customers, improving recall by better identifying potential churners would be a crucial next step. This could involve refining the model or incorporating additional features that more accurately capture the early indicators of customer dissatisfaction or disengagement.

In summary, the confusion matrix offers a comprehensive view of the Random Forest classifier's effectiveness in churn prediction. It underscores the importance of balancing both precision and recall ensuring that at-risk customers are correctly identified and appropriately targeted for retention efforts, thus enhancing the overall strategy to minimize churn and maximize customer retention.

5.2 Evaluate the influence of customer net worth and geographic factors on churn

The method of evaluation in this section focuses on understanding how customer net worth and geographic factors influence churn rates. This analysis is conducted using a Logistic Regression model, a widely used and effective technique for binary classification problems like churn prediction. The model leverages key data such as customer net worth categories, branch codes, city locations, and various occupational statuses to evaluate the likelihood of customer churn. By analysing these factors, the Logistic Regression model helps to identify which segments of the customer base are at a higher risk of churn, enabling the bank to implement targeted retention strategies that address the specific characteristics of customers across different geographic and financial profiles.

5.2.1 Descriptive Analysis

a. Data Distribution

The histograms presented below offer a detailed exploration of the distribution patterns for customer net worth and geographic factors, which are key in understanding their impact on churn. These visualizations cover important variables such as customer net worth categories, branch locations, city demographics, and occupational statuses. By examining these distributions, trends and outliers can be identified, providing insights into the factors that may influence customer decisions to stay or leave. This analysis is a crucial part of understanding customer behaviour and will guide the development of effective retention strategies.

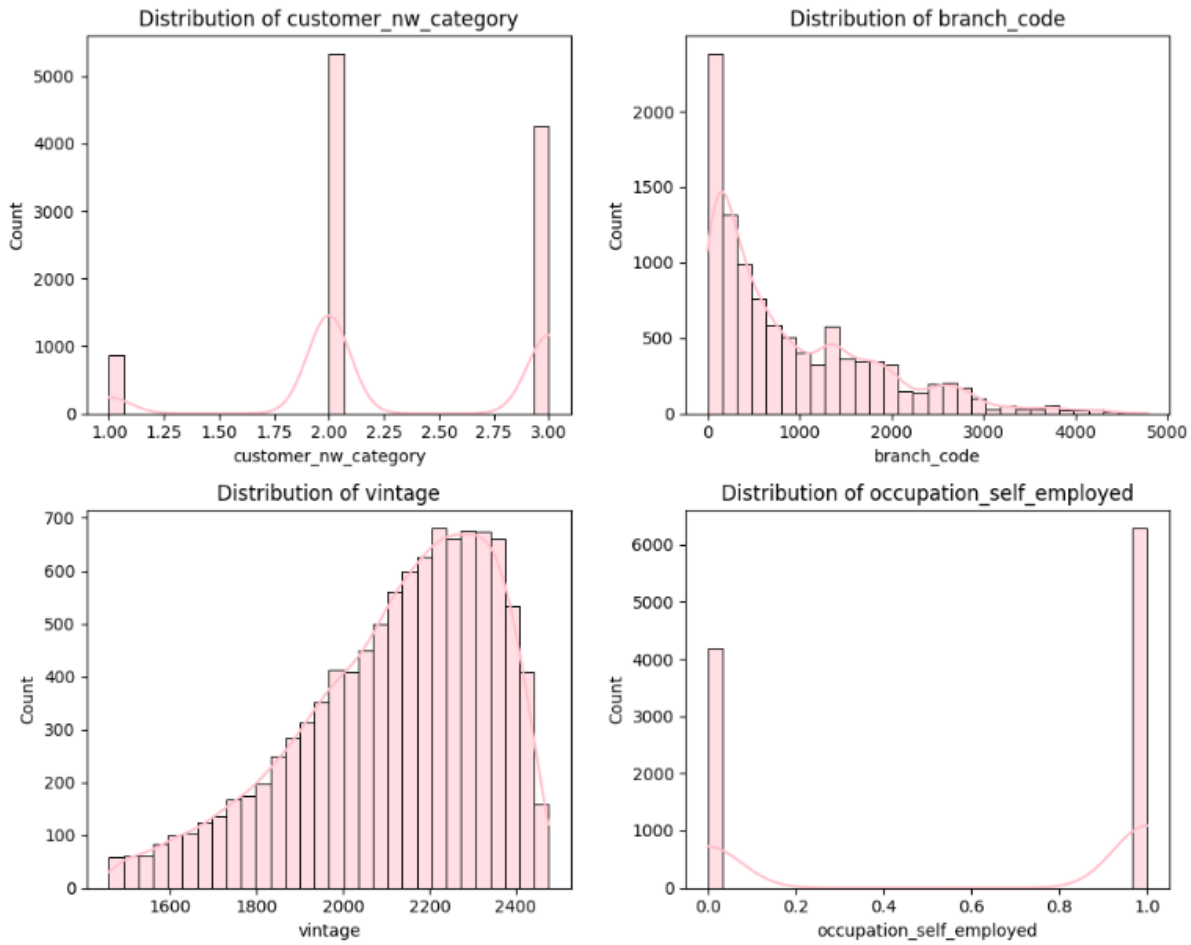

Figure 90 Combined histogram displaying distributions of customer net worth, branch codes, vintage, and self-employment status

Distribution of customer_nw_category:

- The histogram reveals distinct peaks at specific net worth categories.
- Most customers fall into categories 2.0 and 3.0, indicating higher net worth groups are more prevalent.
- There are smaller groups in the lower net worth categories, which might reflect different financial strategies or customer segments within the bank.

Distribution of branch_code:

- The distribution is heavily skewed towards lower branch codes, suggesting a concentration of customers in a few branches.
- As the branch code increases, the number of customers decreases, indicating fewer customers in higher-numbered branches.
- This pattern could reflect regional preferences or the relative sizes of branches.

Distribution of vintage:

- The histogram exhibits a normal distribution with a peak between 2000 and 2200 days, indicating that most customers have been with the bank for about 5 to 6 years.
- The left tail suggests fewer customers with shorter associations, while the right tail indicates a drop-off in long-term customers.
- This information can be crucial for understanding customer loyalty and tenure with the bank.

Distribution of occupation_self_employed:

- The histogram shows a clear binary distribution, where customers are either fully self-employed (value 1) or not self-employed (value 0).
- A significant number of customers are not self-employed, with a smaller yet substantial group being self-employed.
- This binary distribution provides insight into the employment status of the bank's customer base, which can influence their financial behaviours and needs.

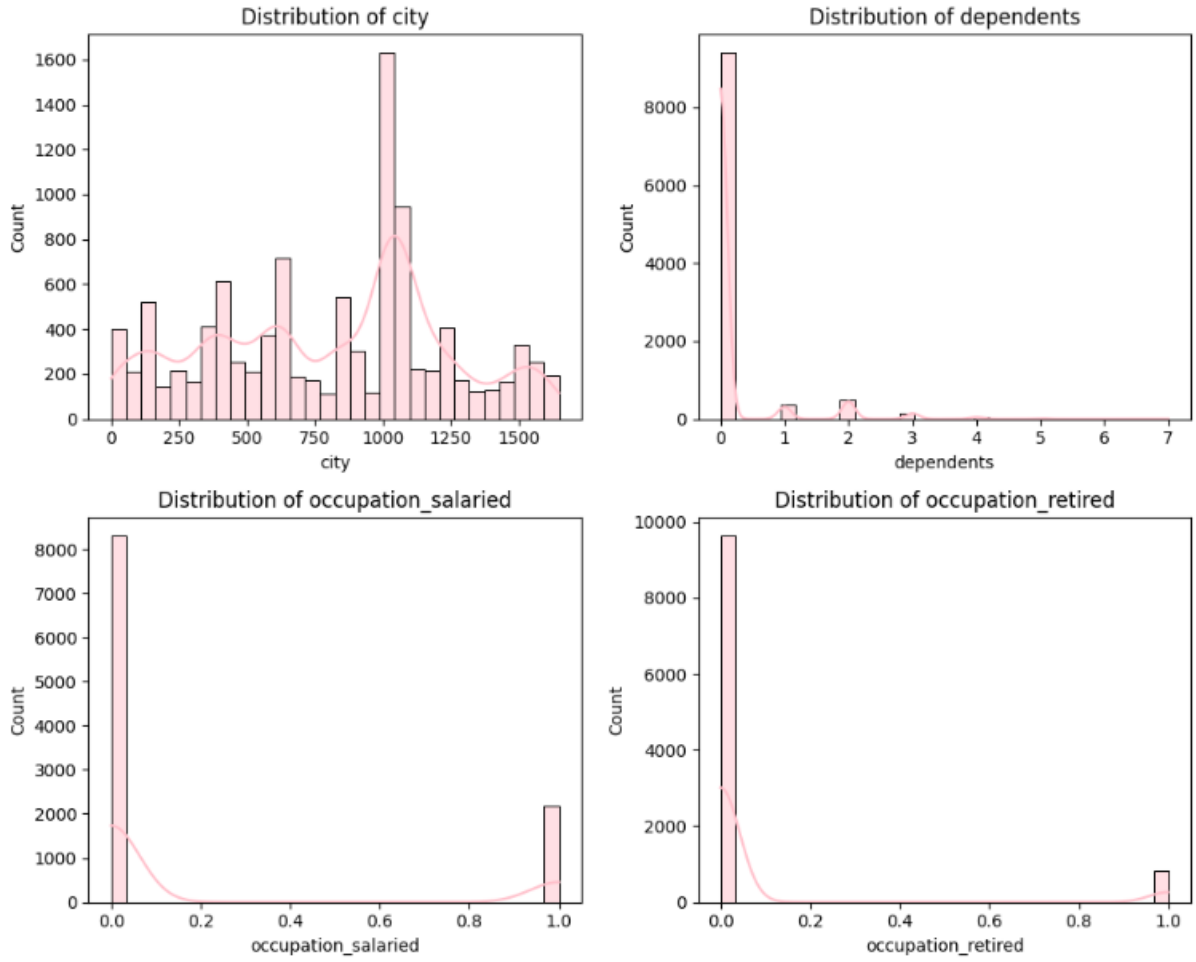

Figure 91 Combined histogram displaying distributions of city, number of dependents, salaried occupation, and retired occupation among customers

Distribution of city:

- The customer base is widely distributed across different city codes.
- There is a noticeable peak around certain city codes, suggesting regional clusters of customers.
- Some cities have significantly higher customer counts, indicating potential regional market strengths.

Distribution of dependents:

- Most customers have zero or one dependent, with a steep drop-off for higher numbers of dependents.
- This suggests that the typical customer profile is either single or has a small family size.
- Very few customers have more than three dependents, showing a strong skew in the data.

Distribution of occupation_salaried:

- A large portion of the dataset is concentrated on customers who are either fully salaried or not salaried at all.
- This distribution highlights two distinct segments: those in salaried employment and those who are not.
- There is a significant number of customers in the salaried category, indicating a stable income source for a substantial part of the customer base.

Distribution of occupation_retired:

- Many customers are not retired, with only a small portion indicating retirement status.
- This suggests that the customer base is predominantly in the working-age group.
- The small number of retired customers could point to either a younger customer demographic or less engagement from older, retired individuals.

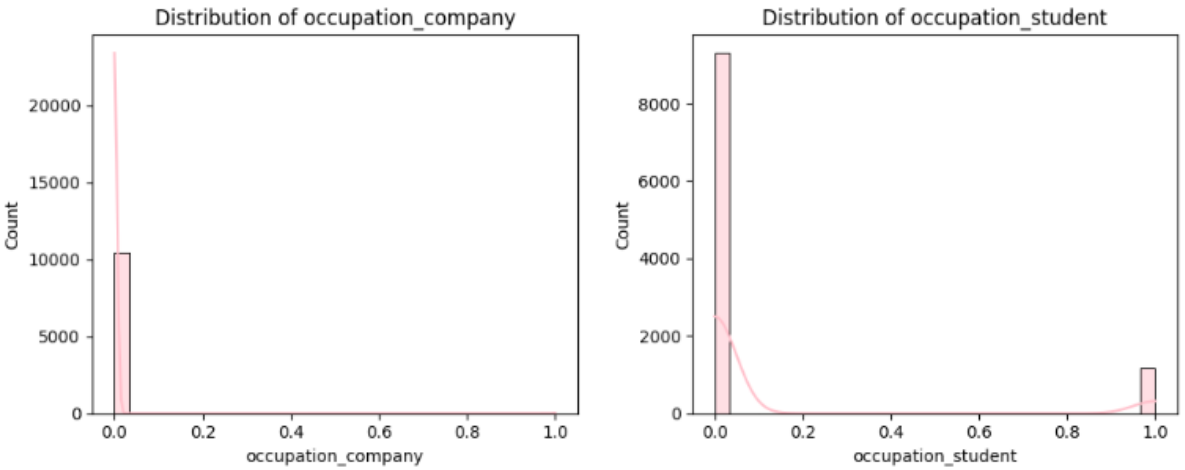

Figure 92 Combined histogram displaying distributions of company occupation and student occupation among customers

Distribution of occupation_company:

- Most customers are not associated with a company occupation, indicating a limited engagement from corporate employees.
- A small percentage of the dataset shows customers linked to company occupations, which could represent employees of specific firms or industries.
- This distribution highlights a potential area for targeted marketing strategies aimed at increasing corporate employee engagement.

