PGP-DSBA PROJECT REPORT

CAPSTONE PROJECT NOTES - 1

CUSTOMER CHURN PREDICTION

BY

ISHAAN SHAKTI JAYARAMAN

PGPDSBA.O.JULY24.A

Contents

LIST OF FIGURES

INTRODUCTION OF THE BUSINESS PROBLEM

1.1 Problem Statement

In a highly competitive market, an E-Commerce company is facing increasing difficulty in retaining its existing accounts, where each account may represent multiple individual customers, thus amplifying the business impact of churn. To address this, the company aims to develop a predictive model to identify accounts at high risk of churn and implement a targeted, cost-effective retention campaign. The model must accurately forecast potential churners using historical account data and customer behavior, enabling the company to proactively engage them with segmented offers. These offers must strike a balance between being attractive enough to retain customers while ensuring financial sustainability, as overly generous incentives risk rejection by the revenue assurance team.

1.2 Need of the Study/Project

The need for this project arises from the critical business challenge of customer retention in a highly competitive E-Commerce environment, where losing a single account can lead to the loss of multiple customers and substantial revenue. A data-driven churn prediction model is essential to identify at-risk customers, allowing the company to intervene before churn occurs. Additionally, integrating intelligent segmentation and targeted campaign offers ensures that retention efforts are focused on the right accounts, using incentives that are both persuasive to the customer and financially viable for the company.

1.3 Understanding Business/Social Opportunity

This case study focuses on a DTH service provider where each customer is linked to a unique account ID, which can cover multiple individuals within a household, such as family plans that include various genders and marital statuses. Customers are given the flexibility to select their preferred payment method, and they are grouped into different service plans based on their usage patterns and the devices they use (like mobile or computer). Additionally, customers receive cashbacks on their payments. The company’s success relies heavily on customer loyalty and retention, which is maintained through quality services, value-added features, and ongoing promotions that attract both new users and help retain current ones. Retaining customers ensures a steady revenue stream, while new sign-ups bring in fresh income. However, losing customers—especially given that a single account ID may represent multiple users—can result in significant revenue loss. The growing need for DTH services in many households this poses both an opportunity for growth and increased competition for the company. The key challenge lies in distinguishing itself from rivals by understanding and addressing the factors that drive customer loyalty and satisfaction, as these will ultimately determine the company's position in the market.

DATA REPORT

2.1 Data Dictionary

AccountID: Unique identifier for the account

Churn: Account churn flag (Target variable)

Tenure: Duration of account existence

City_Tier: Tier classification of the primary customer's city

CC_Contacted_L12m: Number of times customers in the account have contacted customer care in the last 12 months

Payment: Preferred payment method of customers in the account

Gender: Gender of the primary customer of the account

Service_Score: Satisfaction score given by account customers for the services provided by the company

Account_user_count: Total number of customers associated with the account

account_segment: Segmentation of the account based on spending patterns

CC_Agent_Score: Satisfaction score for customer care services given by account customers

Marital_Status: Marital status of the primary customer of the account

rev_per_month: Average monthly revenue generated by the account in the past 12 months

Complain_l12m: Whether any complaints have been raised by the account in the last 12 months

rev_growth_yoy: Year-on-year revenue growth percentage (comparison between the last 12 months and the prior 12 months)

coupon_used_l12m: Number of times customers used coupons for payments in the last 12 months

Day_Since_CC_connect: Number of days since no customer in the account has contacted customer care

cashback_l12m: Average monthly cashback earned by the account in the past 12 months

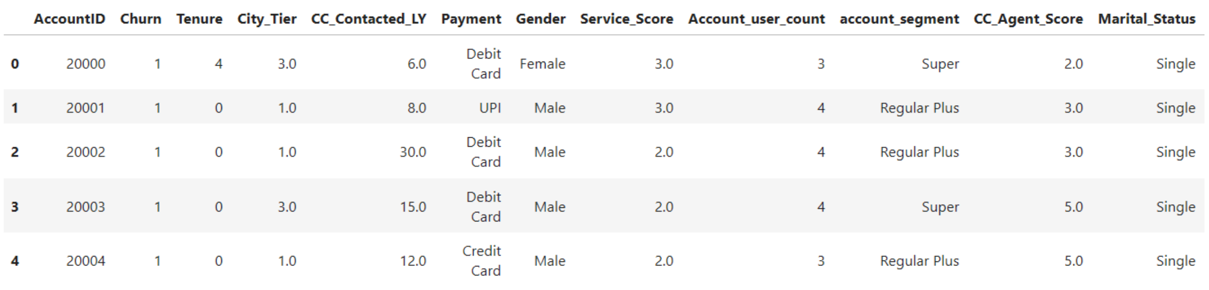
Login_device: Preferred login device used by customers in the account

2.2 Understanding Data Collection

- Data represents 11,260 unique account IDs, covering a wide range of customer demographics such as gender and marital status
- Key variables like "CC_Contacted_L12m", "rev_per_month", "Complain_l12m", "rev_growth_yoy", "coupon_used_l12m", "Day_Since_CC_connect", and "cashback_l12m" implies a 12-month period for data collection
- The dataset consists of 19 variables, with 18 independent and 1 dependent variable, which tracks customer churn (yes or no)
- It provides insights into the services used by customers, their payment options, and basic personal information
- The dataset includes both categorical and continuous variables

2.3 Visual Inspection of the Data

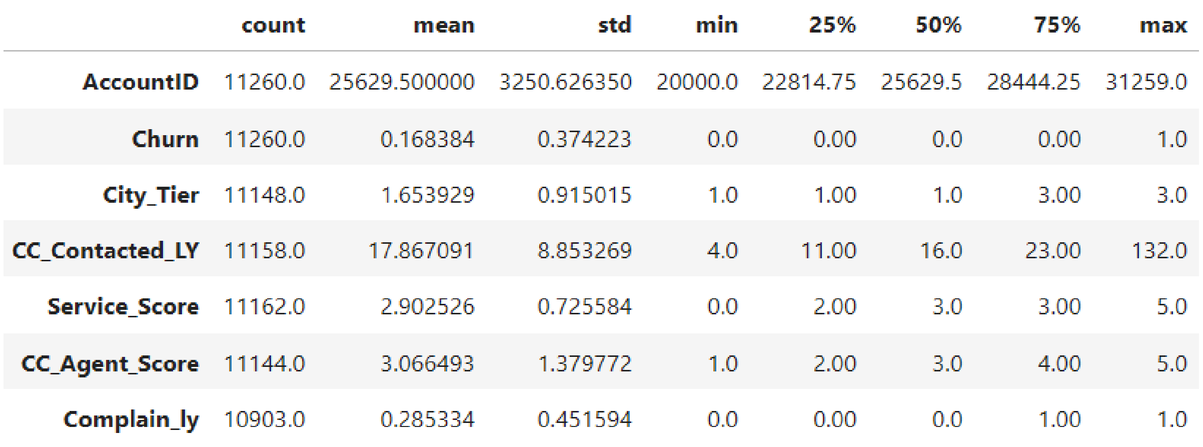
Figure 1 - Snippet of the first 5 rows and 12 columns



	AccountID	Churn	Tenure	City_Tier	CC_Contacted_LY	Payment	Gender	Service_Score	Account_user_count	account_segment	CC_Agent_Score	Marital_Status
0	20000	1	4	3.0	6.0	Debit Card	Female	3.0	3	Super	2.0	Single
1	20001	1	0	1.0	8.0	UPI	Male	3.0	4	Regular Plus	3.0	Single
2	20002	1	0	1.0	30.0	Debit Card	Male	2.0	4	Regular Plus	3.0	Single
3	20003	1	0	3.0	15.0	Debit Card	Male	2.0	4	Super	5.0	Single
4	20004	1	0	1.0	12.0	Credit Card	Male	2.0	3	Regular Plus	5.0	Single

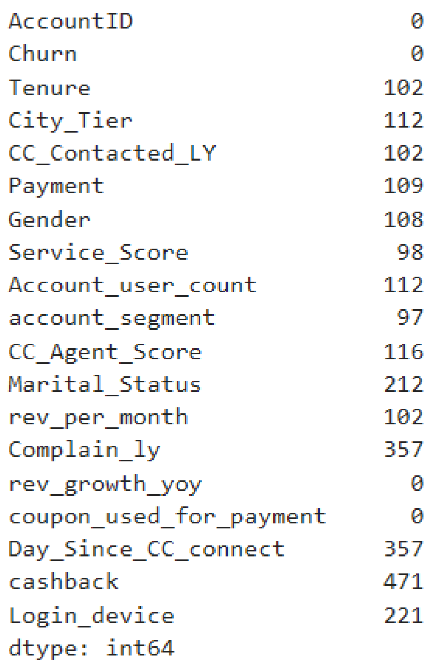
The dataset has a total of 11,260 observations and 19 variables. 5 variables are float types, 2 variables are integer types and the rest are object type variables.

Figure 2 - Statistical Summary of the Dataset



	count	mean	std	min	25%	50%	75%	max
AccountID	11260.0	25629.500000	3250.626350	20000.0	22814.75	25629.5	28444.25	31259.0
Churn	11260.0	0.168384	0.374223	0.0	0.00	0.0	0.00	1.0
City_Tier	11148.0	1.653929	0.915015	1.0	1.00	1.0	3.00	3.0
CC_Contacted_LY	11158.0	17.867091	8.853269	4.0	11.00	16.0	23.00	132.0
Service_Score	11162.0	2.902526	0.725584	0.0	2.00	3.0	3.00	5.0
CC_Agent_Score	11144.0	3.066493	1.379772	1.0	2.00	3.0	4.00	5.0
Complain_ly	10903.0	0.285334	0.451594	0.0	0.00	0.0	1.00	1.0

Figure 3 - Number of Missing Values in the Dataset



```
AccountID                  0
Churn                      0
Tenure                   102
City_Tier                112
CC_Contacted_LY          102
Payment                  109
Gender                   108
Service_Score             98
Account_user_count       112
account_segment           97
CC_Agent_Score           116
Marital_Status           212
rev_per_month            102
Complain_ly              357
rev_growth_yoy             0
coupon_used_for_payment    0
Day_Since_CC_connect     357
cashback                 471
Login_device             221
dtype: int64
```

There are a large number of null values present in the variables except “AccountID”, “Churn”, “rev_growth_yoy” and “coupon_used_for_payment”. The null values will be treated post EDA.

There are no Duplicate values present in the dataset.

2.4 Data Attributes

- AccountID: A unique identifier for each customer, stored as an integer. No missing values present.
- Churn: The target variable indicating whether the customer has churned. It is categorical with no null values. "0" means "No" and "1" means "Yes".

- Tenure: The total duration of an account since it was created. It is a continuous variable with 102 missing values.
- City_Tier: Categorizes the primary customer's city into three tiers based on city size. It is a categorical variable with 112 missing values.
- CC_Contacted_L12m: Indicates the number of times customers from the account have contacted customer care in the last 12 months. This is a continuous variable with 102 missing values.
- Payment: The preferred method of payment chosen by customers for their bills. It is a categorical variable with 109 missing values.
- Gender: Represents the gender of the primary account holder. It is categorical, with 108 missing values.
- Service_Score: Satisfaction scores given by customers based on the service provided by the company. It is categorical with 98 missing values.
- Account_user_count: The number of customers associated with a single account ID. It is a continuous variable with 112 missing values.
- Account_segment: Segments customers based on their spending and revenue generation. This is a categorical variable with 97 missing values.
- CC_Agent_Score: Customer ratings for the service provided by customer care representatives. It is categorical with 116 missing values.
- Marital_Status: Represents the marital status of the primary account holder. It is a categorical variable with 212 missing values.
- Rev_per_month: The average revenue generated by the account per month over the last 12 months. This is a continuous variable with 102 missing values.
- Complain_l12m: Indicates whether any complaints were raised by the account in the past 12 months. This is a categorical variable with 357 missing values.
- Rev_growth_yoy: The year-over-year revenue growth percentage of the account, comparing the last 12 months with the previous 12 to 24 months. This is a continuous variable with no missing values.
- Coupon_used_l12m: The number of times customers have used discount coupons for payment in the last 12 months. This is a continuous variable with no missing values.
- Day_Since_CC_connect: The number of days since the account's customers last contacted customer care. A higher number indicates better service. This is a continuous variable with 357 missing values.

- Cashback_l12m: The average cashback earned by customers through bill payments over the past 12 months. This is a continuous variable with 471 missing values.
- Login_device: The type of device used by customers to access the services, whether it's a phone or a computer. It is a categorical variable with 221 missing values.