Distribution of occupation_student:

- Most customers are not students, with a very small portion of the dataset indicating student status.
- The presence of students, although small, suggests a niche segment within the customer base that could benefit from specialized financial products.
- The sharp drop after the initial peak indicates that students make up a minor fraction of the customer base, potentially reflecting their lower financial engagement or earning capacity.

b. Potential Outliers

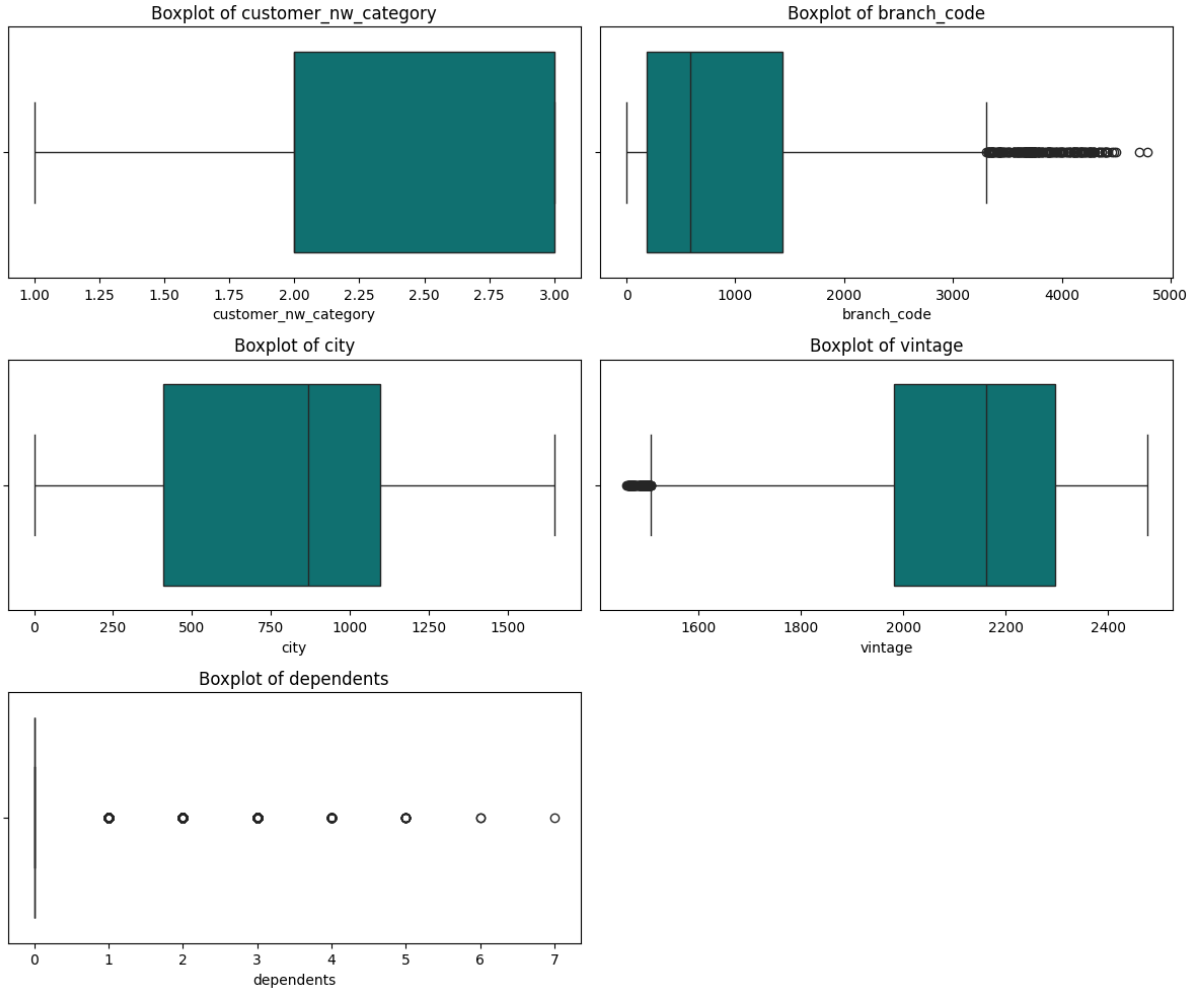

Figure 93 Boxplots Highlighting Potential Outliers in Customer Net Worth and Geographic Factors

The boxplots give a clear picture of the distribution and potential outliers within key variables related to customer net worth and geographic factors. These visualizations are helpful in spotting patterns, identifying anomalies, and understanding how data is spread within the financial context.

For the customer_nw_category (customer net worth category), the distribution looks quite even across the different categories, with no significant outliers. This suggests that most customers fall within the expected range of net worth categories, indicating a balanced spread

of wealth among the bank's clients. This balance could be important for the bank in terms of offering financial products and services that cater to a wide range of customers.

The branch_code boxplot shows some outliers, particularly on the higher end. These outliers might be linked to newer or specialized branches with higher codes. The variation here could reflect differences in geographic or administrative factors, showing that some branches cater to areas with unique demographic or economic characteristics. Understanding these outliers could provide insights into how different regions perform or highlight the need for specialized services.

The vintage boxplot, which measures how long customers have been with the bank, shows a smooth distribution with a few outliers on the lower end. This indicates that while most customers have had a moderate to long relationship with the bank, there are some newer clients. These new customers might require different engagement strategies since their shorter tenure could mean they're at a higher risk of leaving.

The city boxplot displays a wide range of values, reflecting the diverse geographic spread of customers. There aren't any extreme outliers here, suggesting that the bank's customers are spread across various cities, with most concentrated in certain urban areas. This spread might point to the bank's strategic placement of branches and its market reach across different regions.

Finally, the dependents boxplot shows that most customers have zero to one dependent, with some notable outliers who have as many as seven. These outliers could represent customers with unique financial needs, such as those supporting larger families, which could influence their banking behaviour and needs.

Together, these boxplots offer valuable insights into the customer base, revealing both common patterns and unique cases. By understanding the distribution and recognizing outliers, the bank can tailor its strategies to better meet the diverse needs of its customers, ensuring more effective engagement and retention efforts.

c. Correlation Heatmap of Customer Net Worth, Geographic, and Demographic Factors

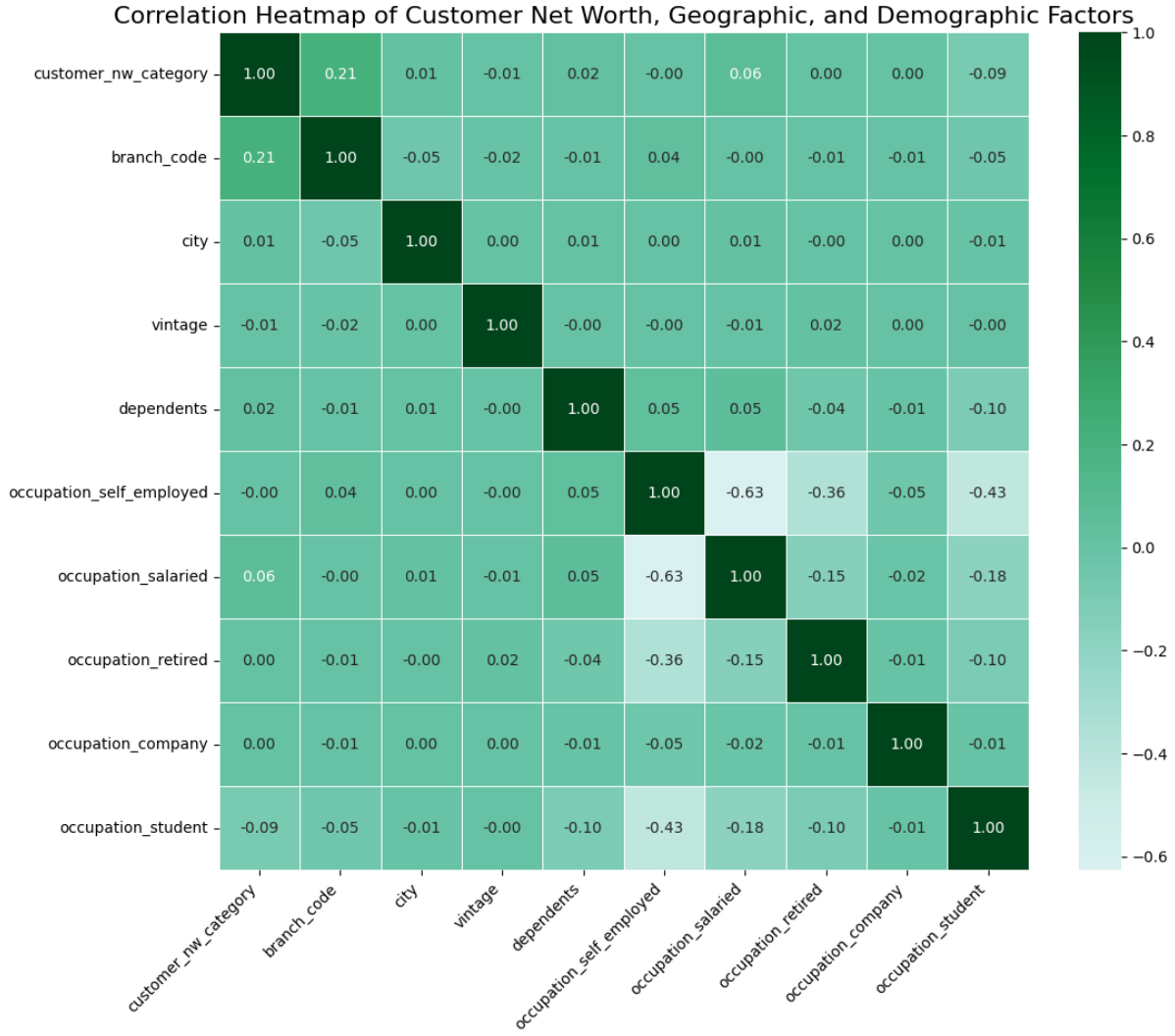

Figure 94 Correlation Heatmap of Customer Net Worth, Geographic, and Demographic Factors

The correlation heatmap above provides a visual representation of the relationships between various customer net worth, geographic, and demographic factors. The strength and direction of these correlations are indicated by the colour intensity, with values ranging from -1 (strong negative correlation) to 1 (strong positive correlation).

Notably, the heatmap reveals that there is a moderate positive correlation between the customer net worth category (customer_nw_category) and the branch code (branch_code), suggesting that certain branches may cater to customers with similar net worth levels. This

insight could be crucial for understanding regional financial strategies or branch-specific customer profiles.

Another significant observation is the strong negative correlation between salaried occupations (occupation_salaried) and self-employed occupations (occupation_self_employed). This indicates that customers who are salaried are less likely to be self-employed, which aligns with typical employment patterns. Similarly, there is a moderate negative correlation between being salaried and being retired (occupation_retired), reflecting the transition from active employment to retirement.

The vintage column, which represents the length of time a customer has been with the bank, shows very little correlation with other variables, suggesting that customer tenure does not heavily influence or is not influenced by net worth, geography, or occupation. This could imply that customer loyalty is relatively stable across different demographic groups.

Overall, the heatmap underscores the nuanced relationships between customer characteristics, providing valuable insights into how these factors interact. This analysis can inform more targeted customer segmentation, marketing strategies, and retention efforts, particularly by highlighting which customer demographics are more closely linked and may exhibit similar behaviours or financial needs.

d. Evaluating the Impact of Customer Net Worth and Demographic Factors on Churn: Multivariate Regression Analysis

OLS Regression Results

Dep. Variable:	churn	R-squared:	0.001
Model:	OLS	Adj. R-squared:	0.000
Method:	Least Squares	F-statistic:	1.308
Date:	Mon, 19 Aug 2024	Prob (F-statistic):	0.226
Time:	01:25:49	Log-Likelihood:	-852.58
No. Observations:	10463	AIC:	1725.
Df Residuals:	10453	BIC:	1798.
Df Model:	9		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	0.0562	0.025	2.239	0.025	0.007	0.105
customer_nw_category	-0.0039	0.004	-0.915	0.360	-0.012	0.004
branch_code	3.07e-06	2.9e-06	1.057	0.290	-2.62e-06	8.76e-06
city	5.343e-06	5.96e-06	0.896	0.370	-6.35e-06	1.7e-05
vintage	1.087e-06	1.15e-05	0.095	0.925	-2.14e-05	2.36e-05
dependents	0.0010	0.004	0.246	0.805	-0.007	0.009
occupation_self_employed	0.0220	0.013	1.742	0.081	-0.003	0.047
occupation_salaried	0.0201	0.013	1.521	0.128	-0.006	0.046
occupation_retired	0.0009	0.014	0.066	0.948	-0.027	0.029
occupation_company	0.0106	0.057	0.186	0.852	-0.101	0.122
occupation_student	0.0025	0.014	0.181	0.857	-0.025	0.030

Figure 95 OLS Regression Results: Evaluating the Impact of Customer Net Worth and Demographic Factors on Churn

The multivariate regression results above provide a comprehensive view of how different customer characteristics collectively relate to the likelihood of churn. In this analysis, several key metrics, including coefficients, t-statistics, p-values, and the R-squared value, provide insight into the model's effectiveness in explaining churn behaviour.

Coefficients in this context represent the estimated impact of each customer characteristic on churn, holding all other factors constant. For instance, the coefficient for customer_nw_category is -0.0039, suggesting that as a customer's net worth category increases, the likelihood of churn decreases slightly. However, the coefficients for all variables are relatively small, indicating that changes in these characteristics lead to only minimal shifts in churn probability.

T-statistics measure the strength of the relationship between each independent variable and churn, considering the variability in the data. In this model, the t-statistics are low across the board, implying that these customer characteristics do not have a strong statistical impact on churn. This is consistent with the p-values, which indicate the probability that the observed relationships could have occurred by chance. With p-values above the conventional significance threshold of 0.05 for all variables, there is no strong evidence to suggest that any of these characteristics are significantly associated with churn.

The R-squared value of 0.001 indicates that the model explains only 0.1% of the variation in churn, which is extremely low. This suggests that the selected variables in the model do not capture the key factors influencing customer churn. The Adjusted R-squared, which accounts for the number of predictors in the model, also remains low, reinforcing the conclusion that the model has limited explanatory power.

Overall, the multivariate regression analysis reveals that the chosen customer characteristics such as net worth category, branch code, city, and occupation—have little to no significant impact on predicting churn. The small coefficients, low t-statistics, insignificant p-values, and very low R-squared value all indicate that these factors, as modeled, are not strong predictors of customer churn. This outcome suggests that other variables, perhaps more directly related to customer behaviour or external factors, need to be explored to develop a more robust and predictive model for churn.

e. OLAP Dashboard for Customer Net Worth and Geographic Factors Distribution

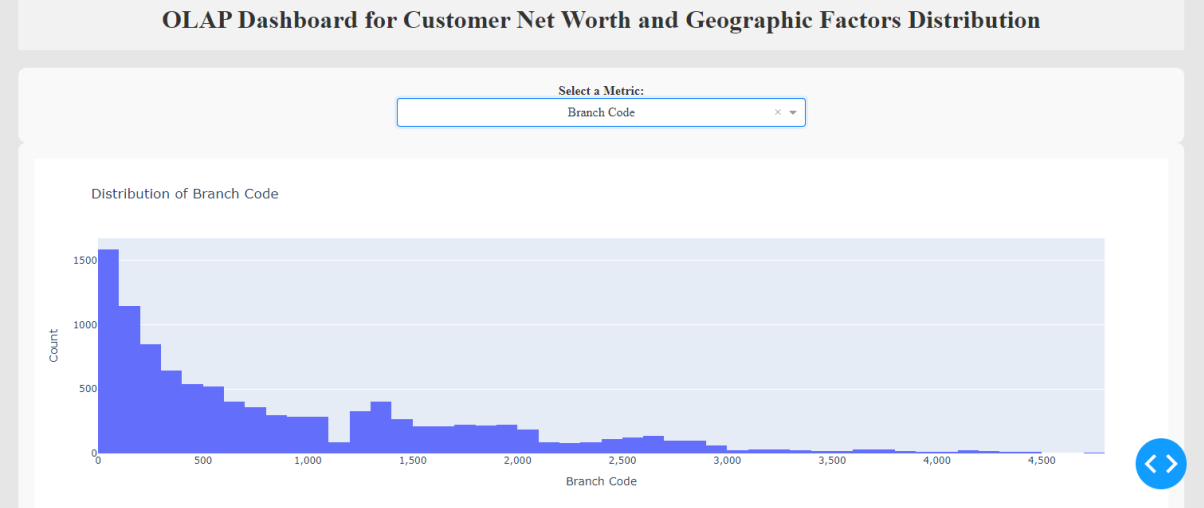

Figure 96 Interactive OLAP Dashboard showcasing the distribution of the 'Branch Code' metric, part of the Customer Net Worth and Geographic Factors analysis

The OLAP Dashboard is created to display the data distribution of customer net worth and various geographic factors in an interactive and visually engaging format. This dashboard allows for an in-depth exploration of customer-related metrics, such as net worth and geographic location, through dynamically generated histograms. The dashboard features a dropdown menu to select specific columns for analysis, which triggers the creation of corresponding visualizations. These features offer valuable insights into customer behaviour patterns, identifying trends, outliers, and potential relationships between different factors. The integration of Dash and Plotly ensures that the dashboard is both interactive and visually engaging, making it an effective tool for data analysis and decision-making.