EXPLORATORY DATA ANALYSIS

3.1 Univariate Analysis

Figure 4 - Countplot of Churned Customers

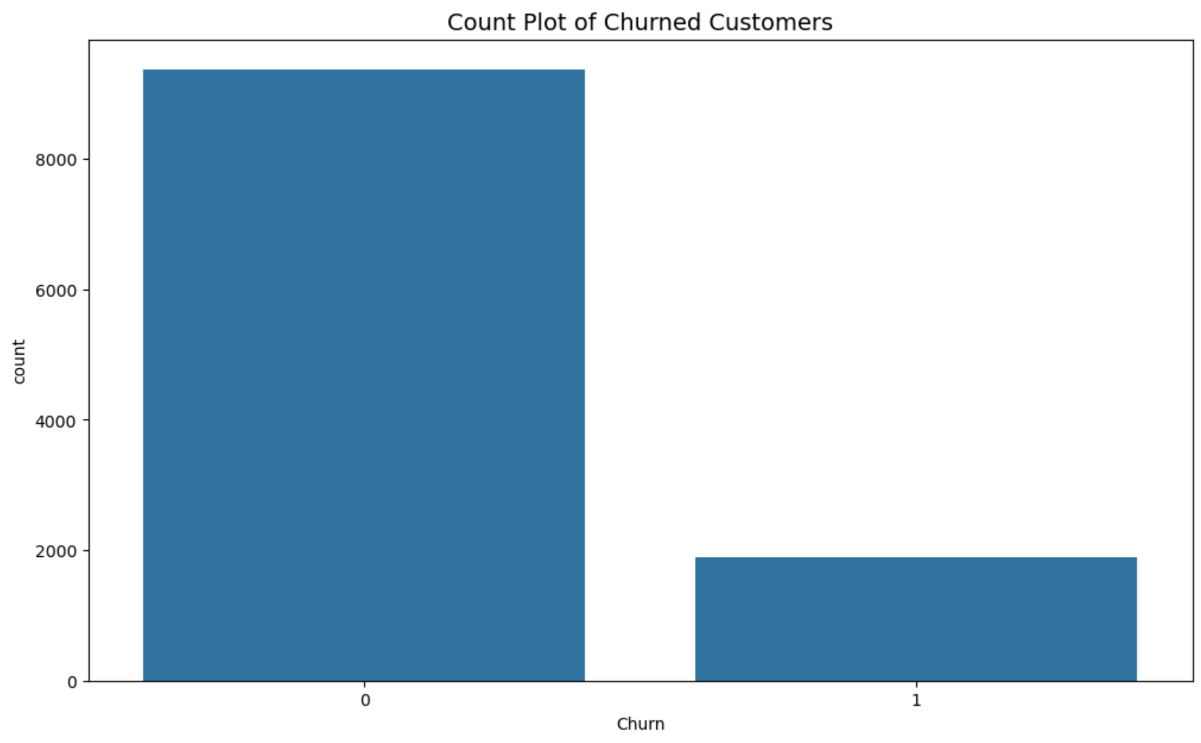


Figure 5 - Countplot of City Tier

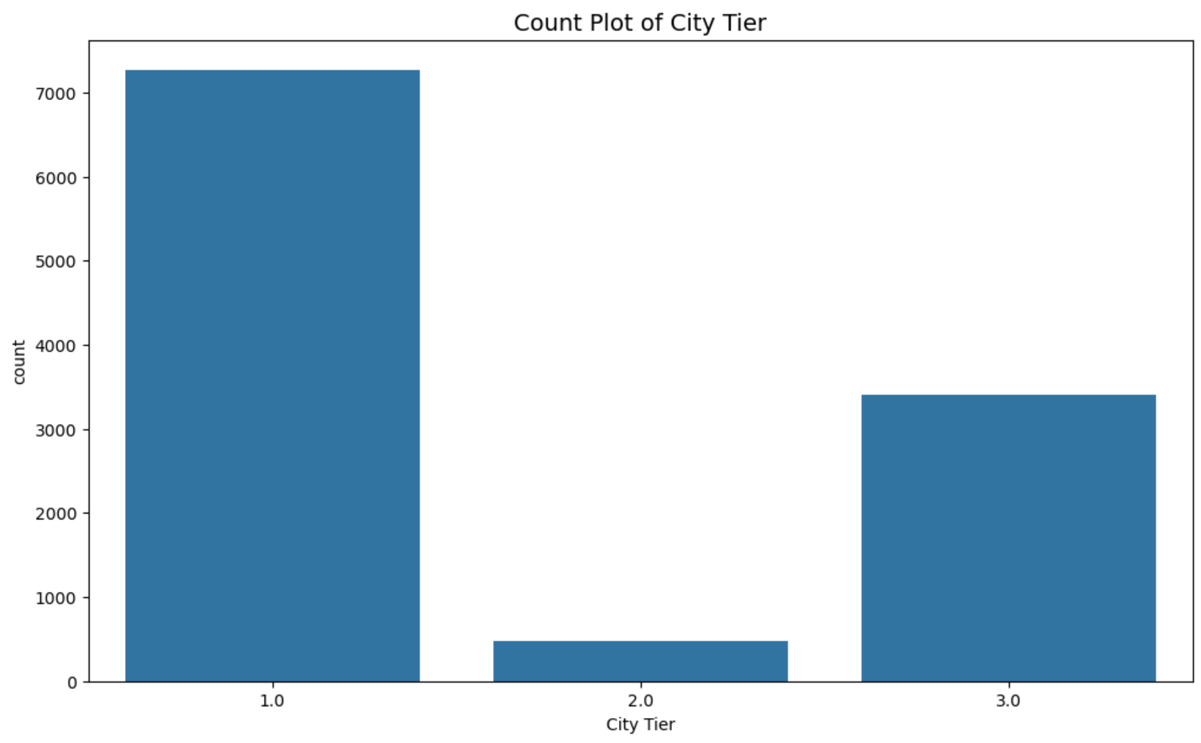


Figure 6 - Countplot of Payment Method

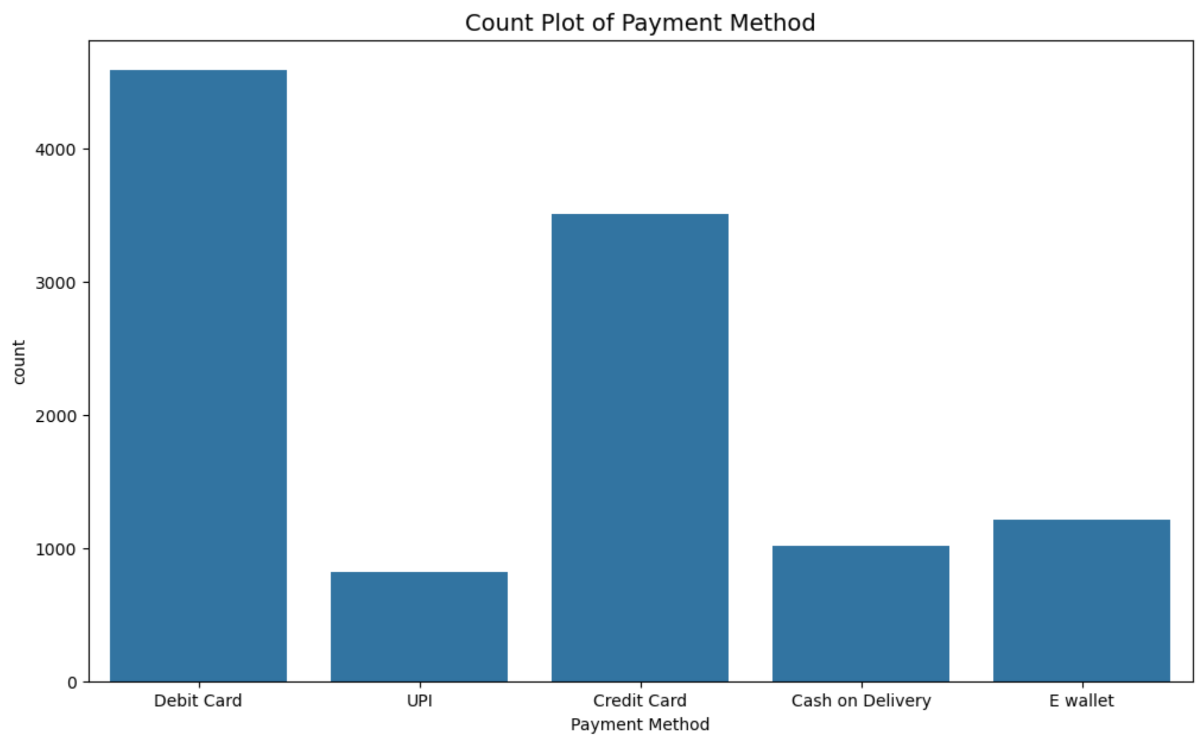


Figure 7 - Countplot of Gender

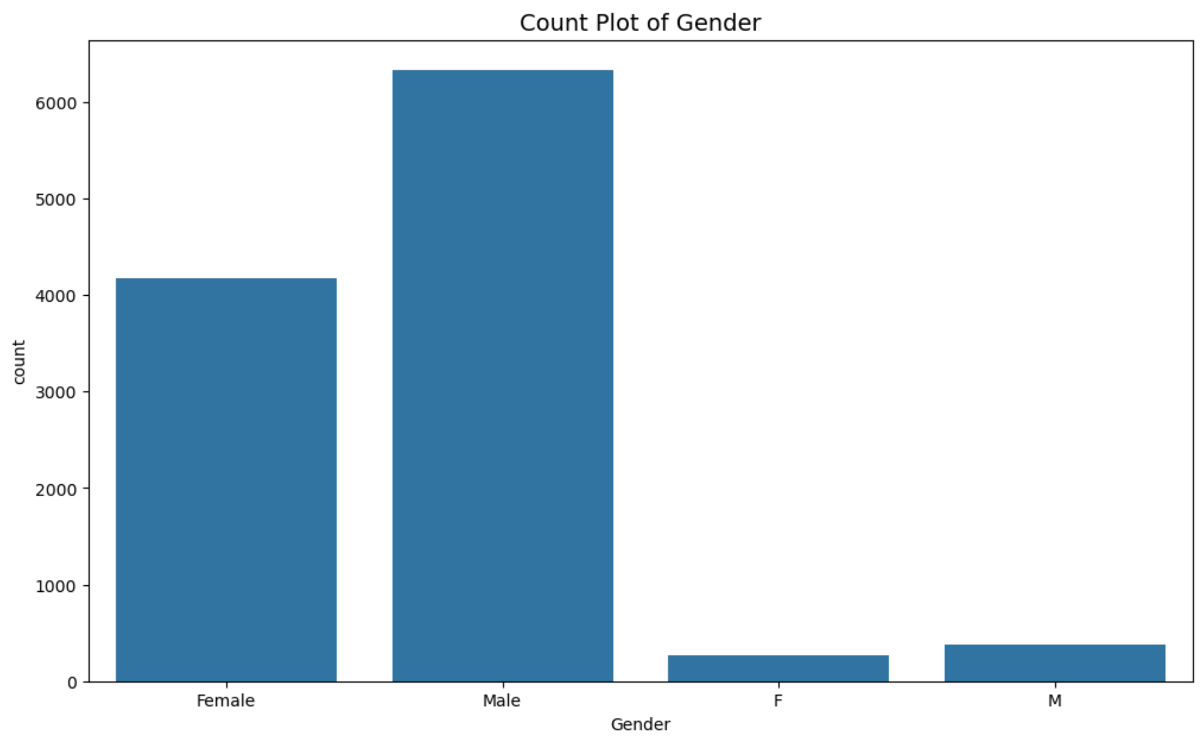


Figure 8 - Countplot of Service Score

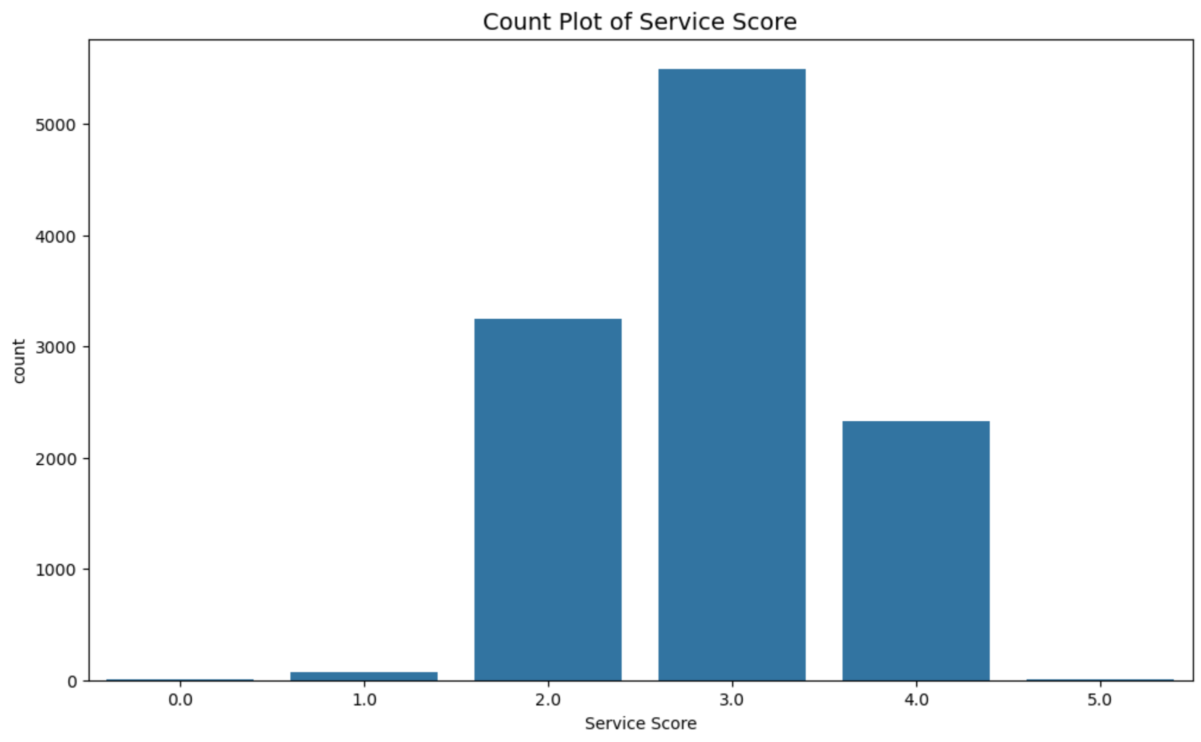


Figure 9 - Countplot of Account Users

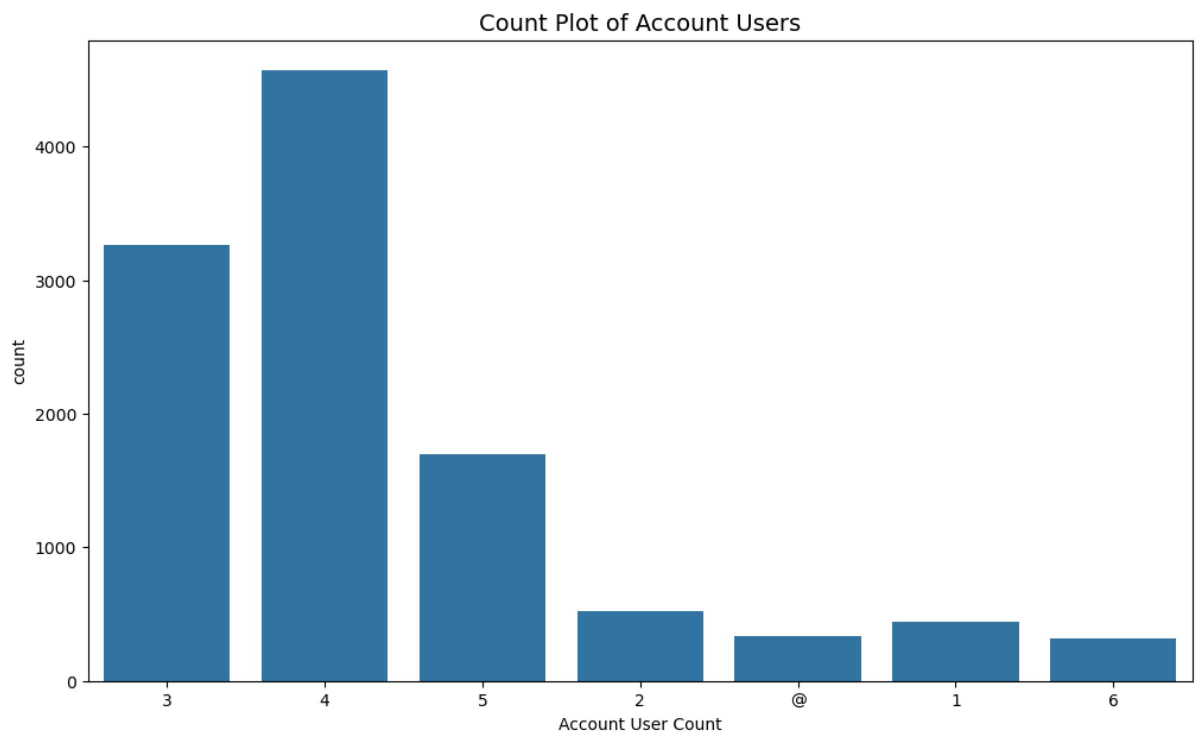


Figure 10 - Countplot of Account Segment

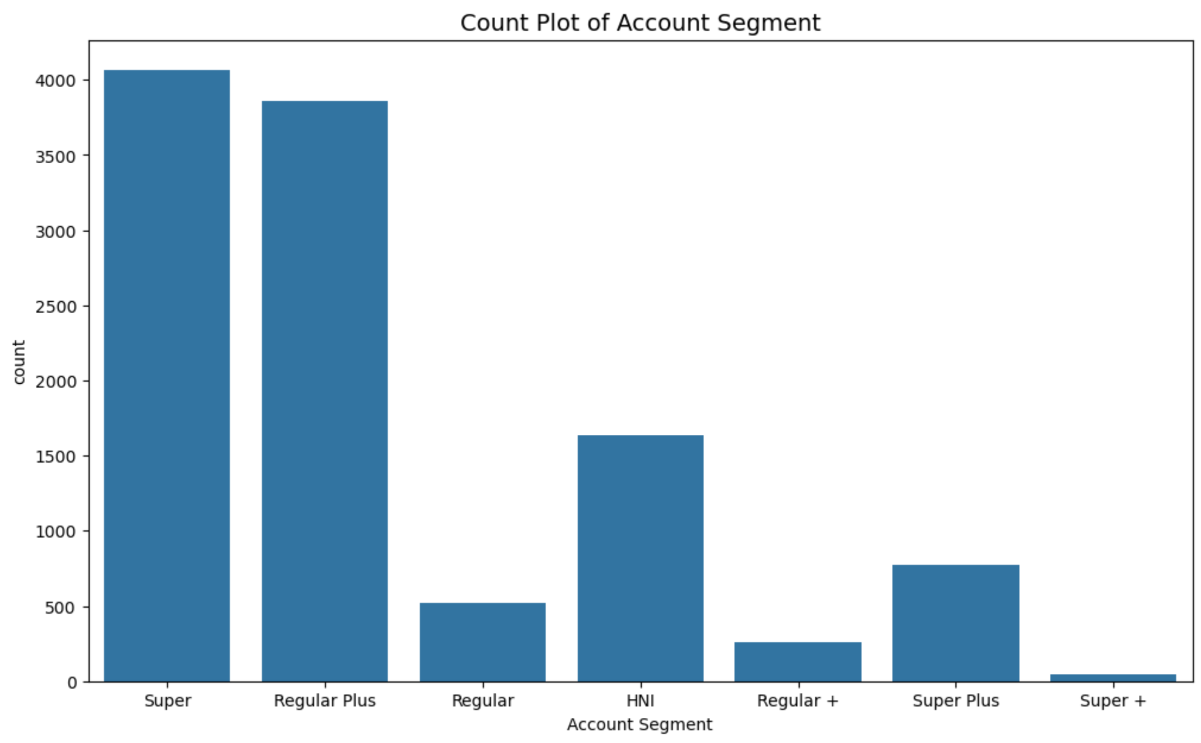


Figure 11 - Countplot of CC Agent Score

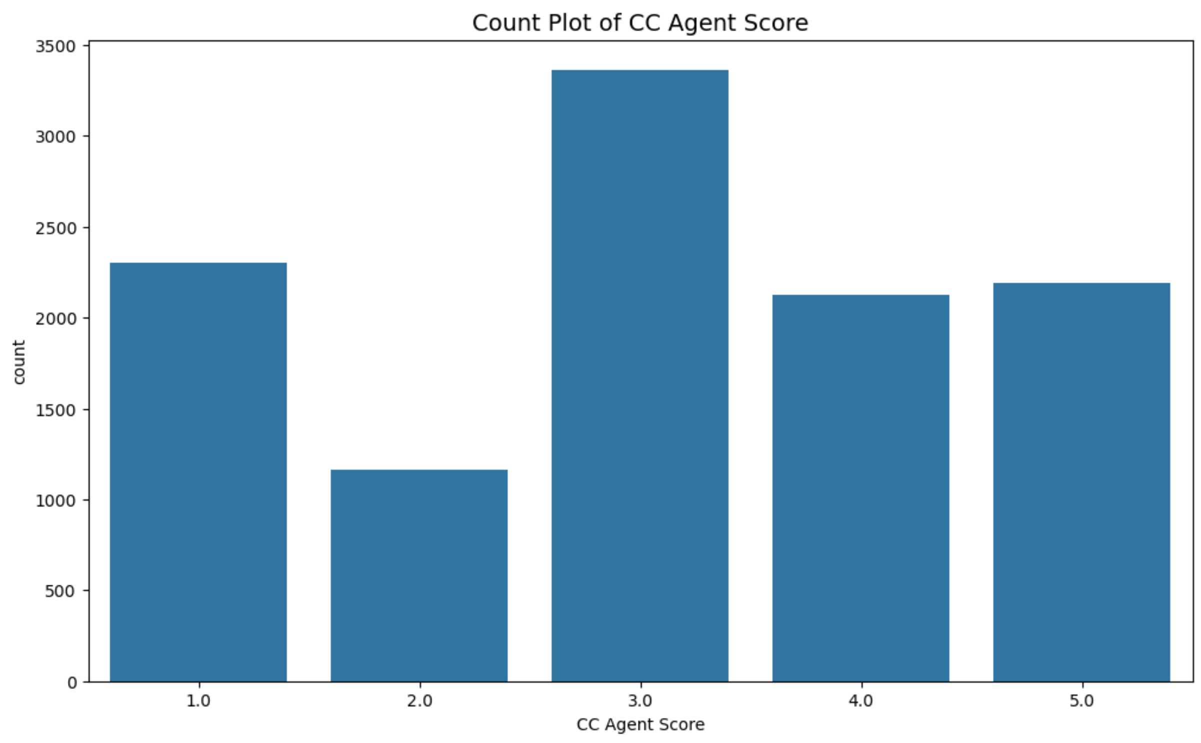


Figure 12 - Countplot of Marital Status

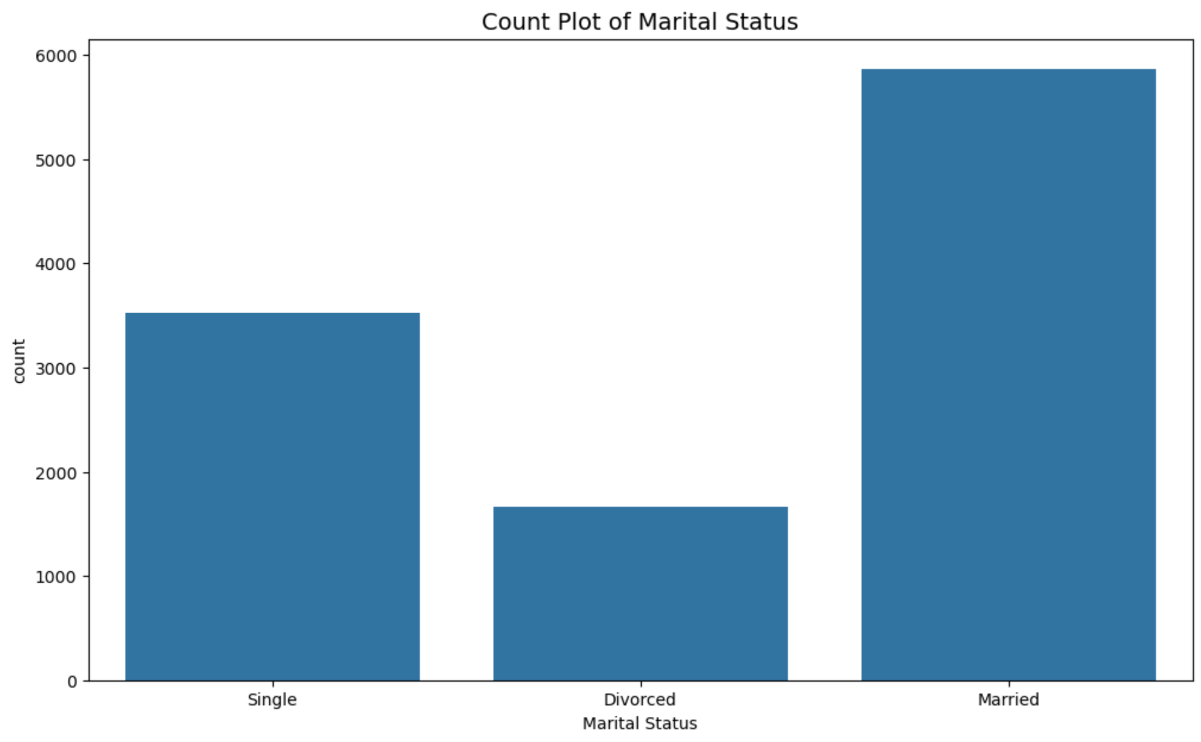