5.2.2 Predictive Analysis – Logistic Regression

a. Import Library

```
import pandas as pd
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report, accuracy_score
```

Figure 97 Import libraries

The code above begins by importing essential libraries for data manipulation, model training, and evaluation. pandas are used for handling and processing data. The train_test_split function from sklearn.model_selection is used to split the dataset into training and testing subsets, while GridSearchCV is employed to optimize hyperparameters through cross-validation. The LogisticRegression class from sklearn.linear_model is the model used for predicting customer churn, a binary classification task. Finally, classification_report and accuracy_score from sklearn.metrics are used to evaluate the model's performance by providing detailed insights into the accuracy, precision, recall, and other key metrics. This setup is integral to assessing how customer net worth and geographic factors impact churn in a banking context.

b. Model building

```
# Define the features and target variable
X = dataset[['customer_nw_category', 'branch_code', 'city',
             'vintage', 'dependents', 'occupation_self_employed',
             'occupation_salaried', 'occupation_retired',
             'occupation_company', 'occupation_student']]

y = dataset['churn']
```

Figure 98 Features and target variable

The code snippet above begins by defining the feature variables X, which are used to predict customer churn. The X DataFrame is created by selecting specific columns from the dataset. These columns include customer_nw_category, branch_code, city, vintage, dependents, and various occupation categories such as occupation_self_employed, occupation_salaried, occupation_retired, occupation_company, and occupation_student. Each of these features represents a different aspect of customer information. For instance, customer_nw_category categorizes customers based on their net worth, while branch_code and city capture geographic details. The vintage column indicates the duration of the customer's relationship with the bank, and dependents provides information on family size. The occupation-related features classify the customers based on their employment status, contributing to the model's understanding of how different job types might influence churn.

The target variable y is defined as dataset['churn'], which represents the outcome the model aims to predict. This variable indicates whether a customer has churned or not, with binary values where typically 0 denotes a customer who has not churned and 1 denotes a customer who has churned. By defining y in this way, the code sets up the dataset for supervised learning, where the model will learn to associate patterns in the features X with the likelihood of churn as specified in y. The target variable is essential for training and evaluating the performance of the predictive model, as it provides the actual values against which the model's predictions will be compared.

```
# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

Figure 99 Data splitting

The code above splits the dataset into training and testing sets using the train_test_split function. Specifically, 80% of the data (X_train, y_train) is used for training the model, while the remaining 20% (X_test, y_test) is reserved for testing its performance. The random_state=42 ensures that the split is consistent and reproducible every time the code is run. This approach helps in evaluating the model's ability to generalize to unseen data by testing it on the separate test set.

```
# Define the Logistic Regression model
log_reg = LogisticRegression(max_iter=1000)
```

Figure 100 Logistic Regression Model

This line of code defines a Logistic Regression model, which is a type of statistical model used for binary classification tasks, such as predicting whether a customer will churn or not. The model is instantiated using the LogisticRegression class from scikit-learn. The max_iter=1000 parameter specifies that the algorithm can perform up to 1,000 iterations to converge on an optimal solution. This higher iteration limit is set to ensure the model has enough opportunity to find the best fit, especially if the dataset is complex or the default number of iterations (usually 100) is insufficient for convergence.

e. Hyperparameter tuning

```
    # Set up the hyperparameter grid for tuning
    param_grid = {
        'C': [0.01, 0.1, 1, 10, 100],  # Regularization strength
        'penalty': ['l1', 'l2', 'elasticnet', 'none'],  # Penalty types
        'solver': ['lbfgs', 'liblinear', 'saga']  # Solvers
    }

    # Initialize GridSearchCV
    grid_search = GridSearchCV(estimator=log_reg, param_grid=param_grid,
                               cv=5, n_jobs=-1, verbose=2, scoring='accuracy')

    # Perform Grid Search to find the best hyperparameters
    grid_search.fit(X_train, y_train)

    # Display the best parameters
    print("Best Hyperparameters:", grid_search.best_params_)

    # Use the best model to make predictions
    best_model = grid_search.best_estimator_
    y_pred = best_model.predict(X_test)
19]

.  Fitting 5 folds for each of 60 candidates, totalling 300 fits
   Best Hyperparameters: {'C': 0.01, 'penalty': 'l1', 'solver': 'liblinear'}
```

Figure 101 Hyperparameter tuning

The code above begins by setting up a hyperparameter grid for tuning a Logistic Regression model using GridSearchCV. The param_grid dictionary specifies a range of values for three hyperparameters: C, penalty, and solver. The C parameter controls the regularization strength and is tested across five values: [0.01, 0.1, 1, 10, 100]. Regularization helps prevent overfitting by penalizing larger coefficients. The penalty parameter specifies the type of regularization to apply and includes 'l1', 'l2', 'elasticnet', and 'none'. The choice of penalty affects how the model handles different types of regularization. Finally, the solver parameter indicates the algorithm used to optimize the logistic regression model and includes 'lbfgs', 'liblinear', and 'saga'. These solvers differ in their optimization methods and efficiency.

Next, GridSearchCV is initialized with the logistic regression model (log_reg) and the parameter grid (param_grid). It is configured to use 5-fold cross-validation (cv=5), meaning the data is split into five subsets to evaluate the model's performance across different combinations of hyperparameters. The n_jobs=-1 setting enables parallel processing, speeding up the computation by using all available processors. The verbose=2 parameter provides detailed output about the progress of the grid search, and scoring='accuracy' specifies that the accuracy metric is used to evaluate model performance. The grid search is executed with grid_search.fit(X_train, y_train), which trains and evaluates the model for each combination of hyperparameters on the training data.

The output from the grid search shows that the best hyperparameters identified are {'C': 0.01, 'penalty': 'l1', 'solver': 'liblinear'}. This result means that, according to the grid search, the optimal configuration for the logistic regression model involves a regularization strength (C) of 0.01, an 'l1' penalty, and the 'liblinear' solver. The chosen model configuration has been evaluated using 5-fold cross-validation and was found to perform best in terms of accuracy with these hyperparameters. The grid search has conducted a total of 300 fits (5 folds × 60 parameter combinations) to find this optimal setup, which will be used to make predictions on the test set to assess the final model's performance.

```
    # Evaluate the model
    print("Classification Report:\n", classification_report(y_test, y_pred))
    print("Accuracy Score:", accuracy_score(y_test, y_pred))
[120]
...  Classification Report:
                precision    recall  f1-score   support

             0       0.93      1.00      0.96      1938
             1       0.00      0.00      0.00       155

      accuracy                           0.93      2093
     macro avg       0.46      0.50      0.48      2093
  weighted avg       0.86      0.93      0.89      2093

  Accuracy Score: 0.9259436215957955
```

Figure 102 Evaluation

The provided code above evaluates the performance of the logistic regression model using two key metrics: the classification report and accuracy score. The classification_report function generates a detailed report that includes precision, recall, and F1-score for each class, as well as average metrics across all classes. Precision measures the proportion of true positive predictions among all positive predictions made by the model. Recall, also known as sensitivity, measures the proportion of actual positives that were correctly identified by the model. The F1-score is the harmonic mean of precision and recall, providing a single metric that balances both concerns. The accuracy_score function calculates the overall accuracy of the model, which is the proportion of correct predictions (both true positives and true negatives) out of the total number of predictions.

The output shows the performance metrics of the model. The classification report indicates that the model achieves high precision (0.93) and recall (1.00) for class 0, but performs poorly on class 1, with a precision, recall, and F1-score of 0.00. This suggests that the model is highly effective at predicting the majority class (class 0) but fails to identify the minority class (class 1) correctly. The accuracy score of approximately 0.93 (or 92.6%) indicates that the model correctly predicts the outcome about 93% of the time overall. However, the imbalance in performance across classes, particularly the failure to predict class 1, highlights potential issues with class imbalance or the need for additional techniques to improve minority class prediction. The macro average metrics (precision 0.46, recall 0.50, F1-score 0.48) reflect the model's balanced performance across both classes, while the weighted averages (precision 0.86, recall 0.93, F1-score 0.89) show the model’s overall performance, giving more weight to the majority class.

```python
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.metrics import confusion_matrix

# Generate the confusion matrix
conf_matrix = confusion_matrix(y_test, y_pred)

# Plotting the confusion matrix with labels
plt.figure(figsize=(8, 6))
sns.heatmap(conf_matrix, annot=True, fmt="d", cmap='Oranges', cbar=True,
            xticklabels=['Not Churned (0)', 'Churned (1)'],
            yticklabels=['Not Churned (0)', 'Churned (1)'])

# Adding titles and labels
plt.title('Confusion Matrix for Logistic Regression', fontsize=16)
plt.xlabel('Predicted Label')
plt.ylabel('True Label')
plt.show()
```

Figure 103 Generate Confusion Matrix

The provided code above is used to visualize the performance of a logistic regression model through a confusion matrix. First, it imports the necessary libraries: matplotlib.pyplot and seaborn for plotting, and confusion_matrix from sklearn.metrics to compute the confusion matrix. The confusion_matrix function generates a matrix that shows the counts of true positives, true negatives, false positives, and false negatives by comparing the true labels (y_test) with the predicted labels (y_pred).

The code then creates a plot to visualize this matrix. By using plt.figure(figsize=(8, 6)), it sets up a figure with dimensions of 8 inches by 6 inches. The sns.heatmap function from seaborn is employed to create a heatmap of the confusion matrix. This function annotates the matrix with integer values, uses the 'Oranges' color map for visualization, and includes a color bar to indicate the scale. The xticklabels and yticklabels parameters are set to label the axes according to whether the predictions and actual values correspond to "Not Churned (0)" or "Churned (1)."

Finally, titles and labels are added to the plot to enhance readability: plt.title provides a title for the confusion matrix, while plt.xlabel and plt.ylabel label the x-axis and y-axis, respectively. The plt.show() function is called to display the plot, offering a clear visual representation of how well the model's predictions align with the actual outcomes. This

visualization helps in assessing the model's performance by highlighting areas of correct and incorrect classifications.

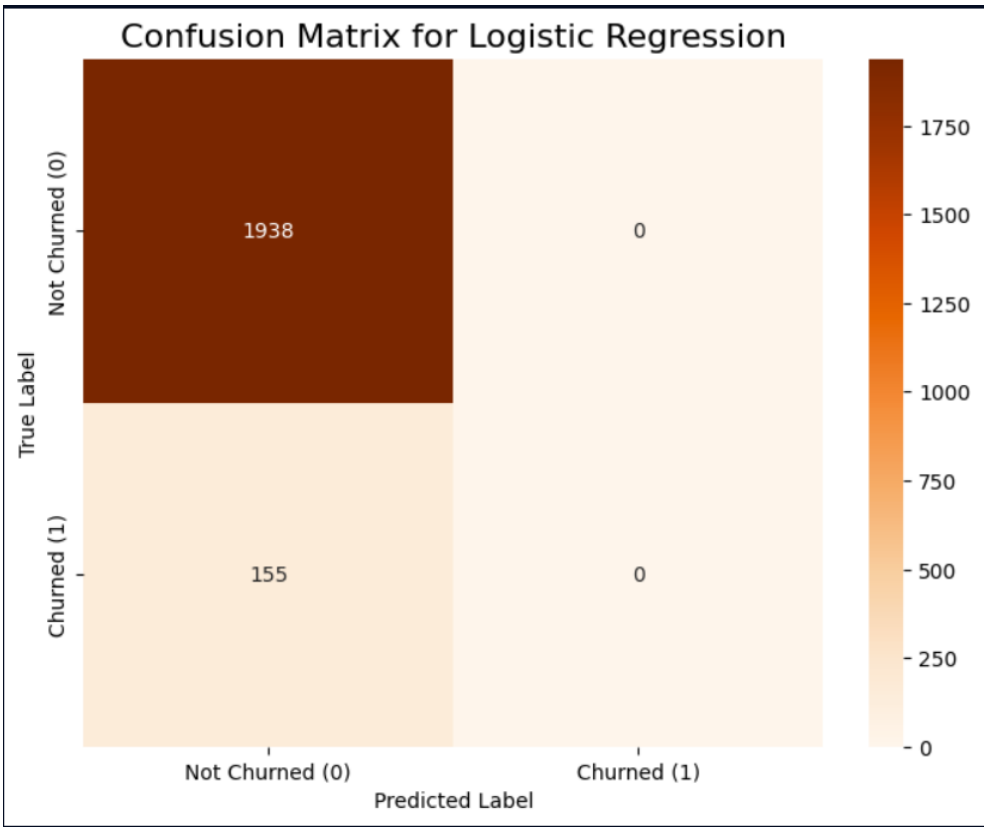

Figure 104 Confusion Matrix

The confusion matrix reveals the performance of the logistic regression model used to predict customer churn. The matrix is divided into four sections that show how well the model distinguished between customers who did and did not churn.

In the top left quadrant, there are 1938 true negatives, indicating that the model correctly identified these customers as not churning. This suggests that the model is reliable in predicting which customers will remain with the bank. However, the bottom left quadrant, with 155 false negatives, shows that the model incorrectly predicted these customers as not churning, even though they did churn. This is a significant error because it means that the model failed to identify at-risk customers, which are crucial for retention efforts.

The other two quadrants, true positives and false positives, are empty, indicating that the model did not predict any customers as churning. This suggests that the model is heavily biased towards predicting non-churn and completely missed the churn cases. This outcome highlights a critical flaw: while the model can predict customer retention, it is ineffective in identifying churn, which is the primary objective of such a model.

This analysis shows that while the model is strong in predicting customers who will not churn, it falls short in predicting those who will. This imbalance indicates the need for further refinement of the model, possibly by addressing issues like data imbalance or exploring more complex models better suited to capture the intricacies of customer churn. The goal is to develop a model that can accurately predict both churn and non-churn, thereby supporting more effective customer retention strategies.

5.3 Examine the correlation between temporal transaction patterns and churn

5.3.1 Descriptive Analysis

a. Data Distribution

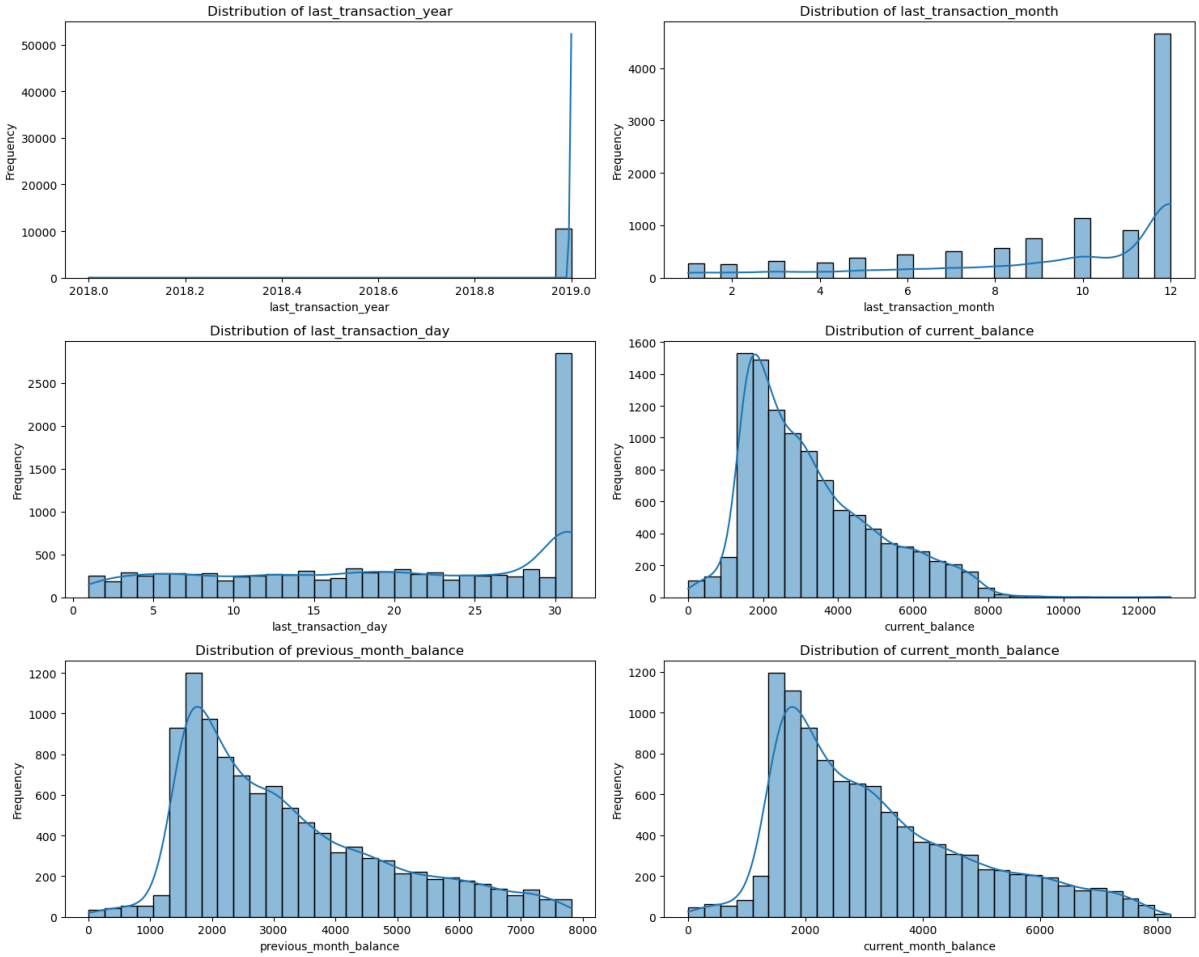

Figure 105 Distribution of Temporal and Financial Behaviour in Relation to Customer Churn

The histograms presented in Figure 105 illustrate the distribution of several critical factors related to the timing of customer transactions and their financial behaviour. These factors are essential for understanding the patterns that may lead to customer churn within the company. Below is the descriptive analysis for each histogram displayed in Figure 105.

Distribution of Last Transaction Year:

- The histogram for last_transaction_year shows a significant concentration of transactions in the year 2019, with very few transactions recorded in 2018. This suggests that most of the customer activity in the dataset is from 2019, possibly due to data collection practices or the recency of customer engagement. The lack of older transactions could indicate that churn is more likely when recent activity is low.

Distribution of Last Transaction Month:

- The histogram for last_transaction_month indicates that a substantial number of transactions occurred towards the end of the year, particularly in December. This could be related to seasonal factors, such as end-of-year spending or holiday promotions, which may influence customer behavior. Understanding these seasonal patterns can help in planning retention strategies during high-risk periods.

Distribution of Last Transaction Day:

- The last_transaction_day histogram shows a notable spike at the end of the month, particularly on the 31st. This pattern may reflect customers' financial behaviours, such as making payments or managing balances at the end of the month. This concentration of activity could be linked to churn if customers are consistently waiting until the last day to make transactions, indicating potential financial stress.

Distribution of Current Balance:

- The distribution of current_balance is right-skewed, with most customers having lower balances and a few holdings much higher amounts. This suggests that customers with lower balances may be more vulnerable to churn, possibly due to financial instability. Targeted financial support or incentives for these customers could help reduce churn rates.

Distribution of Previous Month Balance:

- The previous_month_balance histogram shows a similar right-skewed pattern, indicating that many customers had lower balances at the end of the previous month. This trend could be indicative of ongoing financial challenges, which may lead to higher churn if not addressed through proactive customer engagement and support.

Distribution of Current Month Balance:

- The distribution of current_month_balance mirrors the previous month's balance, with a concentration of customers at lower balance levels. This suggests a pattern of consistent financial difficulty among certain segments of the customer base. Identifying and addressing the needs of these customers could be crucial for improving retention.

Summary and Insights:

- **Temporal Patterns:** The analysis reveals that most transactions occur in the latter part of the year and at the end of the month. This could indicate periods of higher financial activity and potential stress, which may correlate with increased churn.
- **Financial Stability:** The consistent right-skewed distribution in balance-related histograms indicates that customers with lower balances are more likely to churn. Providing tailored financial products or support to these customers could help stabilize their financial situation and reduce churn risk.
- **Opportunities for Improvement:** The data highlights the importance of focusing on customers with low balances and those whose transactions cluster around the end of the month. Early intervention and personalized financial solutions could be key strategies to enhance customer retention.

b. Potential Outliers

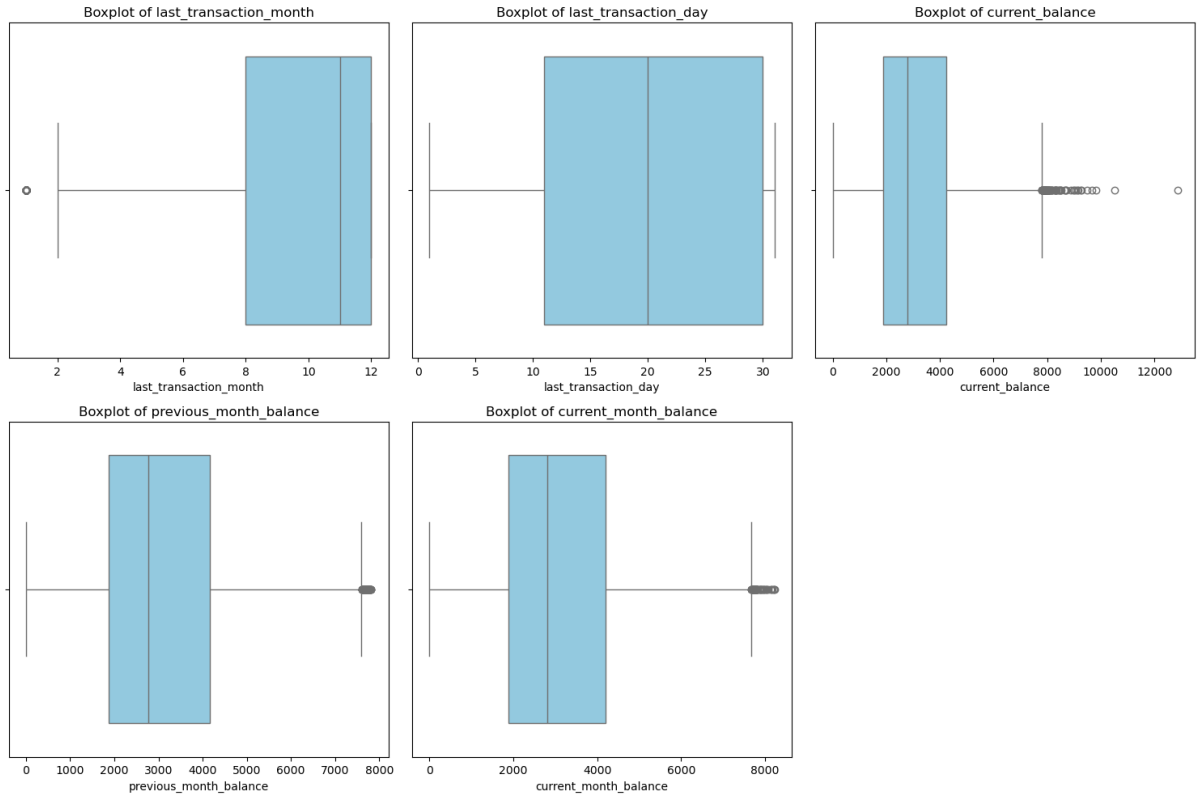

Figure 106 Visualizing Central Tendency, Dispersion, and Outliers for Transaction Timing and Financial Variables

The visualizations in Figure 106 display boxplots for key variables associated with the timing of customer transactions and their financial behaviours. These plots help identify the central tendency, dispersion, and potential outliers that might indicate unusual customer behaviours or profiles.

- **Last Transaction Month**: The boxplot for last_transaction_month indicates a concentration of transactions towards the end of the year, particularly in December. The presence of a few outliers at the beginning of the year suggests that these customers may have different financial patterns or behaviours, possibly indicating seasonal variations or specific financial strategies. Monitoring these outliers can provide insights into customer behaviour throughout the year.

- **Last Transaction Day**: The last_transaction_day boxplot reveals that most transactions occur throughout the month, with a notable increase towards the end of the month. This pattern may indicate that customers are more active financially towards the end of the month, possibly due to payroll cycles or monthly financial planning. Outliers near the end of the month may represent customers who are particularly sensitive to timing, which could influence their risk of churn.
- **Current Balance**: The boxplot for current_balance shows a right-skewed distribution with a significant number of outliers at the higher end of the spectrum. These outliers represent customers with notably higher balances than the average, who may have different financial needs or behaviours. Ensuring that these high-balance customers receive tailored financial services could be crucial in maintaining their loyalty and preventing churn.
- **Previous Month Balance**: Similar to current balance, the previous_month_balance boxplot also shows outliers, indicating customers who had exceptionally high balances at the end of the previous month. These outliers could signify customers with fluctuating financial behaviours, and understanding these fluctuations could help in developing strategies to enhance customer retention.
- **Current Month Balance**: The current_month_balance boxplot highlights outliers similar to the current and previous month balances. Customers with extreme values might require targeted engagement strategies to address their unique financial needs and prevent potential churn.

The presence of outliers in these variables suggests that there are customers whose transaction timing and financial behaviours deviate significantly from the norm. These outliers could represent customers who either present a higher risk of churn due to unusual financial patterns or those who require more personalized financial services to retain their loyalty. Understanding and addressing the needs of these outlier groups is essential for improving customer retention strategies.

c. Correlation Heatmap of Transaction Timing and Financial Variables in Relation to Customer Churn

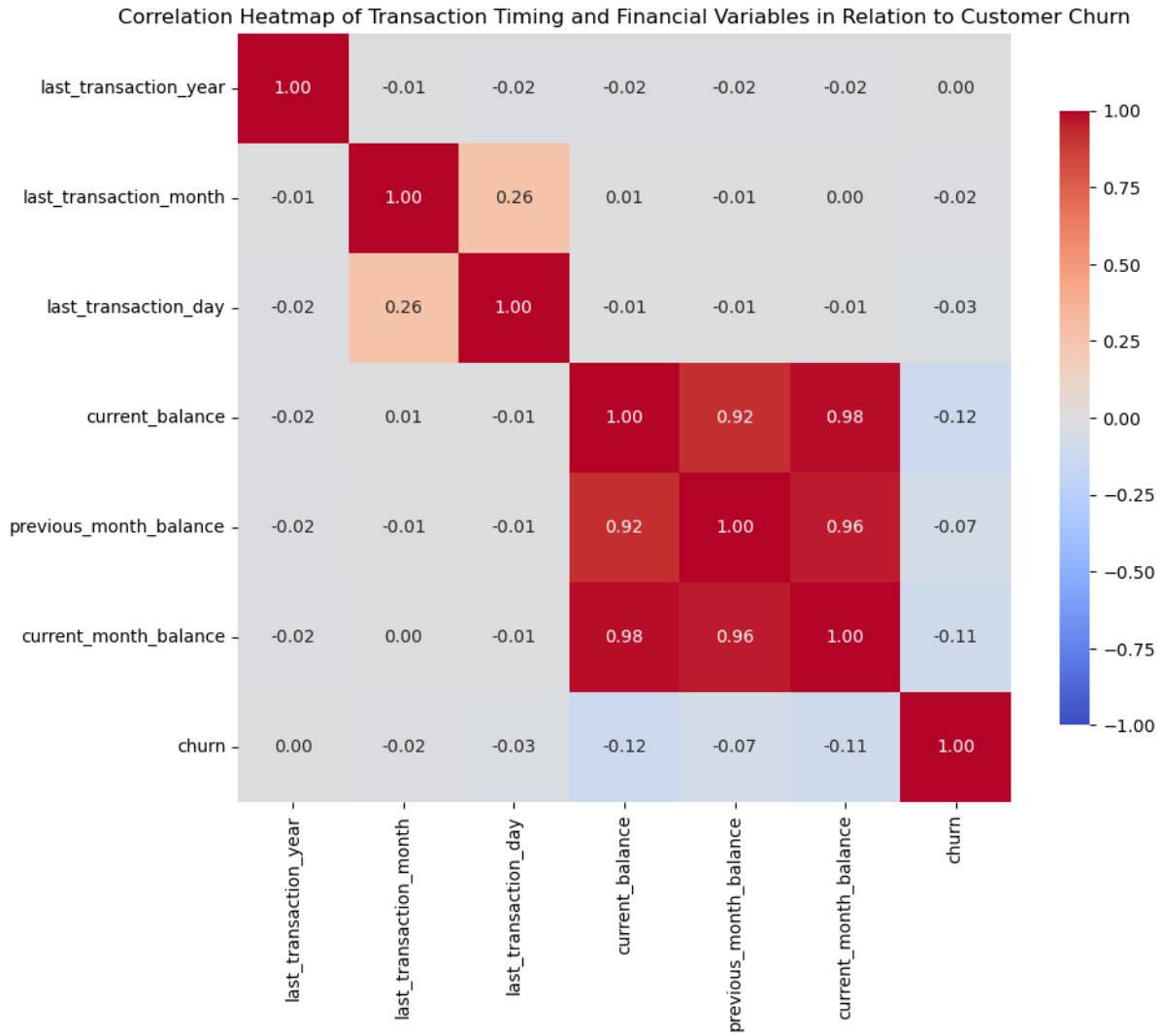

Figure 107 Correlation Heatmap of Transaction Timing and Financial Variables in Relation to Customer Churn

The correlation heatmap above highlights the relationships between key variables related to transaction timing, financial behaviours, and customer churn. While the overall correlations with churn are relatively weak, the heatmap offers valuable insights into the associations between these variables and potential patterns that may influence customer retention.

- **Last Transaction Year:** The correlation between the year of the last transaction and churn is nearly non-existent (0.00). This suggests that the specific year in which a customer's last transaction took place does not play a significant role in determining whether they will churn. This lack of correlation indicates that the timing of transactions across different years has little to no direct impact on customer retention.
- **Last Transaction Month:** The month of the last transaction shows a weak negative correlation with churn (-0.02). This implies that the timing of a transaction within a specific year has minimal influence on the likelihood of customer churn. However, there is a moderate positive correlation between the month and day of the last transaction (0.26), indicating that transactions may be somewhat clustered within certain months. Despite this, the overall impact on churn remains weak.
- **Last Transaction Day:** The day of the last transaction within a month also exhibits a weak negative correlation with churn (-0.03). This indicates that the specific day on which a transaction occurs has little effect on a customer's decision to churn. The slight negative correlation suggests that customers who make transactions later in the month may be marginally less likely to churn, though the effect is not pronounced.
- **Current Balance:** The current balance shows a weak negative correlation with churn (-0.12), suggesting that customers with higher current balances are slightly less likely to churn. This correlation, although weak, indicates that financial stability, as reflected by a higher balance, may contribute to customer retention. The current balance is strongly positively correlated with both the previous month balance (0.92) and the current month balance (0.98), which indicates consistency in financial behaviour over time.
- **Previous Month Balance:** Similar to the current balance, the previous month balance also exhibits a weak negative correlation with churn (-0.07). This suggests that customers who maintained higher balances at the end of the previous month are slightly less likely to churn. The strong positive correlations with current balance and current month balance further reinforce the consistency in financial behaviour among customers.
- **Current Month Balance:** The current month balance shows a weak negative correlation with churn (-0.11), indicating that customers who have higher balances in the current month are marginally less likely to churn. The strong correlations with

current and previous month balances suggest that customers' financial behaviours are stable over time, though this stability alone is not a strong predictor of churn.

Summary

The correlation heatmap provides an insightful overview of the relationships between transaction timing, financial variables, and customer churn. Although the direct correlations with churn are relatively weak, the analysis reveals some important trends and patterns:

- **Financial Stability:** Customers with higher balances, whether current, in the previous month, or in the current month, are slightly less likely to churn. This suggests that financial stability plays a role, albeit a minor one, in customer retention. The strong correlations among the balance variables indicate consistency in financial behaviour, which may be an indicator of long-term customer relationships.
- **Transaction Timing:** The timing of the last transaction, whether by year, month, or day, shows little to no direct correlation with churn. This indicates that while when a transaction occurs may influence financial behaviour, it is not a strong determinant of whether a customer will churn.
- **Interrelationships Among Financial Variables:** The strong correlations among the financial variables themselves suggest that while individual factors like current balance or previous month balance may not independently predict churn, they are closely related to each other. This interconnectedness highlights the importance of considering multiple financial factors together when analysing customer behaviour.

In conclusion, while no single variable strongly predicts churn, the heatmap underscores the importance of financial stability and consistent transaction behaviour in customer retention. To better understand and predict churn, a more complex model that considers the interactions between these variables may be necessary. This approach could provide deeper insights into the factors that influence customer loyalty and help develop more targeted strategies for reducing churn.

d. Multivariate Regression Analysis

The multivariate regression analysis was conducted to explore the relationship between customer churn (the dependent variable) and several independent variables related to the timing of customer transactions and financial balances. The goal was to determine how these factors collectively influence the likelihood of a customer churning.

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                  churn   R-squared:                       0.074
Model:                            OLS   Adj. R-squared:                  0.073
Method:                 Least Squares   F-statistic:                     92.90
Date:                Fri, 16 Aug 2024   Prob (F-statistic):          2.18e-167
Time:                        22:51:40   Log-Likelihood:                -455.93
No. Observations:               10463   AIC:                             931.9
Df Residuals:                   10453   BIC:                             1004.
Df Model:                           9
Covariance Type:            nonrobust
================================================================================================
                                   coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------------------------
const                          -70.9690    294.950     -0.241      0.810    -649.128     507.190
last_transaction_year            0.0352      0.146      0.241      0.810      -0.251       0.322
last_transaction_month          -0.0009      0.001     -1.111      0.267      -0.003       0.001
last_transaction_day            -0.0003      0.000     -1.025      0.305      -0.001       0.000
current_balance              -5.048e-05    7.5e-06     -6.734      0.000   -6.52e-05   -3.58e-05
previous_month_balance       -6.505e-05   7.22e-06     -9.012      0.000   -7.92e-05   -5.09e-05
current_month_balance        -4.215e-05   1.04e-05     -4.041      0.000   -6.26e-05   -2.17e-05
current_month_debit              0.0001   1.41e-05      7.969      0.000    8.47e-05       0.000
previous_month_credit        -6.768e-06   1.74e-05     -0.388      0.698   -4.09e-05    2.74e-05
average_monthly_balance_prevQ    0.0002   6.92e-06     21.797      0.000       0.000       0.000
==============================================================================
```