Figure 13 - Countplot of Complains in the Last 12 Months

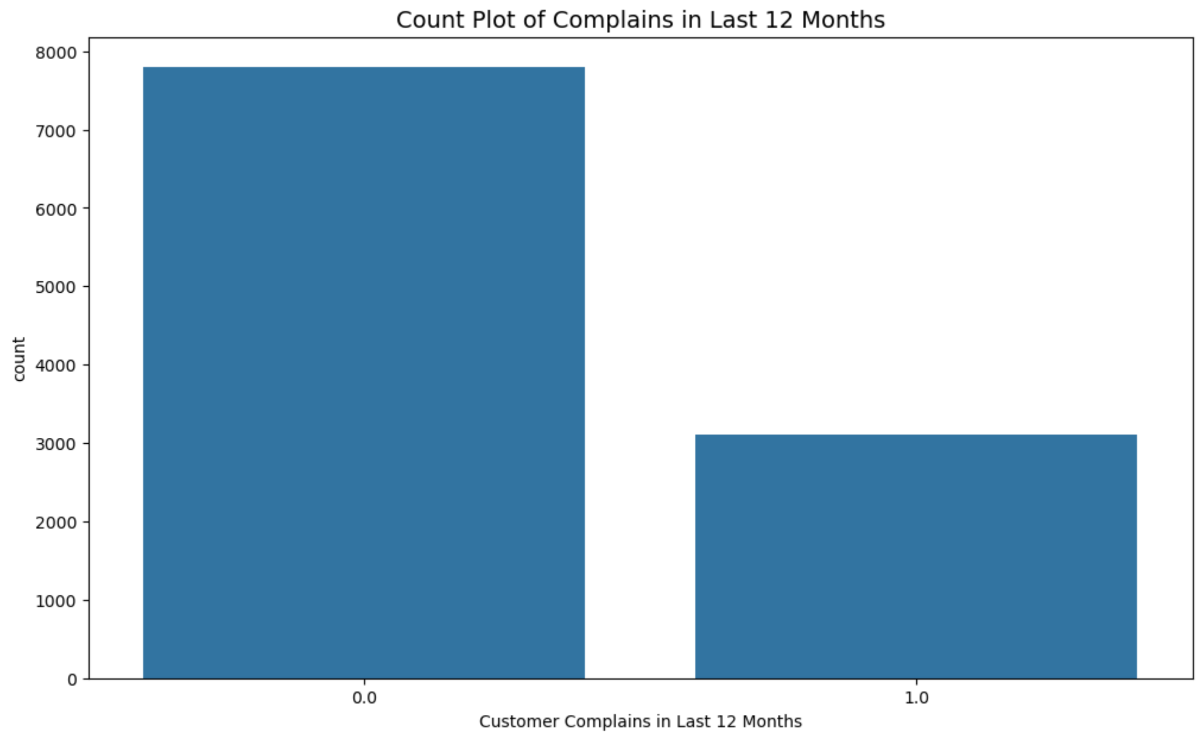
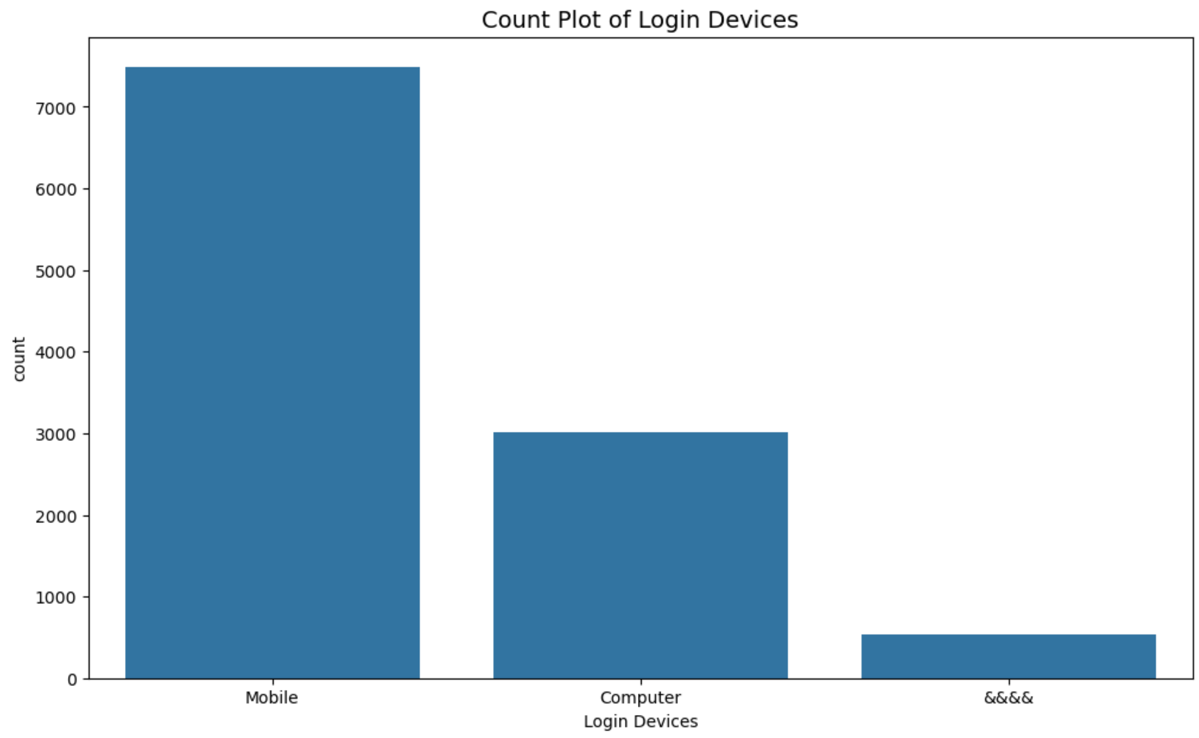


Figure 14 - Countplot of Login Devices



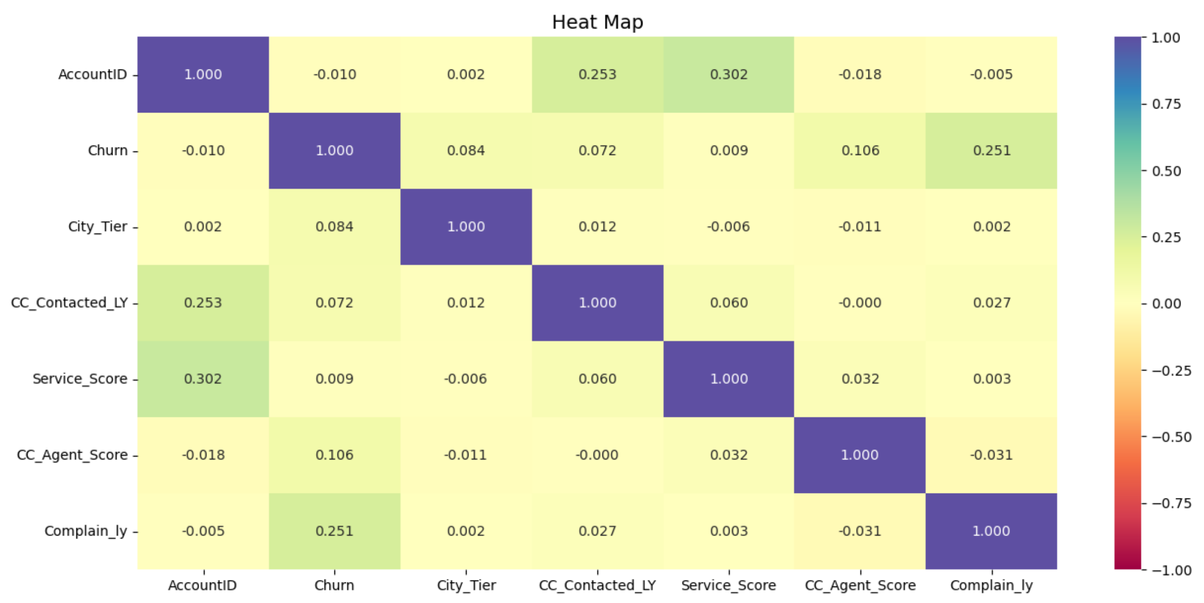
Observations of all Variables:

- There are over 8000 customer accounts who have not churned while there are less than 2000 customer accounts who have churned.
- A large proportion of customers come from city tier 1, indicating a high population density in this area.
- Debit and Credit Cards are the most popular payment methods among customers.
- There are more male customers compared to female customers.
- Customers rate the service score around 3 which can be considered decent but 4-5 rating is always the target.
- The highest number of users in a single account is 4 users followed by 3. 6 total users is the lowest.
- The majority of customers belong to the "Super+" segment, with the "Regular" segment having the fewest customers.
- There have been around 3000 complains made by an account in the last 12 months.
- Most customers who use the services are married.
- Mobile devices are the preferred choice for customers when accessing services.

3.2 Bivariate Analysis

3.2.1 Heatmap

Figure 15 - Heatmap of all Numerical Variables



Observations of Heatmap:

- From the heatmap we can see that maximum correlation in the dataset is between Churn and Complain_ly which shows that customers who made a complaint in the last 12 months had a higher chance of churning.
- Correlation between service score and AccountID can be ignored as AccountID is completely unique values.
- There is little correlation among the other variables.