Figure 108 Multivariate Regression Analysis

Output Overview

- **R-squared (0.074)**: The R-squared value of 0.074 indicates that the independent variables (transaction timing and financial balances) together explain 7.4% of the variation in customer churn. While this suggests that these factors have some explanatory power, the majority of the variability in churn is influenced by other factors not included in this model.
- **Adjusted R-squared (0.073)**: The adjusted R-squared, which accounts for the number of predictors in the model, is slightly lower at 0.073. This suggests that the addition of more variables slightly improves the model's ability to predict churn, but the overall explanatory power remains modest.

- **F-statistic (92.90) and Prob (F-statistic) (2.18e-167)**: The F-statistic tests the overall significance of the regression model. Here, the F-statistic is 92.90 with a highly significant p-value (2.18e-167), indicating that the model is statistically significant, meaning that at least one of the predictors has a significant effect on churn.

Coefficients and Significance

- **Constant (-70.9690)**: The constant (intercept) of -70.9690 represents the expected value of churn when all independent variables are set to zero. While this value is not meaningful in isolation (since it doesn't make sense for these variables to be zero), it serves as a baseline for understanding the impact of the other variables.
- **Last Transaction Year (0.0352, t = 0.241, p = 0.810)**: The coefficient for last_transaction_year is 0.0352, indicating a positive but very weak relationship between the year of the last transaction and churn. However, with a t-statistic of 0.241 and a p-value of 0.810, this relationship is not statistically significant.
- **Last Transaction Month (-0.0009, t = -1.111, p = 0.267)**: The coefficient for last_transaction_month is -0.0009, suggesting a very slight negative relationship between the month of the last transaction and churn. The t-statistic of -1.111 and the p-value of 0.267 indicate that this relationship is not statistically significant.
- **Last Transaction Day (-0.0003, t = -1.025, p = 0.305)**: The coefficient for last_transaction_day is -0.0003, showing a weak negative relationship with churn. However, with a t-statistic of -1.025 and a p-value of 0.305, this effect is not statistically significant.
- **Current Balance (-5.048e-05, t = -6.734, p = 0.000)**: The coefficient for current_balance is -5.048e-05, suggesting that as the current balance increases, the probability of churn decreases slightly. The t-statistic of -6.734 is large, and the p-value is 0.000, indicating that this relationship is highly statistically significant.
- **Previous Month Balance (-6.505e-05, t = -9.012, p = 0.000)**: The coefficient for previous_month_balance is -6.505e-05, indicating a significant negative relationship with churn. The t-statistic of -9.012 and the p-value of 0.000 further confirm its significance.

- **Current Month Balance (-4.215e-05, t = -4.041, p = 0.000)**: The coefficient for current_month_balance is -4.215e-05, also showing a significant negative relationship with churn. The t-statistic of -4.041 and the p-value of 0.000 indicate that this variable is a significant predictor of churn.
- **Current Month Debit (0.0001, t = 7.969, p = 0.000)**: The coefficient for current_month_debit is 0.0001, suggesting a positive relationship with churn. This relationship is statistically significant, as indicated by the t-statistic of 7.969 and the p-value of 0.000.
- **Previous Month Credit (-6.768e-06, t = -0.388, p = 0.698)**: The coefficient for previous_month_credit is -6.768e-06, showing a very weak negative relationship with churn, but the t-statistic of -0.388 and the p-value of 0.698 indicate that this relationship is not statistically significant.
- **Average Monthly Balance PrevQ (0.0002, t = 21.797, p = 0.000)**: The coefficient for average_monthly_balance_prevQ is 0.0002, suggesting a significant positive relationship with churn. The t-statistic of 21.797 and the p-value of 0.000 indicate that this variable is a strong predictor of churn.

Interpretation

- **Coefficients**: The coefficients provide insights into the direction and magnitude of the relationship between each independent variable and churn. Positive coefficients indicate that an increase in the variable is associated with an increase in churn, while negative coefficients suggest the opposite.
- **t-statistic**: The t-statistic measures the strength of the evidence against the null hypothesis (that the coefficient is zero). Larger absolute values indicate stronger evidence. In this model, current_balance, previous_month_balance, current_month_balance, and average_monthly_balance_prevQ have significant t-statistics.
- **p-value**: The p-value helps determine the statistical significance of each coefficient. A p-value below 0.05 indicates significance. In this model, several variables, including current_balance, previous_month_balance, current_month_balance, current_month_debit, and average_monthly_balance_prevQ, are statistically significant.
- **R-squared**: The R-squared value suggests that the model explains only a modest portion of the variance in churn. This implies that other variables not included in the model may be important in predicting customer churn.

Conclusion

The multivariate regression model reveals that while variables such as current_balance, previous_month_balance, current_month_balance, and average_monthly_balance_prevQ are significant predictors of churn, the overall model only explains a small fraction of the variance in churn behaviour. This suggests that additional variables or more complex models may be necessary to fully understand the factors driving customer churn.

e. OLAP Dashboard

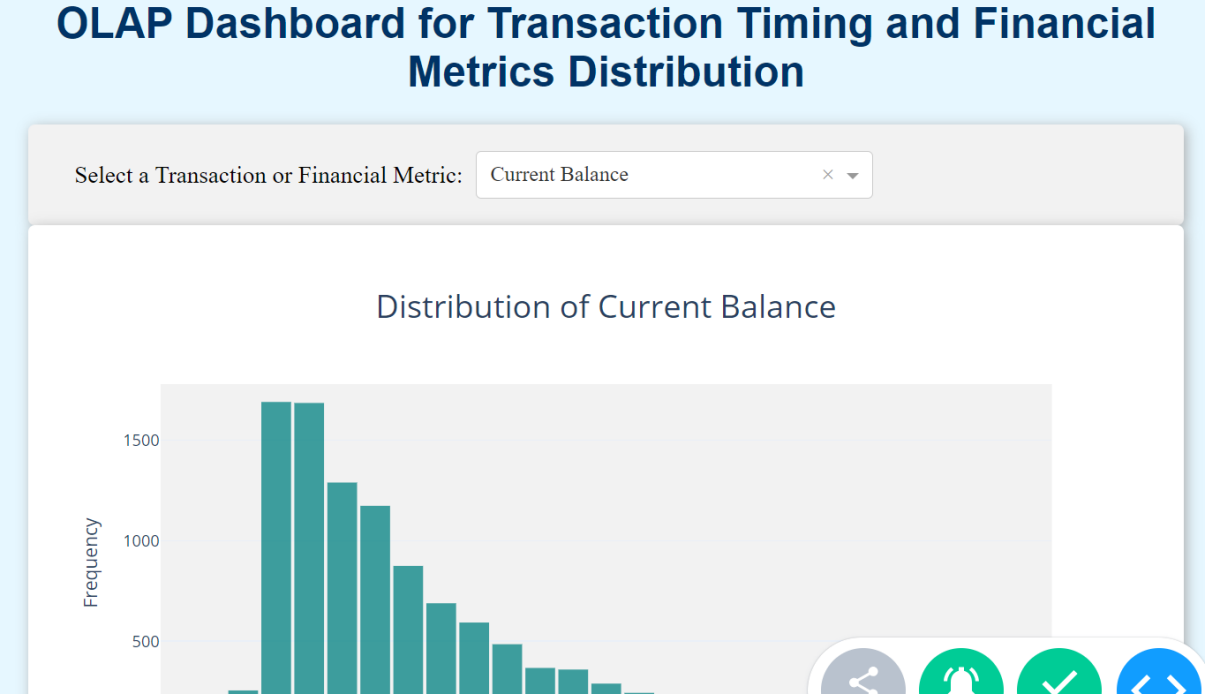

Figure 109 OLAP Dashboard

The OLAP (Online Analytical Processing) Dashboard for "Transaction Timing and Financial Metrics Distribution" is a sophisticated data visualization tool designed to help users explore the temporal aspects of customer transactions and their associated financial metrics. This dashboard is developed using Python's Dash and Plotly libraries, providing an interactive interface for users to analyse how different temporal and financial factors relate to customer behaviour, particularly focusing on their transaction history and balance fluctuations.

Objective

The primary objective of this dashboard is to facilitate the temporal analysis of customer transactions by visualizing the distribution of key metrics such as transaction dates and various financial balances. The columns of interest include 'last_transaction_year,' 'last_transaction_month,' 'last_transaction_day,' 'current_balance,' 'previous_month_balance,' 'current_month_balance,' 'current_month_debit,' 'previous_month_credit,' and

'average_monthly_balance_prevQ.' These metrics were selected to provide a comprehensive view of how customer transactions and financial behaviours evolve over time.

Dashboard Structure and Layout

The dashboard is structured to be user-friendly and visually appealing. At the top, a header titled "OLAP Dashboard for Transaction Timing and Financial Metrics Distribution" clearly communicates the purpose of the tool. The layout is designed with a focus on clarity and ease of use, featuring a dropdown menu that allows users to select the specific metric they wish to analyse. The selected metric's distribution is then visualized in the form of a histogram, providing an immediate and intuitive understanding of the data's behaviour.

The design aesthetic of the dashboard incorporates a professional colour scheme, with teal accents used for the histogram bars to ensure a pleasing visual experience. The entire interface is built with responsiveness in mind, ensuring that it performs well across various devices and screen sizes.

Functionality and Interaction

The core functionality of the dashboard revolves around the interactive dropdown menu, which lists all the columns of interest for temporal and financial analysis. Users can easily switch between different metrics, such as the year of the last transaction or the current month's balance, to dynamically update the displayed histogram. This interactivity allows for real-time data exploration, enabling users to uncover patterns and insights that may not be immediately apparent from static reports.

When a user selects a specific metric, the dashboard generates a histogram using Plotly Express. This histogram visualizes the distribution of the selected metric, showing how frequently different values occur within the dataset. The layout of the histogram is meticulously crafted to ensure that it provides clear and accurate information, with titles and axis labels automatically adjusted to reflect the chosen metric.

Technical Implementation

The dashboard is powered by a combination of Dash for creating the web application and Plotly Express for generating the histograms. The application's interactivity is driven by a callback function that updates the histogram based on the user's selection from the dropdown menu. The callback function retrieves the selected column from the dataset and generates a histogram with 50 bins, ensuring that the distribution is detailed and informative.

The histogram itself is customized with various layout options to enhance readability and user experience. For instance, the title is centered and prominently displayed, while the x-axis and y-axis titles are formatted to provide clear context. Additionally, the histogram bars are coloured in teal to maintain a consistent and professional aesthetic throughout the dashboard.

Conclusion

The OLAP Dashboard for Transaction Timing and Financial Metrics Distribution is a powerful tool for conducting temporal analysis of customer transactions. By providing an interactive platform for visualizing key financial metrics, this dashboard enables users to gain deep insights into customer behaviour and transaction patterns. The careful design and implementation ensure that the dashboard is both functional and aesthetically pleasing, making it an invaluable resource for data analysts and decision-makers seeking to understand and leverage customer transaction data.