3.2.2 All Variables VS Churn

Figure 16 - Countplot of City Tier vs Churned Customers

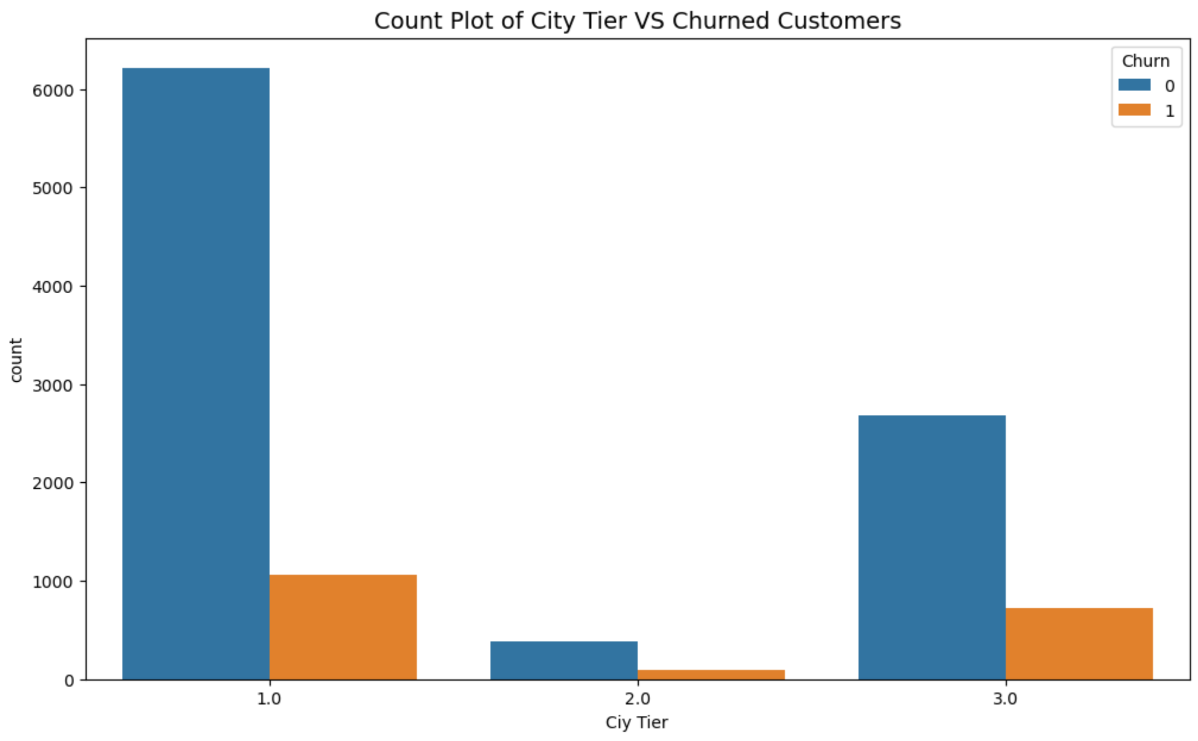


Figure 17 - Countplot of Payment Method vs Churned Customers

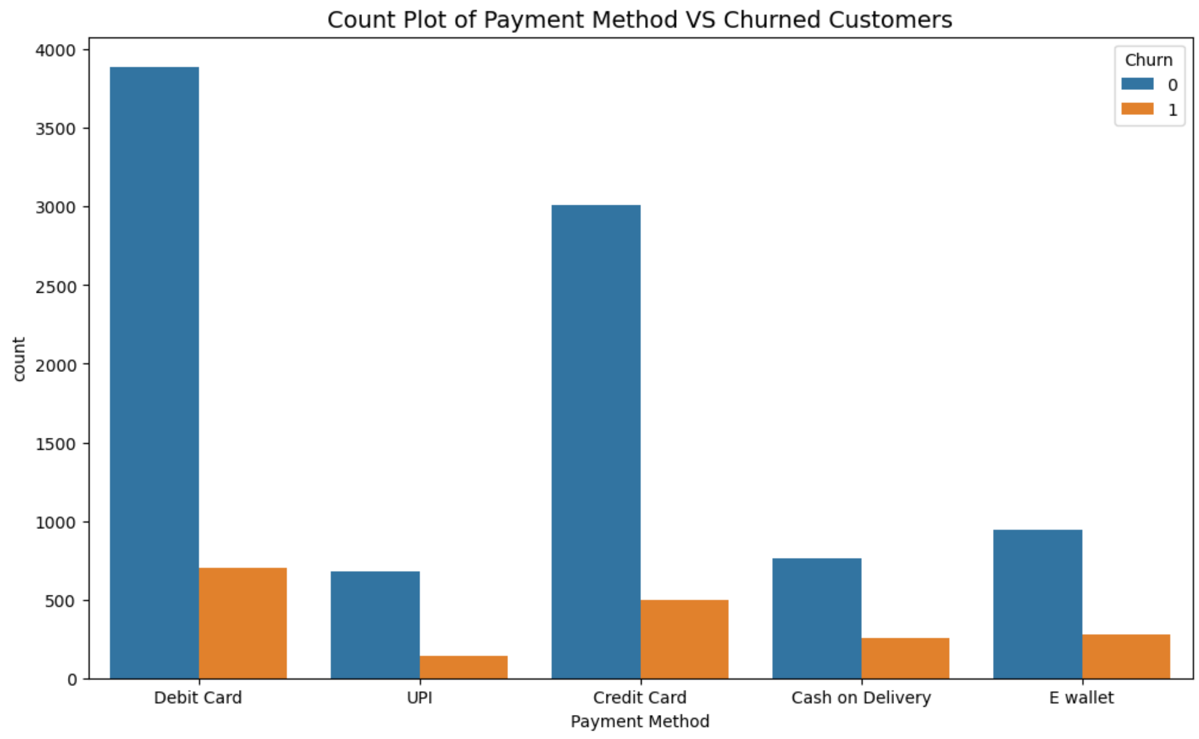


Figure 18 - Countplot of Gender Vs Churned Customers

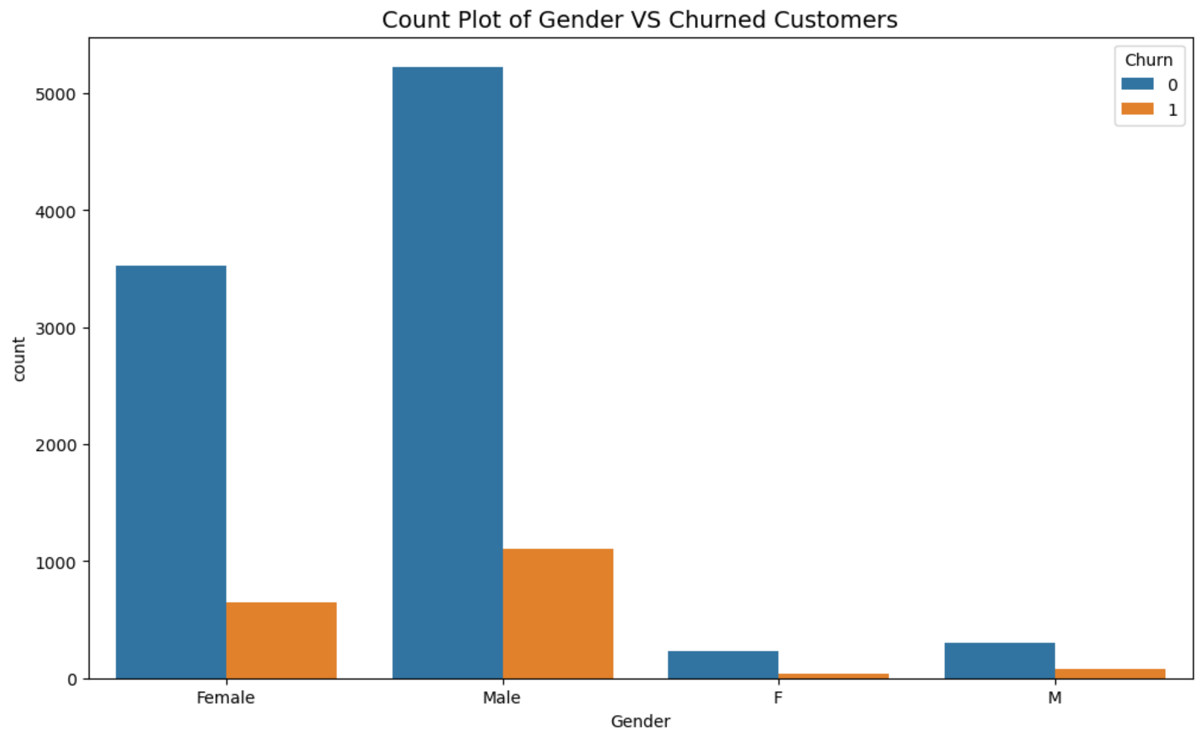


Figure 19 - Countplot of Service Score vs Churned Customers

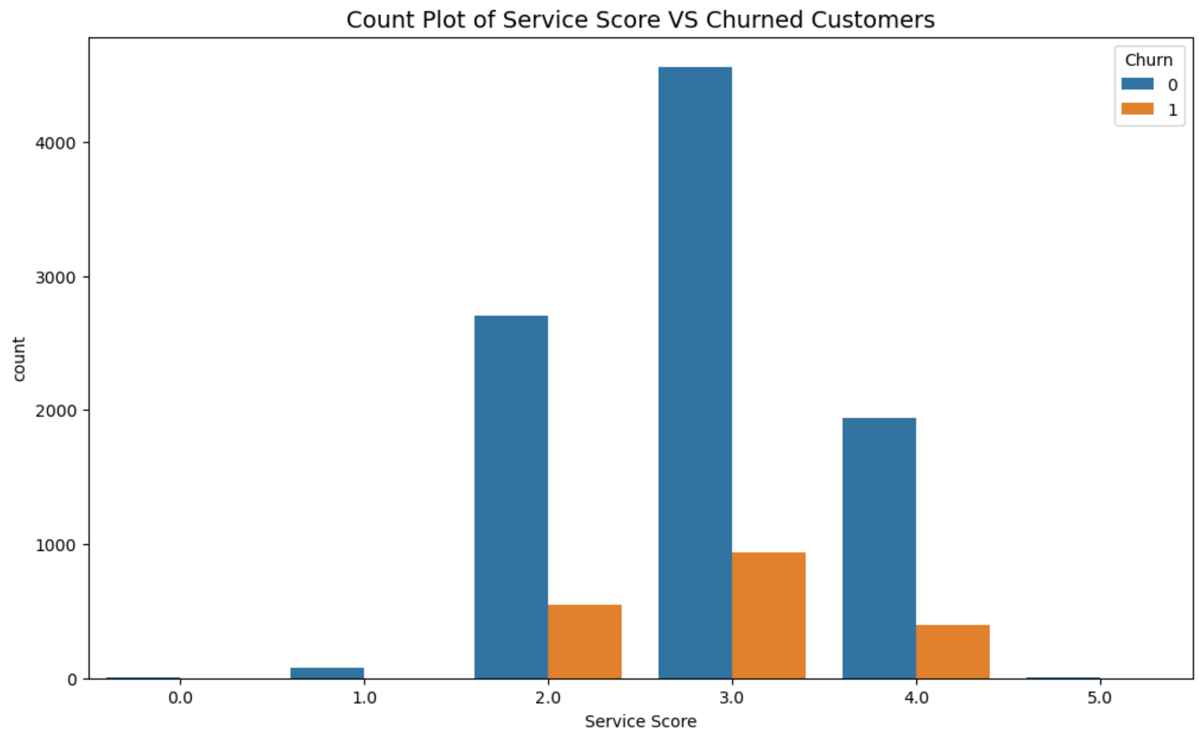


Figure 20 - Countplot of Account User Count vs Churned Customers

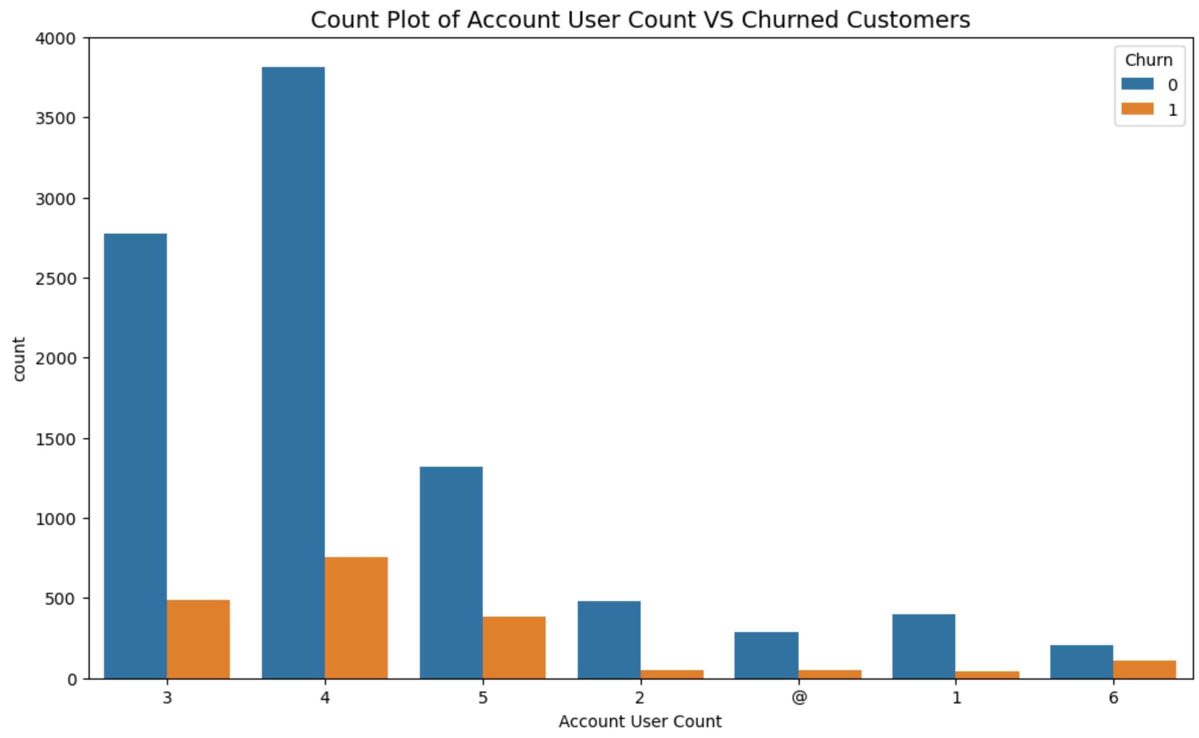


Figure 21 - Countplot of Account Segment vs Churned Customers

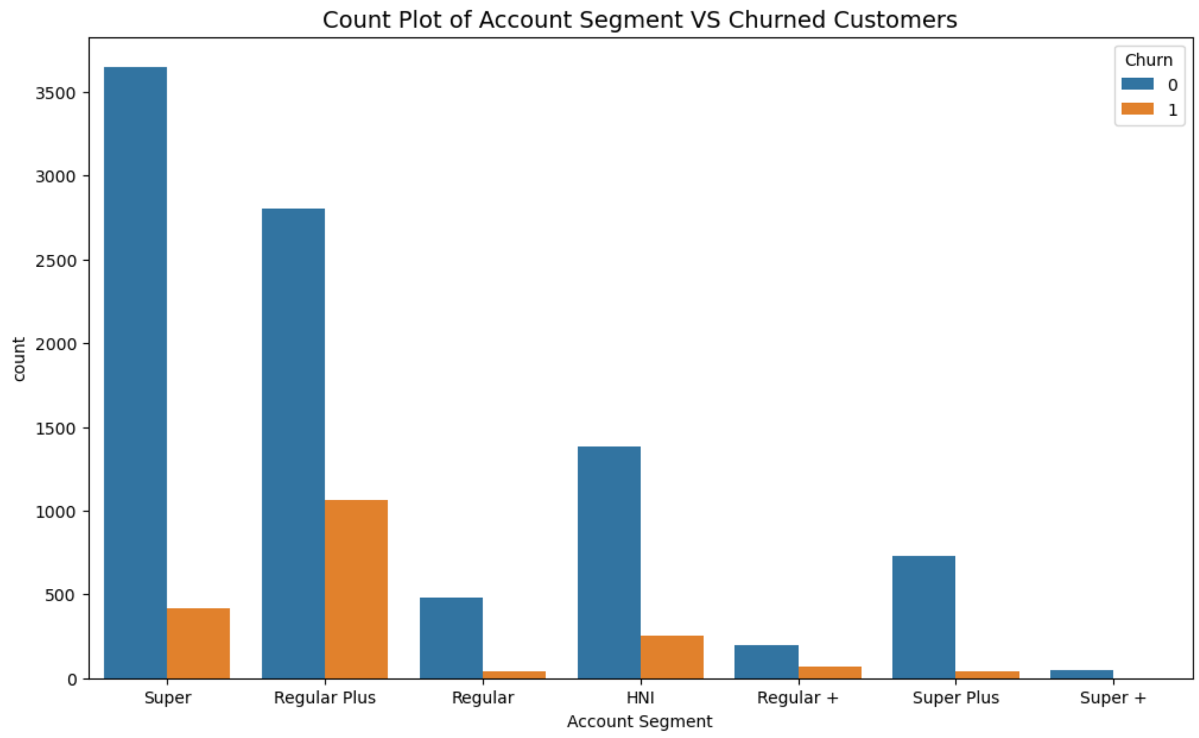


Figure 22 - Countplot of CC Agent Score vs Churned Customers

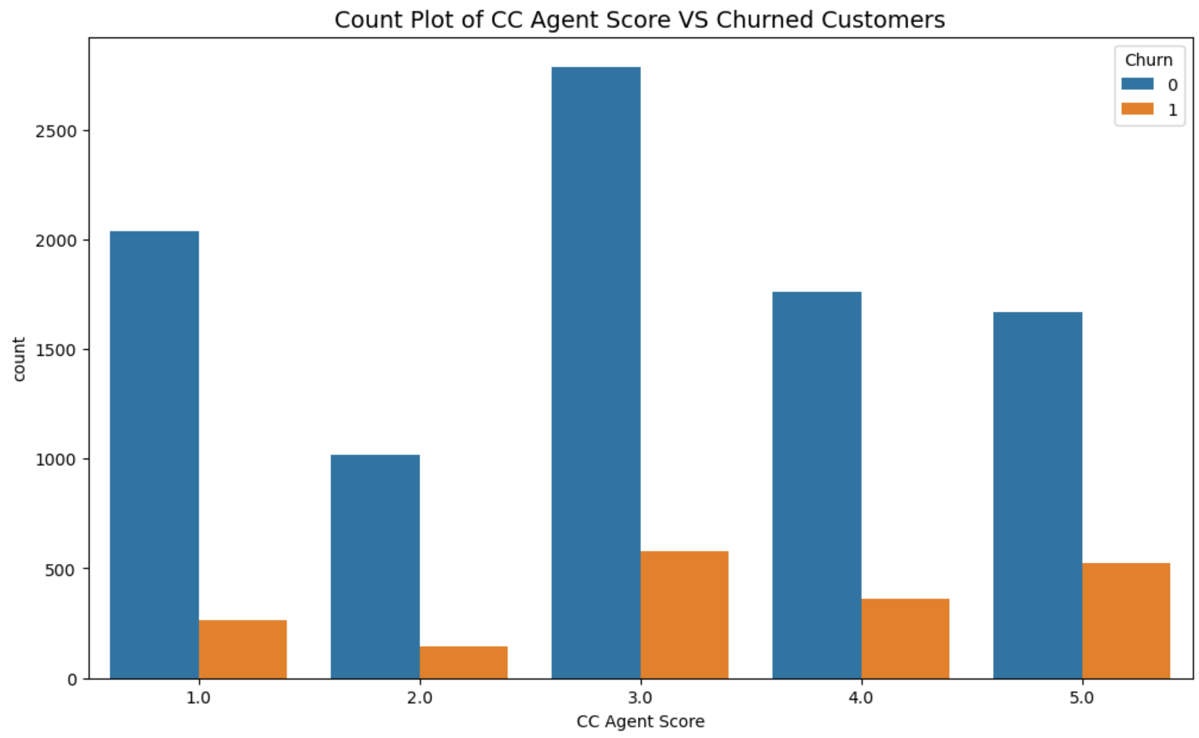


Figure 23 - Countplot of Marital Status vs Churned Customers

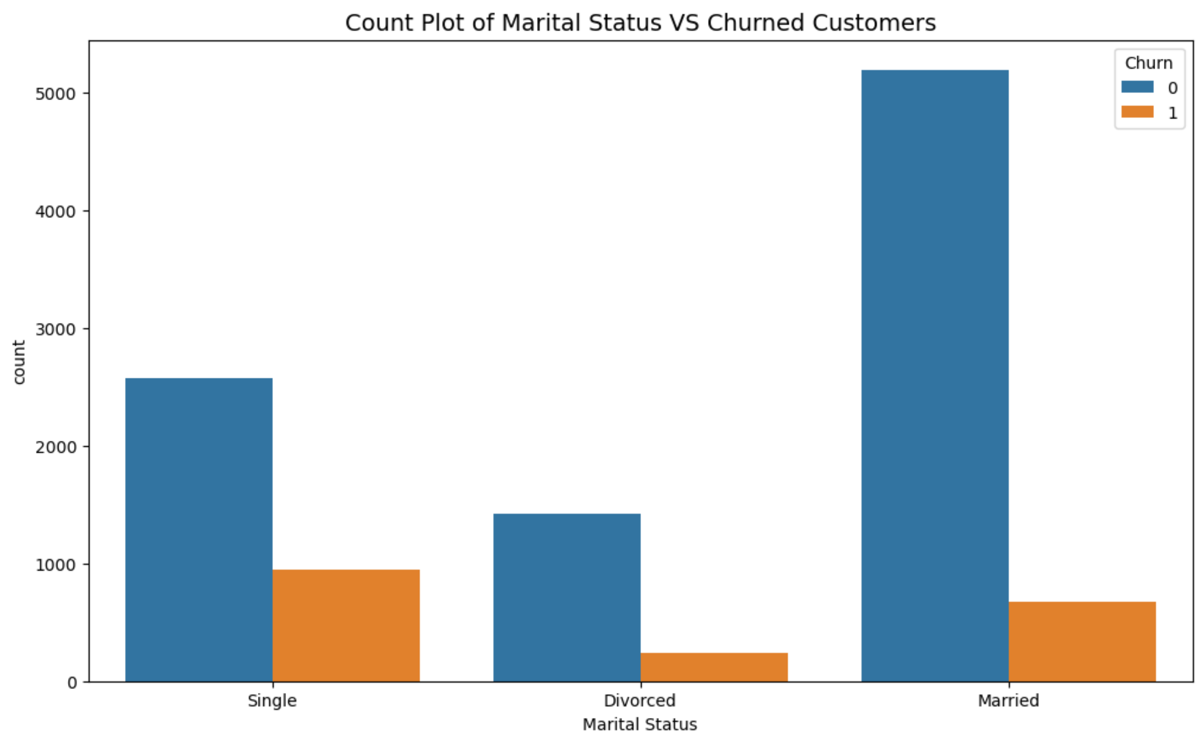


Figure 24 - Countplot of Complains in Last Year vs Churned Customers

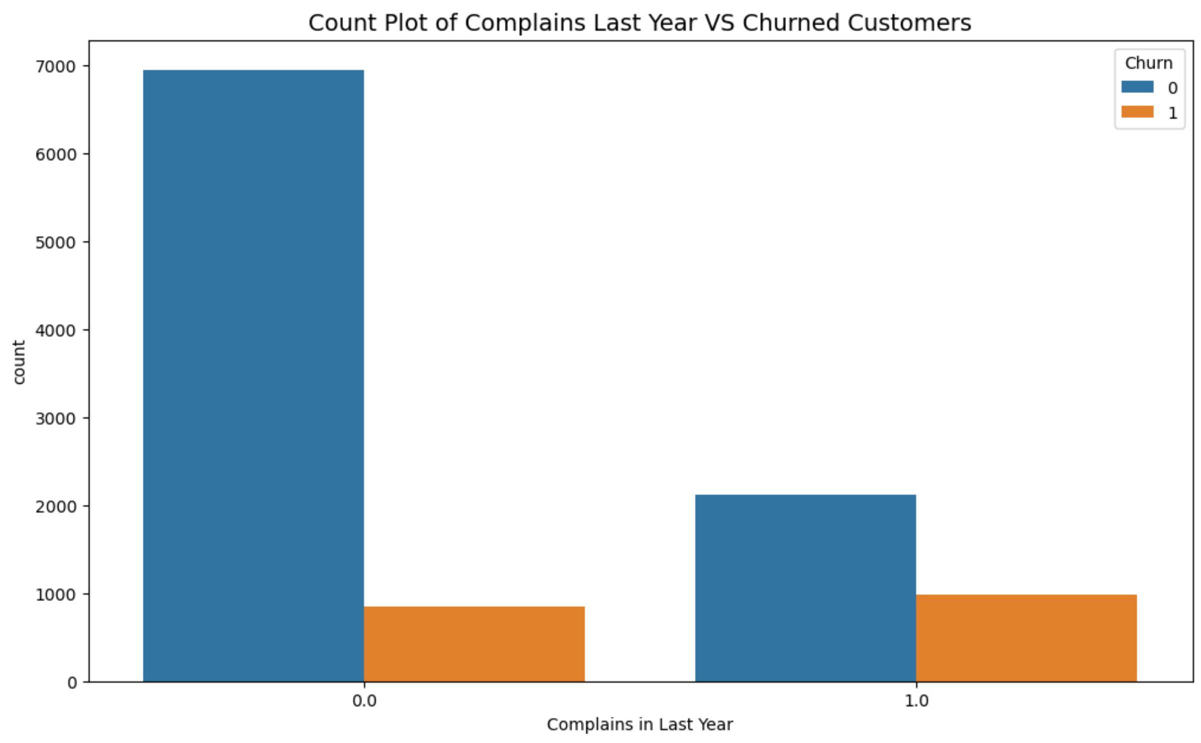
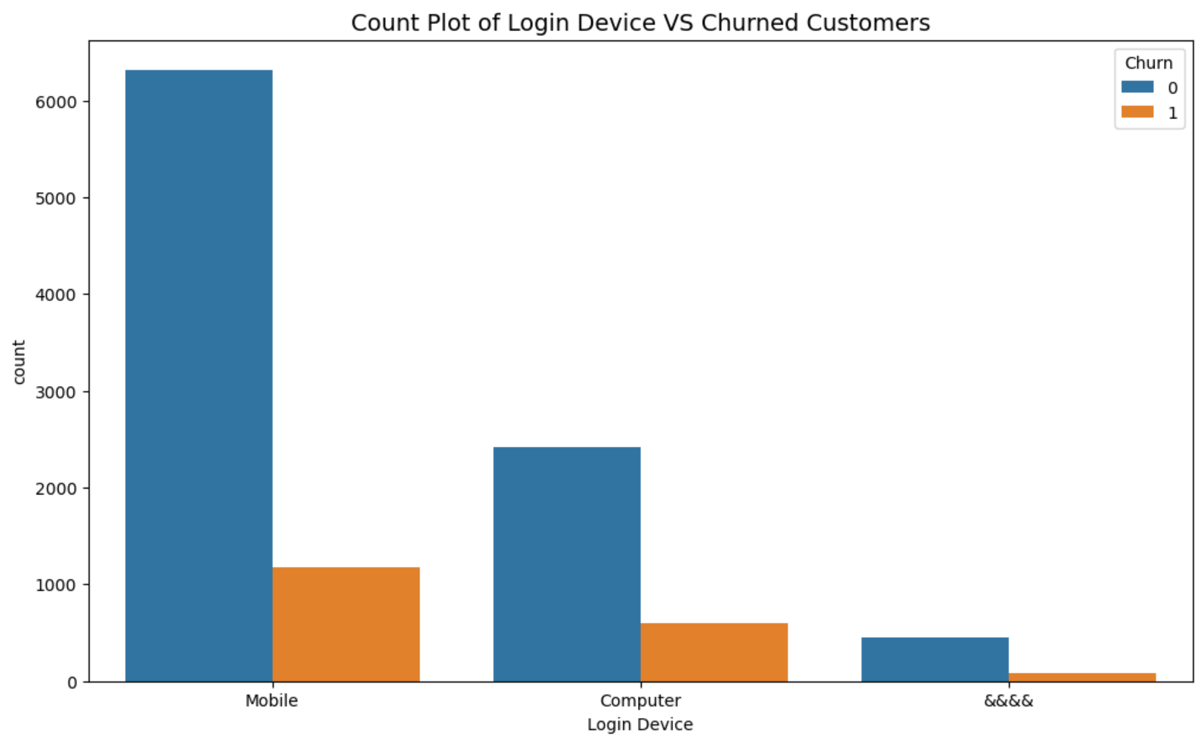


Figure 25 - Countplot of Login Devices vs Churned Customers



Observations of all Variables:

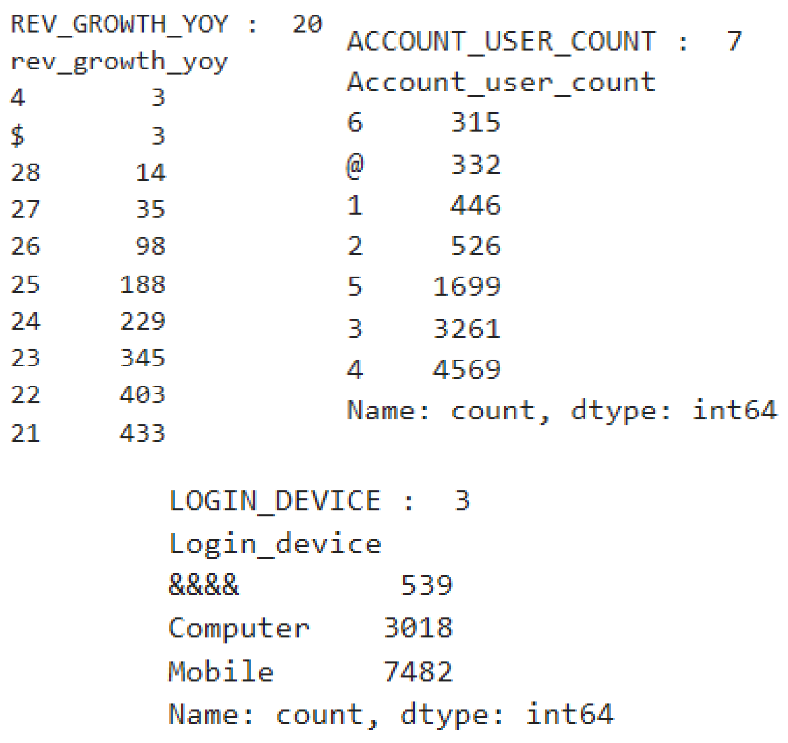
- City tier 1 experiences the highest churn rate compared to city tiers 2 and 3.
- Customers who prefer Debit and Credit Cards are more likely to churn.
- Male customers have a higher churn rate than females.
- Service score rating of 3 is most common and has the highest number of churned customers though there might not be any correlation due to data imbalance.