5.3.2 Predictive Analysis – Decision Tree Classifier

a. Import Library

```
import pandas as pd
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score, precision_score, classification_report, confusion_matrix
import matplotlib.pyplot as plt
import seaborn as sns
```

Figure 110 Code Snippet for Libraries Used

The code utilizes essential Python libraries for machine learning and data analysis. These libraries together provide a full range of tools for data preprocessing, model training, evaluation, and visualization.

b. Model Building

```
# Selecting the relevant columns for the model
X = dataset[['last_transaction_year', 'last_transaction_month', 'last_transaction_day',
             'current_balance', 'previous_month_balance', 'current_month_balance',
             'current_month_debit', 'previous_month_credit', 'average_monthly_balance_prevQ']]
y = dataset['churn']
```

Figure 111 Code Snippet for Defining Features

The selected features include transaction timing (last_transaction_year, last_transaction_month, last_transaction_day) and various financial metrics (current_balance, previous_month_balance, current_month_balance, current_month_debit, previous_month_credit, average_monthly_balance_prevQ). These features were chosen for their potential impact on customer churn prediction.

```
# Splitting the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)
```

Figure 112 Code Snippet for Splitting Data

The data is split into training and testing sets using the train_test_split function, with 30% of the data reserved for testing. This ensures that the model can be evaluated on unseen data to gauge its predictive performance.

```
# Defining the parameter grid
param_grid = {
    'max_depth': [3, 5, 10, 15, 20],
    'min_samples_split': [2, 5, 10],
    'min_samples_leaf': [1, 2, 4],
    'criterion': ['gini', 'entropy']
}
```

Figure 113 Code Snippet for GridSearchCV Setup

Hyperparameter tuning is conducted using GridSearchCV to identify the optimal set of parameters for the Decision Tree model. The grid search systematically tests various combinations of max_depth, min_samples_split, min_samples_leaf, and criterion to find the best model configuration.

```
# Initializing the Decision Tree Classifier
clf = DecisionTreeClassifier(random_state=42)

# Setting up GridSearchCV
grid_search = GridSearchCV(estimator=clf, param_grid=param_grid,
                           cv=5, n_jobs=-1, verbose=2, scoring='accuracy')

# Fitting the model
grid_search.fit(X_train, y_train)
```

Figure 114 Code Snippet for Model Building and Training

A Decision Tree Classifier is initialized and tuned using the parameters defined in param_grid. The GridSearchCV runs a cross-validated grid search to determine the best combination of hyperparameters.

c. Hyperparameter Tuning

```
# Best parameters
print("Best Parameters:", grid_search.best_params_)
```

```
Fitting 5 folds for each of 90 candidates, totalling 450 fits
Best Parameters: {'criterion': 'entropy', 'max_depth': 3, 'min_samples_leaf': 1, 'min_samples_split': 2}
```

Figure 115 Best Parameters from GridSearchCV

The hyperparameter tuning process aimed to optimize the performance of the Decision Tree Classifier by exploring various parameter combinations. The grid search evaluated 90 different parameter sets, testing each combination across five cross-validation folds, leading to a total of 450 model evaluations. This exhaustive search identified the best-performing hyperparameters: a criterion of 'entropy', a maximum tree depth of 3, a minimum of 1 sample per leaf node, and a minimum of 2 samples required to split an internal node.

These parameters suggest a model that prioritizes simplicity and interpretability, as indicated by the shallow tree depth. The use of 'entropy' as the criterion for splitting helps ensure that each split in the tree provides the maximum possible information gain, which is crucial for distinguishing between churners and non-churners. The selected parameters reflect a balanced approach, aimed at preventing the model from overfitting while maintaining robust predictive performance. The model achieved an accuracy of 93.25% after tuning, which underscores the effectiveness of these parameters in enhancing the classifier's ability to generalize to unseen data.

d. Evaluation

```
# Using the best estimator to predict on the test set
best_clf = grid_search.best_estimator_
y_pred = best_clf.predict(X_test)

# Evaluating the model
accuracy = accuracy_score(y_test, y_pred)
precision = precision_score(y_test, y_pred, average='binary')

# Displaying the results
print(f'Accuracy: {accuracy:.4f}')
print(f'Precision: {precision:.4f}')
print('\nClassification Report:')
print(classification_report(y_test, y_pred))
```

```
Classification Report:
              precision    recall  f1-score   support

           0       0.94      0.99      0.96      2909
           1       0.60      0.23      0.33       230

    accuracy                           0.93      3139
   macro avg       0.77      0.61      0.65      3139
weighted avg       0.92      0.93      0.92      3139
```

Figure 116 Classification Report & Results

The classification report provides a detailed look at the model's performance across both the churn (Class 1) and non-churn (Class 0) categories, using key metrics such as precision, recall, and F1-score. For the non-churn class, the model exhibits strong performance with a precision of 0.94 and a recall of 0.99. This means that 94% of the instances predicted as non-churn are correct, and the model successfully identifies nearly all non-churn instances. The high F1-score of 0.96 further confirms the model's reliability in handling non-churn cases.

However, the model's performance deteriorates when predicting churn (Class 1). The precision for churn is 0.60, indicating that 40% of churn predictions are false positives. More concerning is the recall for churn, which stands at only 0.23. This low recall suggests that the model fails to identify a significant portion of actual churners, leading to a high number of false negatives. Consequently, the F1-score for churn is 0.33, which reflects the model's difficulty in balancing precision and recall for this minority class. The disparity in performance between the two classes highlights the challenges posed by class imbalance in the dataset, where the model is biased toward the majority class (non-churn).

```
# Confusion Matrix
conf_matrix = confusion_matrix(y_test, y_pred)
sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='Blues')
plt.title('Confusion Matrix')
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.show()
```

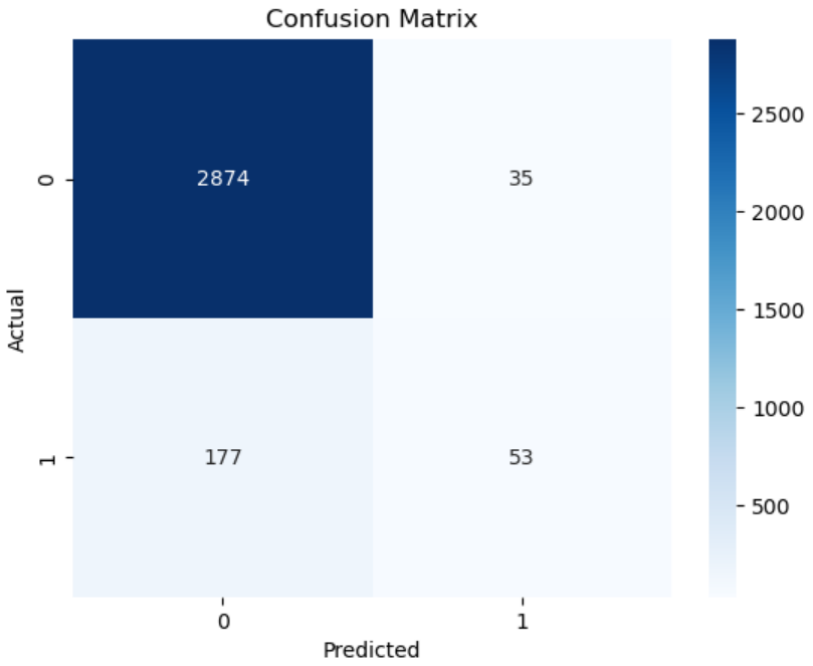

Figure 117 Confusion Matrix

The confusion matrix provides a clear visual representation of the model's predictions compared to the actual outcomes. The matrix reveals that the model correctly identified 2874 out of 2909 non-churn cases (True Negatives) and 53 out of 230 churn cases (True Positives). However, it also misclassified 177 non-churn cases as churn (False Positives) and failed to identify 35 churn cases (False Negatives).

The high number of true negatives reflects the model's proficiency in accurately predicting non-churners, which contributes to its overall high accuracy. However, the substantial number of false positives indicates that the model tends to over-predict churn, potentially leading to unnecessary interventions for customers who are not at risk. More critically, the presence of 177 false negatives—instances where actual churners were missed—suggests that the model may not be reliable in flagging customers who are genuinely at risk of leaving. This imbalance in prediction accuracy between the two classes is a significant concern, as it could result in missed opportunities to retain at-risk customers and unnecessary resource allocation toward non-churners.

In summary, while the Decision Tree Classifier performs well in predicting the majority class (non-churn), its effectiveness in identifying the minority class (churn) is limited. This highlights the need for further model refinement, possibly through additional techniques like class balancing or more sophisticated modeling approaches, to improve the detection of churners and enhance the overall utility of the predictive model.

5.4 Analyse the impact of demographics and customer tenure on churn rates

5.4.1 Descriptive Analysis

a. Data Distribution

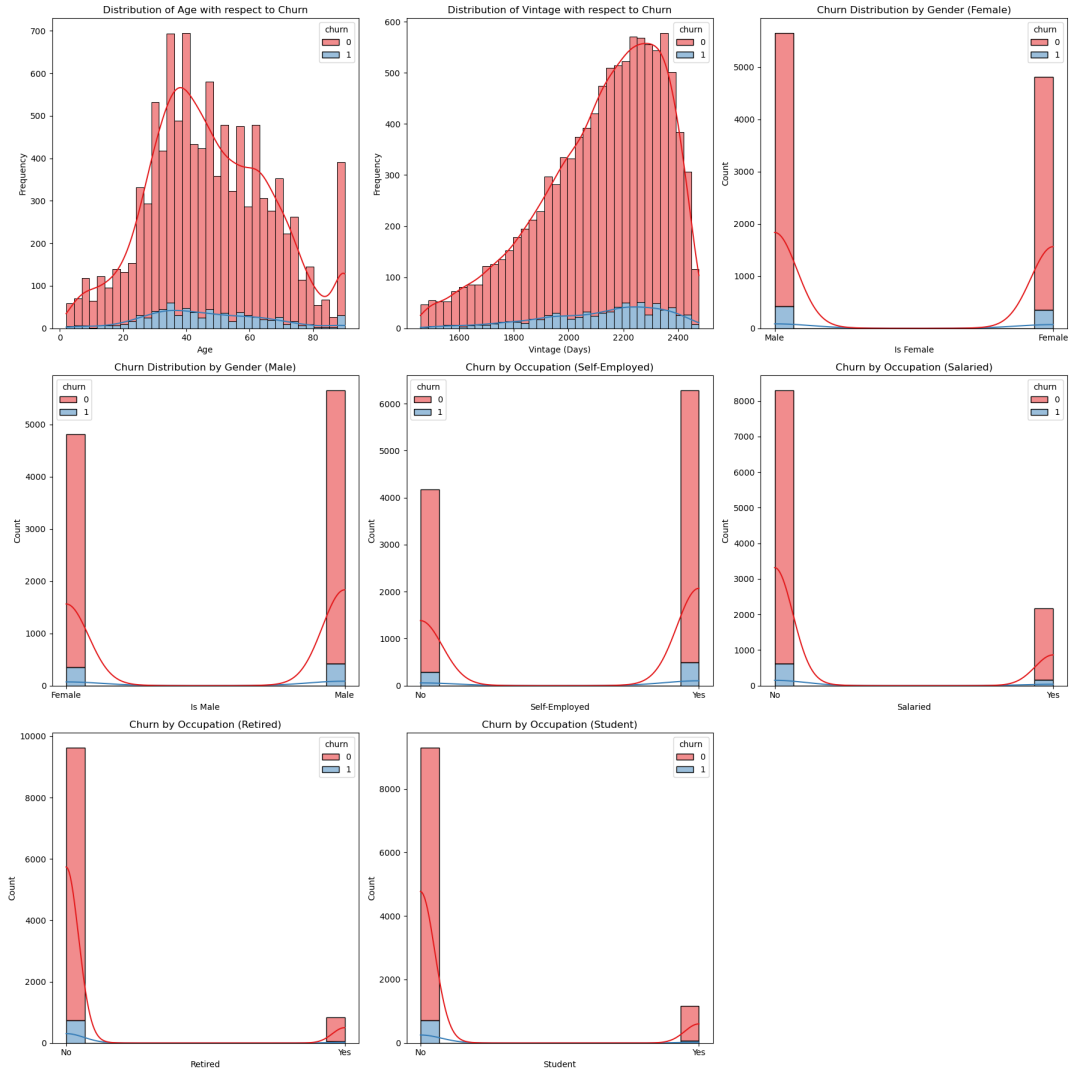

Figure 118 Customer Churn Distribution Across Demographics and Tenure

The histograms presented in the above figure show the distribution of several key factors that are important for understanding customer churn within the company. Below is the descriptive analysis for each histogram displayed in the Figure 118.

Age Distribution:

- **High Frequencies**: The histogram shows high frequencies, particularly among middle-aged customers. The distribution appears multimodal, with distinct peaks across various age groups.
- **Churn Observation**: Churn is notably higher among middle-aged customers, indicating that this demographic may be facing specific challenges or circumstances making them more likely to leave the company. This insight suggests that targeted strategies to retain customers in this age range could be beneficial.

Vintage Distribution:

- **Skewness**: The vintage histogram shows a right-skewed distribution, meaning many customers have a relatively short tenure with the company.
- **Churn Correlation**: Higher churn rates are concentrated among those with shorter customer relationships, highlighting the importance of implementing early engagement and retention strategies soon after customer onboarding.

Gender Distribution:

- **Female Customers**: The histogram for female customers reveals that churn occurs across all segments, with a slight increase among specific subgroups. This data suggests that female customers might benefit from targeted retention efforts.
- **Male Customers**: For male customers, the distribution shows churn across various age groups, with a noticeable concentration among younger males. This suggests that strategies tailored to address the unique needs of younger male customers could be effective in reducing churn.

Occupation Distribution:

- **Self-Employed**: The histogram indicates that self-employed customers experience higher churn rates compared to other occupations. This could be attributed to the financial uncertainties associated with self-employment, indicating a need for financial products offering greater flexibility and stability.
- **Salaried**: Salaried customers exhibit a more stable churn distribution, suggesting that this group may be more financially stable or content with the services provided.

Enhancing loyalty programs for this segment could help maintain their satisfaction and reduce churn.

- **Retired**: Retired customers show lower churn rates, reflecting a potentially loyal customer base. Offering products tailored to the specific needs of retired customers could further enhance this loyalty.
- **Student**: Students display a higher churn rate, likely due to their transient financial situations or changing needs as they progress through different life stages. Developing engagement strategies that evolve with the student's journey could help reduce churn in this segment.