- The "Regular Plus" segment shows the highest churn.
- Single customers are more prone to churn than those who are divorced or married.
- Customers accessing services through mobile devices show higher churn rates.

3.3 Removal of Unwanted Variables

Unwanted variables that have no impact on the study can be removed from the dataset. In this case. “AccountID” variables contains all unique values but it will not be removed for now as one ID can have the same data for different variables. The variable will be removed later post EDA.

On inspection of all the unique values in the dataset, there can be seen a number of incorrect entries for some variables

Figure 26 - Snapshot of Incorrect Values in the Variables



```
REV_GROWTH_YOY :  20
rev_growth_yoy
4        3
$        3
28      14
27      35
26      98
25     188
24     229
23     345
22     403
21     433
```

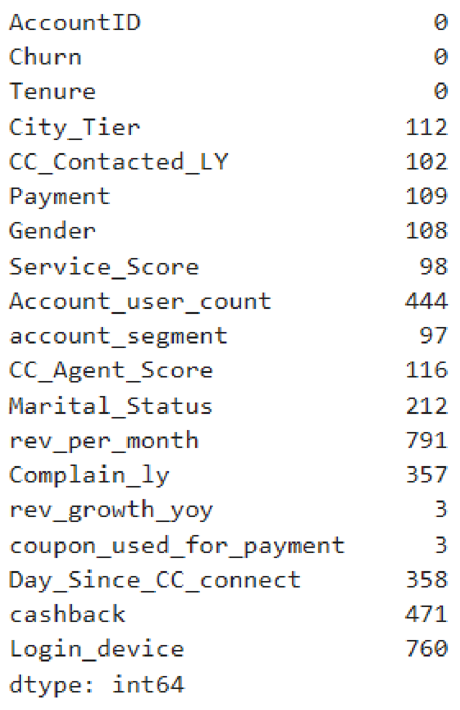
```
ACCOUNT_USER_COUNT :  7
Account_user_count
6     315
@     332
1     446
2     526
5    1699
3    3261
4    4569
Name: count, dtype: int64
```

```
LOGIN_DEVICE :  3
Login_device
&&&&         539
Computer    3018
Mobile      7482
Name: count, dtype: int64
```

From the snippet of some variables showing incorrect data such as “$”, “@”, “&&&&” and more, these incorrect values will be replaced with NaN or missing value and will be treated as a null value.

3.4 Missing Value Treatment

Figure 27 - Total number of Missing Values in the Variables

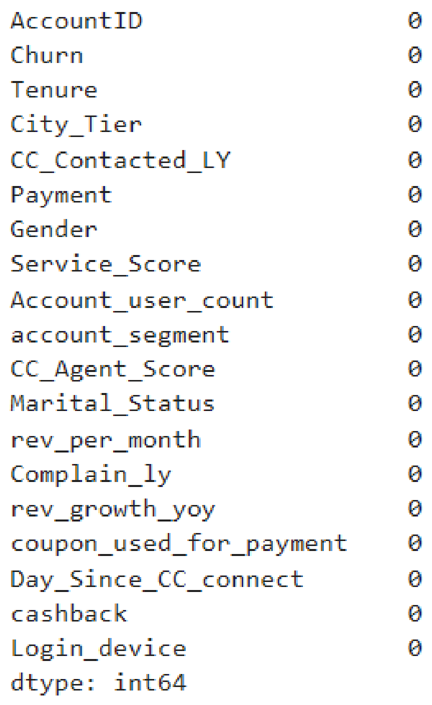


```
AccountID                 0
Churn                     0
Tenure                    0
City_Tier               112
CC_Contacted_LY         102
Payment                 109
Gender                  108
Service_Score            98
Account_user_count      444
account_segment          97
CC_Agent_Score          116
Marital_Status          212
rev_per_month           791
Complain_ly             357
rev_growth_yoy            3
coupon_used_for_payment   3
Day_Since_CC_connect    358
cashback                471
Login_device            760
dtype: int64
```

The missing values need to treated so that it does not impact future analysis and model building, The following steps are taken

- The missing values in the variables, City_Tier, Payment, Gender, Service_Score, account_segment, CC_Agent_Score, Marital_Status, Complain_ly and Login_device will be replaced with the Mode on the data as they are considered as categorical variables.
- The missing values in the variables, CC_Contacted_LY, Account_user_count, rev_per_month, rev_growth_yoy, coupon_used_for_payment, Day_Since_CC_connect and cashback will be replaced with the Median of the data as they are continuous numerical variables.

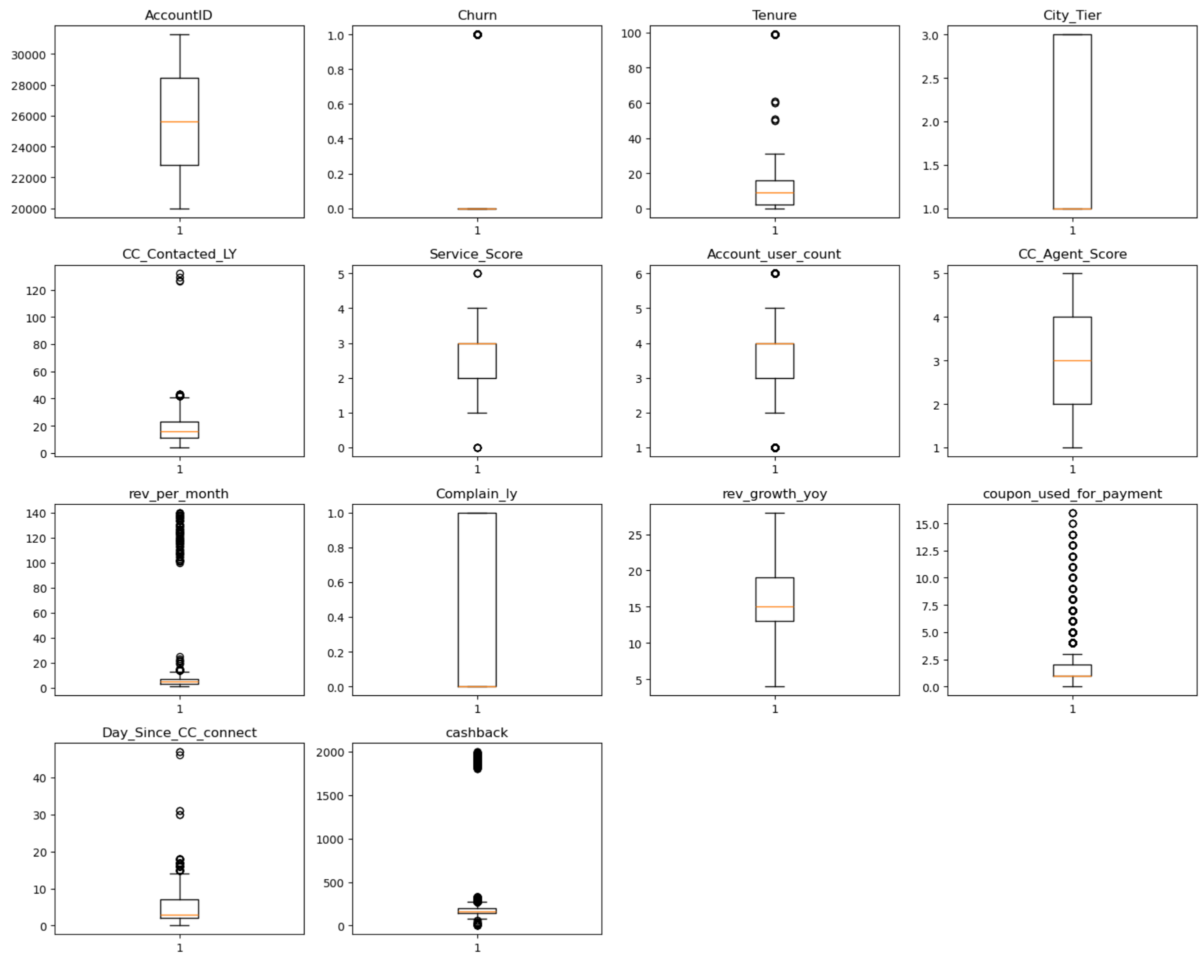
Figure 28 - Missing Values in the Variables Post Treatment



```
AccountID                  0
Churn                      0
Tenure                     0
City_Tier                  0
CC_Contacted_LY            0
Payment                    0
Gender                     0
Service_Score              0
Account_user_count         0
account_segment            0
CC_Agent_Score             0
Marital_Status             0
rev_per_month              0
Complain_ly                0
rev_growth_yoy             0
coupon_used_for_payment    0
Day_Since_CC_connect       0
cashback                   0
Login_device               0
dtype: int64
```

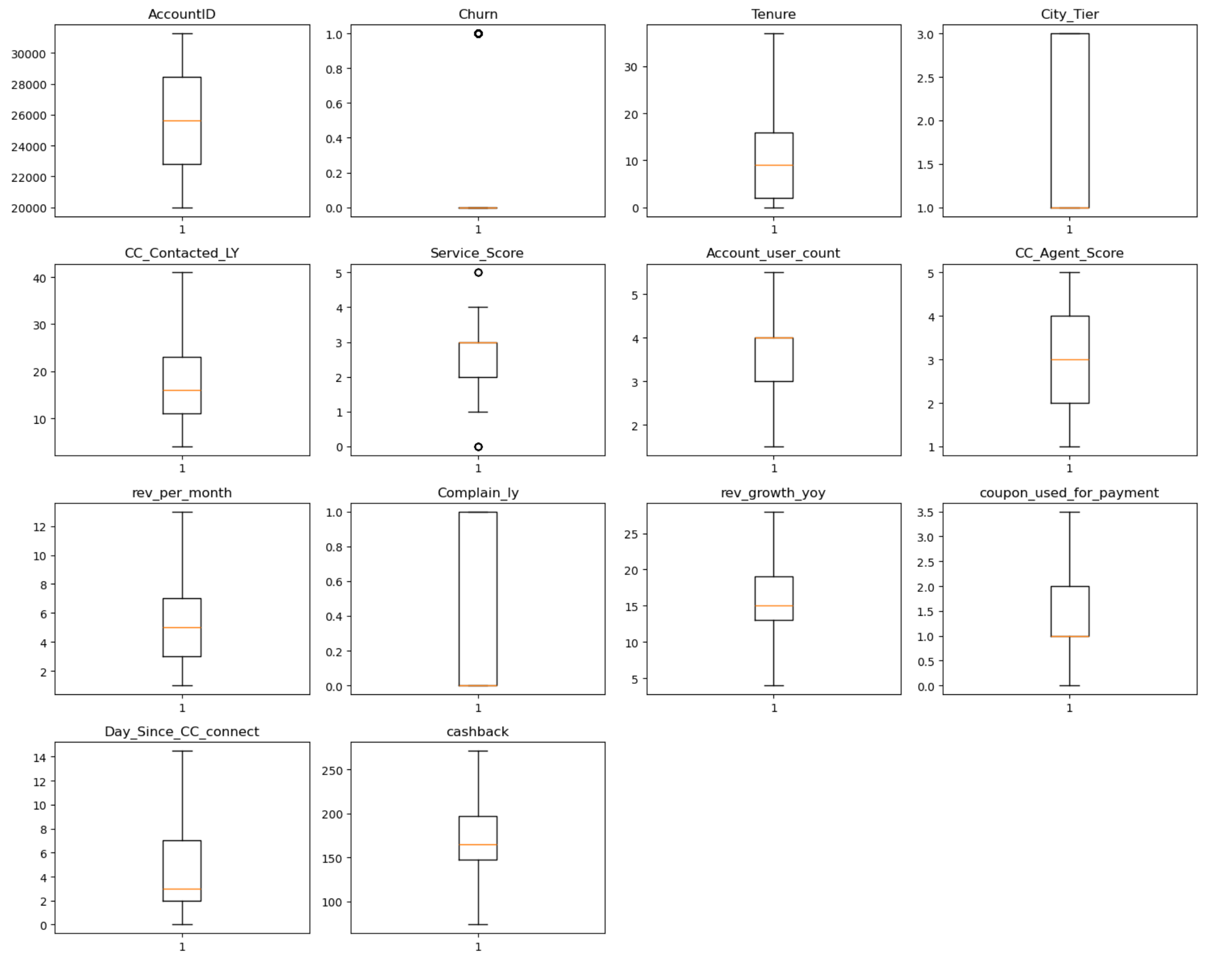
3.5 Outlier Treatment

Figure 29 - Boxplots of Numerical Variables with Outlier



There are a number of outliers present in the variables of the dataset and will need to be treated, Churn which is our target variable and Service_Score shows outlier however as it only contains categorical values, this will not be treated. The outliers present in the other variables will be treated with the upper and lower range of the variable.

Figure 30 - Boxplot of Numerical Variables Post Treatment



3.6 Variable Transformation

Variable transformation is the process of altering the values in a dataset to improve the performance of the model. The values of each of the variable need to be normalized as the data ranges are very varied for each of the variables which can be seen by comparing the ranges of some numerical variables like cashback with a variable like coupon_used_for_payment or rev_per_month.

The dataset will be scaled using the MinMax scaler from the sklearn library in python. Before we process with scaling, the categorical variables will need to be label encoded, which is each of the values will be converted to numerical values, for example in the variable Gender, Female will be encoded as 1 and Male will be encoded as 2 likewise in the variable account_segment_mapping, Super will be encoded as 1, Regular Plus + will be encoded as 2, Regular will be encoded as 3, HNI will be encoded as 4 and Super Plus will be encoded as 5. All the categorical variables are encoded similarly.

Following encoding, the dataset is scaled with MinMax scaler and The dataset is now normalized which will help create an optimized model.

3.7 Addition of New Variables

Currently no new variables will be added to the dataset as the variables in the dataset don't have much correlation and similarities with each other which would make it easier to include any new variable.

BUSINESS INSIGHTS FROM EDA

4.1 Data Imbalance and step to rectify imbalance.

On analysis of our target variable "Churn", we can see that 0 has 9,364 counts whereas 1 has 1,896 counts. The ratio of Churn = 0 to Churn = 1 is around 5:1. This suggests that the "Churn" value 1is much less frequent than the Churn value 0, which will create challenges in machine learning models.

Within the context of the business, SMOTE which is short for Synthetic Minority Over-sampling Technique can be used to address class imbalance in machine learning, especially when the target variable has a minority class that is underrepresented.