Summary and Insights:

- **Age and Gender**: The analysis indicates higher churn among middle-aged customers and among certain segments of both male and female customers. Tailoring strategies to meet the specific needs of these groups could lead to improved retention.
- **Financial Stability**: The data suggests that customers with lower financial stability, particularly those with shorter tenures, are more likely to churn. Products and services aimed at improving financial health and stability could be beneficial in retaining these customers.
- **Occupation**: Varying churn rates across different occupations indicate that self-employed and student customers are more prone to churn. Providing targeted financial products and support for these groups could mitigate this issue.
- **Opportunities for Improvement**: The findings highlight several opportunities for enhancing customer retention, particularly among newer customers and those facing financial instability. By focusing on early engagement, providing tailored financial products, and addressing the specific needs of different demographic groups, the company can strengthen customer loyalty and reduce churn.

b. Potential Outliers

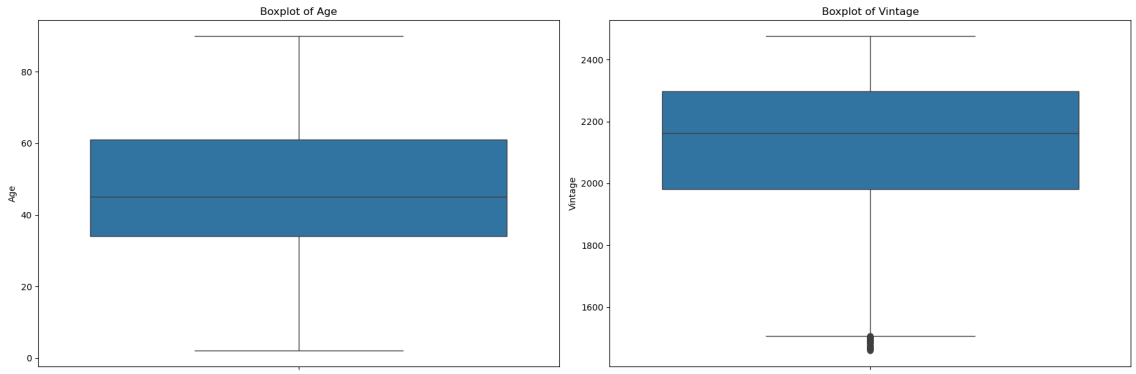

Figure 119 Visualizing Central Tendency, Dispersion, and Outliers for two Key Variables

Analysis of Customer Demographics and Tenure:

The customers' demographic behaviours exhibit varying levels of stability and potential outliers, as depicted in the attached boxplots for age and vintage.

Age Distribution:

- **Symmetry and Stability:** The age distribution appears relatively symmetrical with a moderate interquartile range, suggesting that many customers fall within a mature age bracket, typically between their mid-40s to 60s. This indicates a stable and consistent demographic profile among the customer base, which is a positive sign of long-term customer loyalty and satisfaction.
- **Younger Outliers:** The presence of a few younger outliers suggests that there are younger customers who may exhibit different engagement or financial behaviours compared to the majority. These customers might require targeted retention strategies to ensure their continued loyalty.

Vintage Distribution:

- **Variability in Tenure:** The vintage distribution, which reflects the length of the customer relationship with the company, shows more variability. Most customers have

a tenure ranging from around 2000 to 2400 days, indicating long-standing relationships with the company.

- **Short-Tenure Outliers:** The existence of outliers, especially those with significantly shorter tenures, highlights that newer customers might be at a higher risk of churn. These outliers emphasize the importance of implementing strategies focused on early engagement and fostering long-term relationships to prevent early-stage churn.

Summary:

- **Stability and Loyalty:** The overall distribution of age and vintage suggests that the customer base is largely stable and mature, with long-standing relationships being a positive indicator of customer loyalty and satisfaction.
- **Outlier Significance:** The outliers in both age and vintage distributions indicate segments of the customer base that may require more personalized strategies. Younger customers and those with shorter tenures may benefit from tailored retention strategies to align with their specific needs and reduce the likelihood of churn.

This analysis highlights the importance of not only understanding the general trends within the customer base but also identifying and addressing the unique needs of outlier segments to enhance overall customer retention.

c. Correlation Heatmap

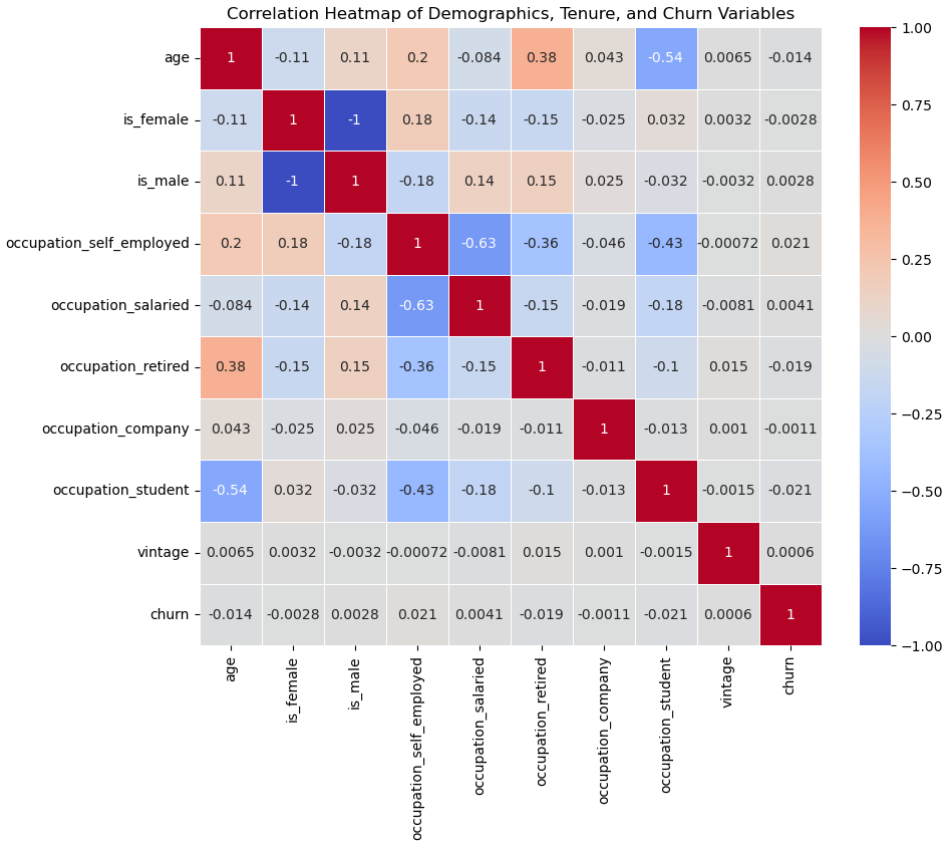

Figure 120 Correlation Heatmap of Demographics, Tenure, and Churn Variables

The correlation heatmap provides an overview of the relationships between various demographic factors, customer tenure, and churn. While the correlations between these variables and churn are generally weak, the heatmap offers valuable insights into potential underlying patterns and associations.

Age:

- **Retirement and Age:** There is a moderate positive correlation between age and the occupation retired (0.38), indicating that older customers are more likely to be retired. This correlation aligns with the typical lifecycle where customers tend to retire as they age.

- **Students and Age:** Conversely, there is a moderate negative correlation between age and the occupation student (-0.54), reflecting the expected relationship where younger customers are predominantly students. This correlation underscores the distinct demographic segment of younger, student customers.
- **Churn:** The correlation between age and churn is weak (-0.014), suggesting that age alone does not have a significant impact on whether a customer decides to churn.

Gender:

- **Perfect Negative Correlation:** As expected, the variables is_female and is_male show a perfect negative correlation (-1.0). This reflects the binary nature of gender data in the dataset.
- **Gender and Churn:** Both gender variables exhibit very weak correlations with churn, indicating that gender does not play a major role in predicting customer churn. The slight correlations with occupation variables suggest that any gender influence on churn might be indirect, potentially through occupational roles.

Occupation:

- **Self-Employed:** This occupation category shows a moderate positive correlation with age (0.2), meaning that older customers are more likely to be self-employed. There is also a moderate negative correlation with being salaried (-0.63), reflecting the mutual exclusivity of these occupations. The weak correlation with churn (0.021) indicates that being self-employed is not a strong predictor of customer churn.
- **Salaried:** Salaried customers exhibit a strong negative correlation with being self-employed (-0.63) and a weak correlation with churn (0.0041). This suggests that salaried status alone does not significantly predict churn, though it may contribute to a broader set of factors influencing churn.
- **Retired:** The occupation category of retired is positively correlated with age (0.38), confirming that older customers are more likely to be retired. The weak negative correlation with churn (-0.019) suggests that retirement status does not significantly affect the likelihood of churn.

- **Student:** Students show a negative correlation with age (-0.54), which is expected given that students are generally younger. The weak negative correlation with churn (-0.021) implies that student status has a limited impact on the likelihood of churning.

Vintage:

- **Length of Relationship and Churn:** The correlation between vintage (length of the customer relationship) and churn is almost non-existent (0.0006). This suggests that the duration of a customer's relationship with the company does not strongly influence their likelihood of churning.

Summary:

- **Weak Correlations with Churn:** The heatmap reveals that individual demographic variables and tenure have weak correlations with churn. This suggests that churn cannot be easily predicted by a single factor alone.
- **Interrelationships among Demographic Factors:** There are notable correlations among the demographic factors themselves, such as between age and occupation categories. These relationships indicate that while one variable might not predict churn effectively on its own, a combination of variables could provide more predictive power.
- **Need for a Comprehensive Approach:** The analysis emphasizes the importance of a more comprehensive approach to understanding customer churn, potentially through the use of complex models that capture interactions between various demographic factors and tenure.

This heatmap analysis highlights the need to consider multiple factors in tandem when predicting churn, as the weak individual correlations suggest that customer behaviour is influenced by a combination of demographic and tenure-related factors.

d. Multivariate Regression Analysis

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                  churn   R-squared:                       0.016
Model:                            OLS   Adj. R-squared:                  0.015
Method:                 Least Squares   F-statistic:                     55.00
Date:                Fri, 16 Aug 2024   Prob (F-statistic):           2.89e-35
Time:                        00:15:55   Log-Likelihood:                -776.59
No. Observations:               10463   AIC:                             1561.
Df Residuals:                   10459   BIC:                             1590.
Df Model:                           3
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const          0.1379      0.025      5.416      0.000       0.088       0.188
age           -0.0001      0.000     -0.960      0.337      -0.000       0.000
vintage     1.855e-06   1.14e-05      0.163      0.871   -2.05e-05    2.42e-05
current_balance -1.9e-05   1.49e-06    -12.768      0.000   -2.19e-05   -1.61e-05
==============================================================================
Omnibus:                     6603.413   Durbin-Watson:                   1.985
Prob(Omnibus):                  0.000   Jarque-Bera (JB):            47194.086
Skew:                           3.169   Prob(JB):                         0.00
Kurtosis:                      11.251   Cond. No.                     4.13e+04
==============================================================================
```

Figure 121 Multivariate Regression Analysis

The multivariate regression analysis was conducted to examine the relationship between customer churn (the dependent variable) and three key independent variables: age, vintage (length of customer relationship), and current balance. The goal was to understand how these factors together influence the likelihood of a customer churning.

Output Overview

- **R-squared (0.016)**: The R-squared value for this model is 0.016, indicating that the independent variables (age, vintage, and current balance) collectively explain only 1.6% of the variation in customer churn. This low R-squared value suggests that these variables alone do not capture much of the complexity behind why customers decide to churn, implying that other factors not included in this model might be more influential.

- **Adjusted R-squared (0.015)**: The adjusted R-squared, which accounts for the number of predictors in the model, is slightly lower at 0.015. This further supports the notion that adding more variables in this case does not significantly improve the model's ability to predict churn.
- **F-statistic (55.00) and Prob (F-statistic) (2.89e-35)**: The F-statistic tests whether the overall regression model is a good fit for the data. Here, the F-statistic is 55.00 with a highly significant p-value (2.89e-35). This indicates that the model, despite its low R-squared, is statistically significant, meaning that at least one of the predictors has a significant effect on churn.

Coefficients and Significance

- **Constant (0.1379)**: The constant (intercept) represents the expected value of churn when all independent variables are set to zero. In this context, a constant of 0.1379 suggests a baseline churn rate of about 13.79% when no other factors are considered.
- **Age (-0.0001, t = -0.960, p = 0.337)**: The coefficient for age is -0.0001, indicating a very slight negative relationship between age and churn. However, the t-statistic of -0.960 and the p-value of 0.337 suggest that this relationship is not statistically significant. Therefore, age does not appear to be a meaningful predictor of churn in this model.
- **Vintage (1.855e-06, t = 0.163, p = 0.871)**: The coefficient for vintage is 1.855e-06, which also indicates an extremely small and positive relationship with churn. The t-statistic of 0.163 and the p-value of 0.871 show that this effect is not statistically significant, meaning that the length of the customer relationship (vintage) does not significantly impact the likelihood of churn.
- **Current_balance (-1.9e-05, t = -12.768, p = 0.000)**: The coefficient for current_balance is -1.9e-05, which suggests that as current balance increases, the probability of churn decreases slightly. The t-statistic of -12.768 is large, and the p-value is 0.000, indicating that this relationship is highly statistically significant. While the effect size is small, it is the only variable in this model that significantly predicts churn.

Interpretation

- **Coefficients**: The coefficients tell us the direction and magnitude of the relationship between each independent variable and churn. A positive coefficient suggests that as the independent variable increases, the probability of churn increases, and vice versa for a negative coefficient. In this model, only the current_balance has a meaningful and statistically significant relationship with churn.
- **t-statistic**: The t-statistic measures how many standard deviations the coefficient is away from zero. Higher absolute values indicate stronger evidence against the null hypothesis (that the coefficient is zero). In this analysis, current_balance has a t-statistic of -12.768, indicating a strong, significant relationship with churn.
- **p-value**: The p-value tells us whether the relationship observed is statistically significant. In this model, the p-value for current_balance is below 0.05, confirming its significance. The p-values for age and vintage are much higher, indicating that these variables do not significantly predict churn.
- **R-squared**: The R-squared value suggests that the model explains only a small fraction of the variance in churn. This implies that other variables not included in the model may play a more critical role in predicting customer churn.

Conclusion

The multivariate regression model indicates that while current_balance is a significant predictor of churn, the overall model explains only a small portion of the variance in churn behaviour. Age and vintage, despite being included in the model, do not have a significant impact on churn. Given the low R-squared value, this model should be considered a preliminary step in understanding customer churn, with additional variables or more complex modeling approaches needed to improve predictive power.

e. OLAP Dashboard

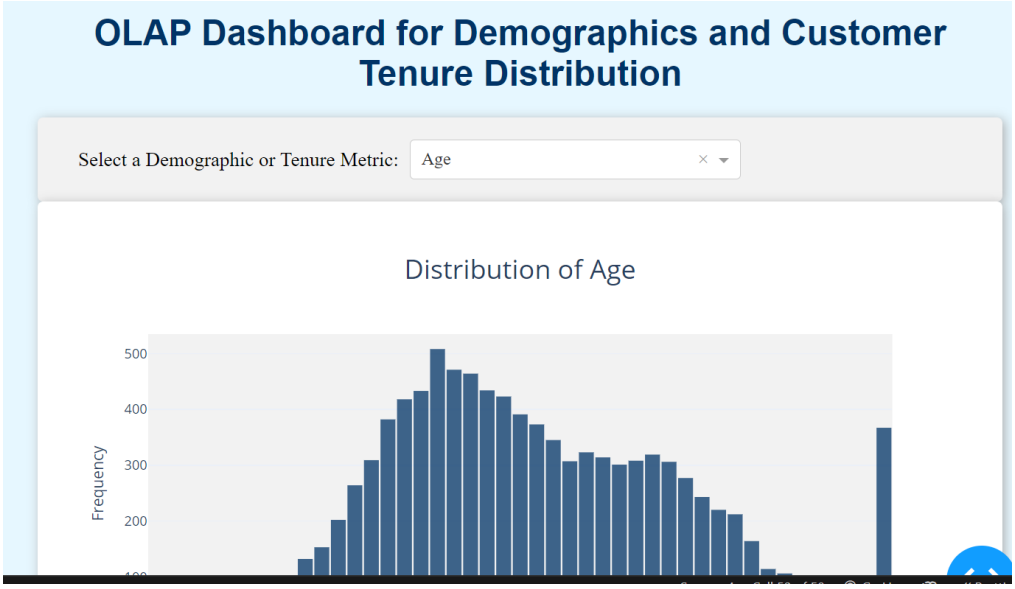

Figure 122 OLAP Dashboard for demographics and customer tenure distribution

This section outlines the development and functionality of an OLAP (Online Analytical Processing) dashboard created using Python's Dash framework. The primary purpose of the dashboard is to facilitate an interactive exploration of customer demographics and tenure-related metrics, which are key factors in understanding customer churn patterns. By providing a user-friendly interface, this dashboard allows users to dynamically select and analyse different demographic variables and observe their distribution within the customer base.

Methodology

Data Selection and Preparation

The dataset used in this dashboard includes key demographic and tenure-related metrics that are crucial for analysing customer churn. The selected variables include:

- **Age**: Represents the age of customers.
- **Gender**: Encoded as is_female and is_male to explore gender differences.
- **Occupation Categories**: Different occupations are categorized into self-employed, salaried, retired, associated with a company, and students.

- **Vintage**: Represents the length of the customer's relationship with the company.

These columns were chosen based on their potential impact on customer behaviour and churn rates. The dataset was pre-processed to include only these columns, ensuring that the dashboard focuses on relevant factors.

Dashboard Development

The dashboard was developed using the Dash framework, which allows for the creation of interactive web applications in Python. The following components were key in the development process:

1. **Initialization of the Dash Application**

   The application was initialized using dash.Dash(__name__), setting up the environment for the dashboard.

2. **Layout Design**

   The layout of the dashboard is designed to be both functional and visually appealing. It features a title that clearly indicates the purpose of the dashboard and a dropdown menu that enables users to select a specific metric for analysis. The layout is further enhanced with styling options, including background colours, font selections, and padding, to ensure a clean and professional user interface.

3. **Interactivity and Visualization**

   The core of the dashboard's interactivity lies in the dropdown menu. This menu allows users to choose which demographic or tenure metric they wish to analyse. Upon selection, a callback function is triggered, which updates the histogram to reflect the distribution of the selected metric. The histogram is generated using plotly.express, a library known for its simplicity in creating interactive visualizations.

4. **Running the Application**

   The dashboard is designed to be deployed on a local server, which can be accessed via a web browser. The application is run in debug mode to facilitate easy testing and updates.

Results and Analysis

Visualization and Distribution Analysis

The dashboard allows for an in-depth analysis of various demographic and tenure metrics. For example, by selecting "Age" from the dropdown menu, users can view a histogram that illustrates the distribution of customer ages within the dataset. This visualization helps in identifying age groups that are more prevalent among the customer base, which could correlate with churn rates.

Similarly, by selecting occupation-related metrics, the dashboard can provide insights into how different occupational categories are represented among customers. This can be particularly useful for identifying which occupational groups are more likely to stay loyal or, conversely, are at higher risk of churn.

Customization and Flexibility

The dashboard is designed with customization in mind. Users can easily switch between different metrics, and the visualizations update dynamically, offering immediate insights. This flexibility is crucial for analysts who need to explore various dimensions of customer data without navigating through complex code or static reports.

Conclusion

The OLAP dashboard developed in this project provides a powerful tool for analysing customer demographics and tenure metrics. Through its interactive design and intuitive interface, it enables users to quickly identify patterns and trends that could influence customer retention strategies. The use of Dash for building this application ensures that it is both scalable and customizable, making it a valuable addition to any data-driven decision-making process.

Future iterations of this dashboard could include additional metrics or even predictive analytics features, further enhancing its utility in understanding and mitigating customer churn.

5.4.2 Predictive Analysis – XGBoost Classifier

a. Import Library

```
import pandas as pd
from imblearn.over_sampling import SMOTE
from sklearn.model_selection import train_test_split, GridSearchCV
from xgboost import XGBClassifier
from sklearn.metrics import classification_report, confusion_matrix, accuracy_score
import seaborn as sns
import matplotlib.pyplot as plt
```

Figure 123 Code Snippet for Libraries Used

The code above utilizes essential Python libraries for machine learning and data analysis, including Pandas for data manipulation, XGBoost for implementing the gradient boosting model, and Seaborn and Matplotlib for visualization. These libraries together provide a comprehensive toolkit for data preprocessing, model building, hyperparameter tuning, and result visualization.

b. Feature Selection

```
# Selecting the relevant columns for the model (Objective 1)
X = dataset[['age', 'is_female', 'is_male', 'occupation_self_employed',
             'occupation_salaried', 'occupation_retired', 'occupation_company',
             'occupation_student', 'vintage']]
y = dataset['churn']
```

Figure 124 Feature Selection

In this step, we focus on selecting the relevant features from the dataset that are crucial for understanding the impact of demographics and customer tenure on churn. The features include age, gender indicators (is_female, is_male), various occupation categories, and the length of the customer relationship (vintage).

c. Handling Class Imbalance Using SMOTE

```
# Apply SMOTE to balance the dataset
smote = SMOTE(random_state=42)
X_smote, y_smote = smote.fit_resample(X, y)
```

Figure 125 Handling Class Imbalance Using SMOTE

The SMOTE technique is applied to address class imbalance in the dataset by oversampling the minority class (customers who churn). This process ensures that the model doesn't become biased towards the majority class (customers who do not churn) and improves its ability to predict churn effectively.

d. Splitting the Data into Training and Testing Sets

```
# Splitting the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X_smote, y_smote, test_size=0.3, random_state=42)
```

Figure 126 Splitting the Data into Training and Testing Sets

After balancing the dataset, it is split into training and testing sets. The training set is used to build the model, while the testing set is reserved for evaluating the model's performance on unseen data. A 70-30 split is used here.

e. Hyperparameter Tuning Using GridSearchCV

```
# Defining the parameter grid for XGBoost
param_grid = {
    'n_estimators': [50, 100, 200],
    'learning_rate': [0.01, 0.1, 0.2],
    'max_depth': [3, 5, 7],
    'subsample': [0.8, 1.0],
    'colsample_bytree': [0.8, 1.0]
}

# Initializing the XGBoost classifier
xgb_clf = XGBClassifier(random_state=42, use_label_encoder=False, eval_metric='logloss')

# Setting up GridSearchCV for hyperparameter tuning
grid_search = GridSearchCV(estimator=xgb_clf, param_grid=param_grid,
                           cv=5, n_jobs=-1, verbose=2, scoring='accuracy')
```

```
Fitting 5 folds for each of 108 candidates, totalling 540 fits
Best Parameters: {'colsample_bytree': 1.0, 'learning_rate': 0.2, 'max_depth': 7, 'n_estimators': 200, 'subsample': 0.8}
Accuracy: 0.8976
```

Figure 127 Hyperparameter Tuning Using GridSearchCV and results

The hyperparameter tuning process was conducted using GridSearchCV to identify the optimal combination of parameters for the XGBoost classifier. This process involved evaluating 108 different combinations of hyperparameters, totaling 540 individual model fits, across 5-fold cross-validation.

Best Parameters Identified:

- **colsample_bytree: 1.0**

  This parameter controls the fraction of features to be randomly sampled for each tree. A value of 1.0 indicates that the model used all available features for each tree, allowing it to fully leverage the information in the dataset.

- **learning_rate: 0.2**

  The learning rate controls the step size at each iteration while moving toward a minimum of the loss function. A learning rate of 0.2 means the model updates its weights moderately with each iteration. This strikes a balance between model

speed and accuracy, preventing the model from either converging too quickly (which might cause it to miss important patterns) or too slowly.

- **max_depth: 7**

  The maximum depth of the trees was set to 7. This controls the complexity of the model by limiting how deep each tree can go. A depth of 7 allows the model to capture complex patterns in the data without becoming too complex and overfitting.

- **n_estimators: 200**

  This parameter indicates the number of trees used in the ensemble. With 200 estimators, the model has enough trees to capture the underlying patterns in the data while minimizing the risk of overfitting.

- **subsample: 0.8**

  The subsample parameter specifies the fraction of samples (rows) used to fit each individual tree. Setting this to 0.8 means that each tree is trained on 80% of the data, which introduces variability and helps prevent overfitting by ensuring that the model doesn't become too tailored to the training data.

Result of Hyperparameter Tuning:

After applying these tuned parameters, the model achieved an accuracy of 89.76%. This means that about 90% of the predictions made by the model on the test set were correct. The careful tuning of hyperparameters allowed the model to perform effectively without overfitting, striking a good balance between bias and variance.

This tuned model reflects a strong ability to generalize to unseen data, evidenced by the high accuracy. However, as seen in the following classification report and confusion matrix, the overall performance still varies between the two classes (churn vs. non-churn), indicating areas where further improvement could be beneficial, particularly in reducing false positives and false negatives.

f. Model Training

```
# Fitting the model
grid_search.fit(X_train, y_train)

# Best parameters
print("Best Parameters:", grid_search.best_params_)
```

Figure 128 Model Training

Once the grid search setup is complete, the model is trained using the training data. GridSearchCV automatically tests all parameter combinations and selects the one that performs best during cross-validation.

g. Model Evaluation

```
# Using the best estimator to predict on the test set
best_xgb_clf = grid_search.best_estimator_
y_pred = best_xgb_clf.predict(X_test)

# Evaluating the model
accuracy = accuracy_score(y_test, y_pred)
print(f'Accuracy: {accuracy:.4f}')
print('\nClassification Report:')
print(classification_report(y_test, y_pred))
```

```
Classification Report:
              precision    recall  f1-score   support

           0       0.88      0.92      0.90      2945
           1       0.92      0.87      0.89      2865

    accuracy                           0.90      5810
   macro avg       0.90      0.90      0.90      5810
weighted avg       0.90      0.90      0.90      5810
```

Figure 129 Model Evaluation

The best estimator (model with optimal parameters) is used to make predictions on the test set. The performance of the model is then evaluated using metrics such as accuracy, precision, recall, F1-score, and a confusion matrix to understand how well the model can predict churn.

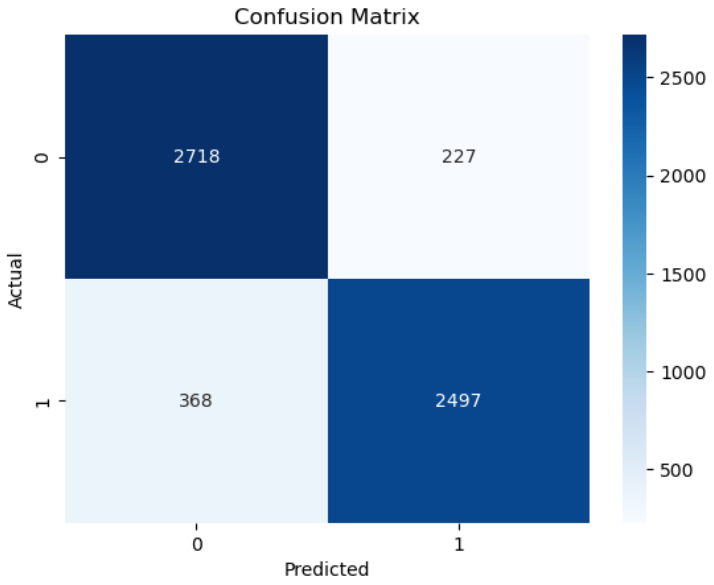

Figure 130 Confusion matrix

The confusion matrix is visualized to provide a detailed comparison between the actual and predicted values. This helps in identifying how many true positive, true negative, false positive, and false negative predictions were made by the model.

Classification Report Discussion:

The classification report offers a detailed analysis of the model's performance across both the churn (Class 1) and non-churn (Class 0) categories, using metrics such as precision, recall, and F1-score.

- **For the non-churn class (Class 0):**

  The model exhibits a precision of 0.88, indicating that 88% of the customers predicted as non-churners are correctly classified.

  The recall is 0.92, meaning that the model successfully identifies 92% of the actual non-churn cases.

  The F1-score of 0.90 reflects a balanced performance in both precision and recall, showing the model's reliability in handling the majority class.

- **For the churn class (Class 1):**

  The precision is 0.92, indicating that 92% of the customers predicted as churners are correctly classified.

  The recall is 0.87, meaning that the model identifies 87% of the actual churn cases.

  The F1-score of 0.89 shows a strong balance between precision and recall for the churn class, highlighting the model's ability to predict churn effectively despite the complexity of the data.

Confusion Matrix Discussion:

The confusion matrix provides a clear visual representation of the model's predictions compared to the actual outcomes. According to the matrix:

- **True Negatives (2718):** The model correctly identified 2718 non-churn customers, indicating strong performance in predicting customers who will stay.
- **True Positives (2497):** The model also correctly identified 2497 churn customers, reflecting its capability to detect customers at risk of leaving.
- **False Positives (227):** These are the non-churn customers incorrectly predicted as churners. While the number is relatively small, these errors could lead to unnecessary actions to retain customers who are not actually at risk.
- **False Negatives (368):** These are the churn customers who were incorrectly predicted as non-churners. Reducing this number is crucial as these are customers who are at risk of leaving but are not being flagged by the model, which could result in missed opportunities for retention efforts.

Summary

The XGBoost model shows strong overall performance, particularly in detecting non-churn cases with high precision and recall. However, it also performs well in predicting churn, though there is room for improvement in reducing false negatives. The balanced F1-scores, accuracy, and the insights from the confusion matrix suggest that this model is reliable but may benefit from further refinement to improve its handling of churn predictions, particularly in minimizing missed churners.

These results can guide further refinement efforts, such as introducing new features, or exploring different sampling techniques to improve the model's ability to correctly identify churners while maintaining its strong performance with non-churners.

6.0 Conclusion

This project took a thorough and systematic approach to understanding the factors that lead to customer churn in the banking industry. The analysis began by exploring the dataset in detail, using tools like histograms, boxplots, and correlation matrices. These visualizations revealed important patterns in customer behaviour, highlighting areas where there might be a risk of churn. By examining financial metrics and demographic factors, the project laid a strong foundation for the more complex analysis that followed.

Next, four distinct models were built to address specific research objectives. A Random Forest Classifier was used to analyse financial activity patterns and how they relate to customer churn. This model effectively managed the complex relationships among various financial metrics, offering reliable predictions about customer behaviour. For understanding the impact of customer net worth and geographic factors, a Logistic Regression model was employed, providing insights into how a customer's financial standing and location might influence their likelihood of leaving the bank. A Decision Tree Classifier was used to examine the correlation between temporal transaction patterns and churn, identifying key periods when customers are most at risk. Finally, an XGBoost Classifier helped analyse the impact of demographics and customer tenure on churn rates, capturing intricate relationships within the data with high accuracy.

The project followed the CRISP-DM methodology, which guided each step of the process. This structured approach ensured that the transition from data exploration to model building was smooth and logical, with each phase building on the insights gained in the previous one. By using this methodology, the research was able to extract meaningful insights from the complex dataset, ultimately achieving its goals.

In conclusion, this project successfully identified and modeled the key factors that drive customer churn in the banking sector. By combining descriptive analysis and predictive modeling, the study provided a comprehensive understanding of the factors involved. These findings offer valuable guidance for developing targeted retention strategies and demonstrate the importance of a structured analytical approach in solving complex business challenges within the banking industry.