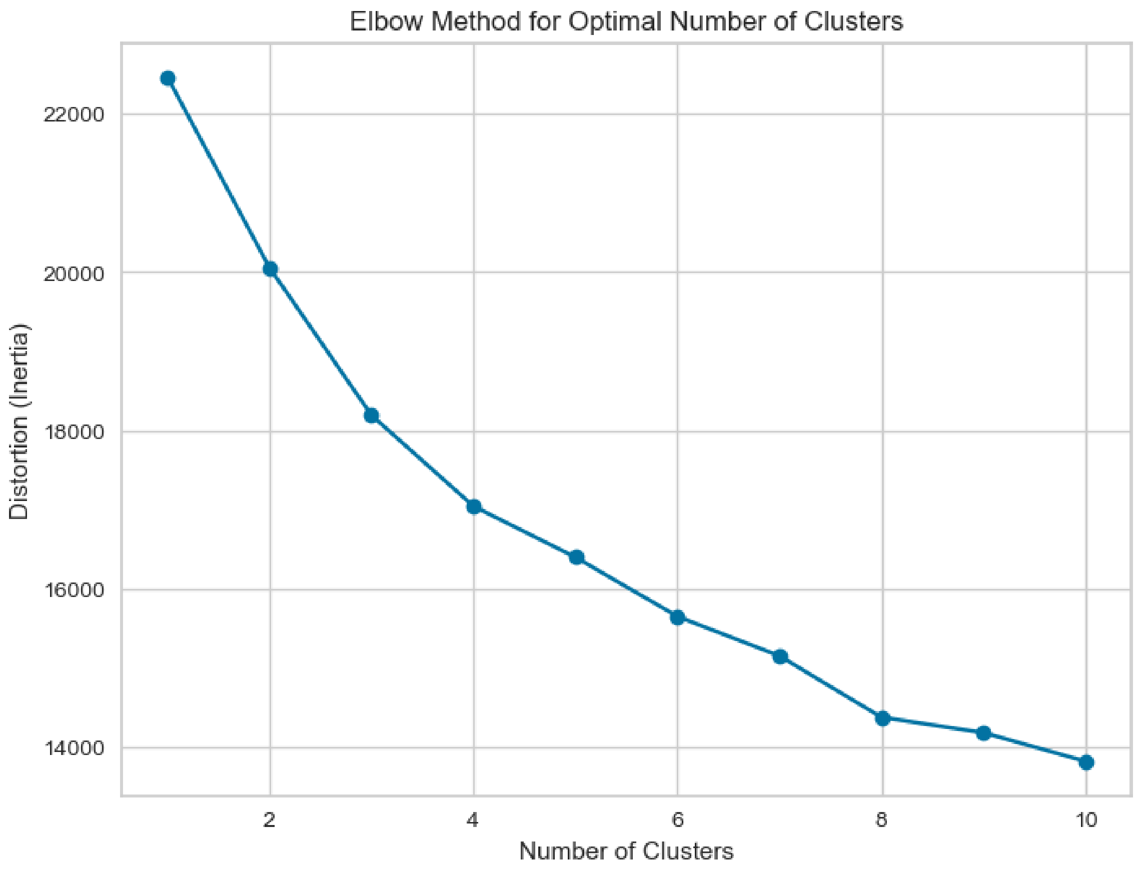
Other methods can be used such as clustering techniques like K-means to identify segments of customers that are more likely to churn. The customers can be segmented based on their behaviors such as spending, etc.

4.2 Insights using Clustering

The customers can be clustered into different segments using the K-means method which identifies similarities in the data and groups various data points which is closest to the centroid.

The elbow plot can be used to identify the optimal number of clusters which will be created.

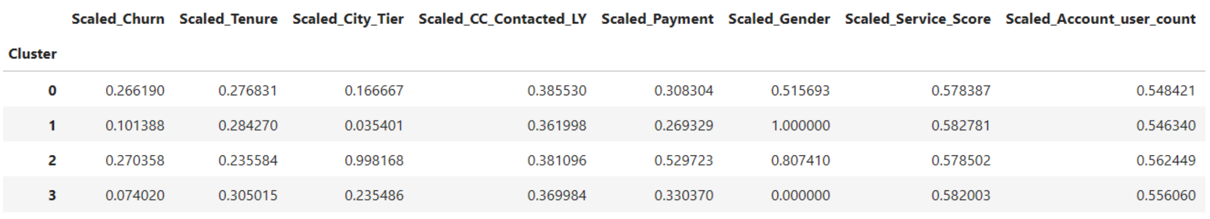
Figure 31 - Elbow Plot



Based on the elbow plot, the optimal range of clusters is 4-5. For this study we will have the optimal clusters as 4 (k=4).

Using k=4, we create 4 clusters in the dataset using k-means and get the following mean values of each variable in the clusters.

Figure 32 - Snapshot of data grouped by Clusters



Cluster	Scaled_Churn	Scaled_Tenure	Scaled_City_Tier	Scaled_CC_Contacted_LY	Scaled_Payment	Scaled_Gender	Scaled_Service_Score	Scaled_Account_user_count
0	0.266190	0.276831	0.166667	0.385530	0.308304	0.515693	0.578387	0.548421
1	0.101388	0.284270	0.035401	0.361998	0.269329	1.000000	0.582781	0.546340
2	0.270358	0.235584	0.998168	0.381096	0.529723	0.807410	0.578502	0.562449
3	0.074020	0.305015	0.235486	0.369984	0.330370	0.000000	0.582003	0.556060

The following insights can be made

Cluster 0:

- Churn: This cluster has a low churn rate compared to the other clusters.
- Tenure: This suggests that customers in this cluster have a moderate length of tenure, neither too short nor long-term.

- City Tier: These customers tend to belong to lower city tiers, suggesting they may come from regions with lower access to services or lower purchasing power.
- Contacted by Customer Care: Customers in this cluster seem to have been contacted by customer care somewhat frequently, which could indicate that they may have some issues or need more assistance.
- Marital Status: This group has a higher proportion of married customers.
- Complaint: This group has more complaint levels, indicating potential dissatisfaction or issues that need to be addressed.
- Coupon Usage: They tend to use fewer coupons compared to other clusters.

Cluster 1:

- Churn: This cluster has a very low churn rate, indicating strong loyalty.
- Tenure: This group has a moderate tenure, implying that they are not the most recent customers but still not long-term users.
- City Tier: Customers from higher city tiers are the majority, suggesting they have access to better services or higher income.
- Contacted by Customer Care: Moderate contact with customer care, which could indicate some concerns but not excessively.
- Gender: The cluster is mostly male customers.
- Marital Status: There is a portion of married customers in this cluster.
- Complaint: This cluster shows moderate complaint levels.
- Coupon Usage: They tend to use moderate coupon amounts.
- Login Device: They mostly use mobile devices.

Cluster 2:

- Churn: This cluster has a moderate churn rate, indicating a moderate risk of losing customers.
- Tenure: This group consists of newer customers.
- City Tier. These customers are from higher city tiers.
- Contacted by Customer Care: Moderate contact with customer care, indicating potential issues by the customer.
- Gender: There is a higher proportion of male customers in this group.
- Marital Status: A moderate proportion of married customers.

- Complaint: High complaint levels, indicating potential dissatisfaction.
- Coupon Usage: Moderate coupon usage suggests price sensitivity.
- Login Device: They seem to use more mobile as login but there are some computer users as well.

Cluster 3:

- Churn: This cluster has the lowest churn rate.
- Tenure: Customers in this group have a longer tenure.
- City Tier: These customers are mostly from mid-tier cities.
- Contacted by Customer Care: They have low contact with customer care, indicating high satisfaction.
- Gender: Majority of the gender is Female for this cluster.
- Marital Status: A moderate proportion of married customers.
- Complaint: Moderate complaint levels.
- Coupon Usage: Higher coupon usage suggests price sensitivity.
- Login Device: The majority of customers here are using mobile devices.

4.3 Other Insights

- Customers with shorter tenure are more likely to churn.
- The majority of customers are male, and most are married.
- The satisfaction score doesn't show significant variation between churned and non-churned customers.
- The most popular products are Mobile Phones and Laptops and Accessories, while Groceries and other categories sell the least.
- Customers from tier-2 cities are fewer in number, indicating limited market reach in these areas.
- The business should increase its presence in tier-2 cities to expand its customer base.
- Promoting bank standing instructions or UPI payments could offer customers a hassle-free and secure payment method.
- There's a need to improve service quality and address gaps in customer satisfaction. A customer survey can help gather better insights into their expectations.
- Investing in training for customer care executives can significantly enhance customer experiences and boost feedback scores.

- Offering personalized plans for customers based on both their spending and tenure with the company could foster loyalty.
- Creating special plans for married customers, such as family-oriented offers, could improve retention and appeal to this segment.

APPENDIX

Code Snapshots:

```
# importing the necessary libararies
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.preprocessing import MinMaxScaler
```

```
# loading the dataset
df = pd.read_excel('C:\\Users\\Ishaan Shakti\\Documents\\Great Lakes\\Capstone Project\\Project Work\\Customer Churn Data.xlsx', sheet_name = 'Data for D
df.head()
```

```
df.tail()
```

```
df.shape
```

```
df.info()
```

```
# viewing the statistical summary
df.describe().T
```

```
df.isnull().sum()
```

```
df.duplicated().sum()
```

```
plt.figure(figsize = (12, 7))
sns.countplot(data = df,x = 'Churn')
plt.title('Count Plot of Churned Customers', fontsize=14)
plt.xlabel('Churn')
plt.show()
```

```
plt.figure(figsize = (12, 7))
sns.countplot(data = df,x = 'City_Tier')
plt.title('Count Plot of City Tier', fontsize=14)
plt.xlabel('City Tier')
plt.show()
```

```
plt.figure(figsize = (12, 7))
sns.countplot(data = df,x = 'Payment')
plt.title('Count Plot of Payment Method', fontsize=14)
plt.xlabel('Payment Method')
plt.show()
```

```
plt.figure(figsize = (12, 7))
sns.countplot(data = df,x = 'Gender')
plt.title('Count Plot of Gender', fontsize=14)
plt.xlabel('Gender')
plt.show()
```

```
plt.figure(figsize = (12, 7))
sns.countplot(data = df,x = 'Account_user_count')
plt.title('Count Plot of Account_user_count', fontsize=14)
plt.xlabel('Account_user_count')
plt.show()
```

```
plt.figure(figsize = (12, 7))
sns.countplot(data = df,x = 'Account_user_count')
plt.title('Count Plot of Account Users', fontsize=14)
plt.xlabel('Account User Count')
plt.show()
```

```
plt.figure(figsize = (12, 7))
sns.countplot(data = df,x = 'account_segment')
plt.title('Count Plot of Account Segment', fontsize=14)
plt.xlabel('Account Segment')
plt.show()
```

```
plt.figure(figsize = (12, 7))
sns.countplot(data = df,x = 'CC_Agent_Score')
plt.title('Count Plot of CC Agent Score', fontsize=14)
plt.xlabel('CC Agent Score')
plt.show()
plt.figure(figsize = (12, 7))
sns.countplot(data = df,x = 'CC_Agent_Score')
plt.title('Count Plot of CC Agent Score', fontsize=14)
plt.xlabel('CC Agent Score')
plt.show()
```

```
plt.figure(figsize = (12, 7))
sns.countplot(data = df,x = 'Marital_Status')
plt.title('Count Plot of Marital Status', fontsize=14)
plt.xlabel('Marital Status')
plt.show()
```

```
plt.figure(figsize = (12, 7))
sns.countplot(data = df,x = 'Complain_ly')
plt.title('Count Plot of Complains in Last 12 Months', fontsize=14)
plt.xlabel('Customer Complains in Last 12 Months')
plt.show()
```

```
plt.figure(figsize = (12, 7))
sns.countplot(data = df,x = 'Login_device')
plt.title('Count Plot of Login Devices', fontsize=14)
plt.xlabel('Login Devices')
plt.show()
```

```
cols_list = df.select_dtypes(include=np.number).columns.tolist()
plt.figure(figsize=(15, 7))
sns.heatmap(df[cols_list].corr(), annot=True, vmin=-1, vmax=1, fmt=".3f", cmap="Spectral")
plt.title('Heat Map', fontsize=14)
plt.show()
```

```
plt.figure(figsize = (12, 7))
sns.countplot(data = df,x = 'City_Tier',hue = 'Churn')
plt.title('Count Plot of City Tier VS Churned Customers', fontsize=14)
plt.xlabel('Ciy Tier')
plt.show()
```

```
plt.figure(figsize = (12, 7))
sns.countplot(data = df,x = 'Payment',hue = 'Churn')
plt.title('Count Plot of Payment Method VS Churned Customers', fontsize=14)
plt.xlabel('Payment Method')
plt.show()
```

```
plt.figure(figsize = (12, 7))
sns.countplot(data = df,x = 'Gender',hue = 'Churn')
plt.title('Count Plot of Gender VS Churned Customers', fontsize=14)
plt.xlabel('Gender')
plt.show()
```

```
plt.figure(figsize = (12, 7))
sns.countplot(data = df,x = 'Service_Score',hue = 'Churn')
plt.title('Count Plot of Service Score VS Churned Customers', fontsize=14)
plt.xlabel('Service Score')
plt.show()
```

```
plt.figure(figsize = (12, 7))
sns.countplot(data = df,x = 'Account_user_count',hue = 'Churn')
plt.title('Count Plot of Account User Count VS Churned Customers', fontsize=14)
plt.xlabel('Account User Count')
plt.show()
```

```
plt.figure(figsize = (12, 7))
sns.countplot(data = df,x = 'account_segment',hue = 'Churn')
plt.title('Count Plot of Account Segment VS Churned Customers', fontsize=14)
plt.xlabel('Account Segment')
plt.show()
```

```
plt.figure(figsize = (12, 7))
sns.countplot(data = df,x = 'account_segment',hue = 'Churn')
plt.title('Count Plot of Account Segment VS Churned Customers', fontsize=14)
plt.xlabel('Account Segment')
plt.show()
```

```
plt.figure(figsize = (12, 7))
sns.countplot(data = df,x = 'CC_Agent_Score',hue = 'Churn')
plt.title('Count Plot of CC Agent Score VS Churned Customers', fontsize=14)
plt.xlabel('CC Agent Score')
plt.show()
```

```
plt.figure(figsize = (12, 7))
sns.countplot(data = df,x = 'Marital_Status',hue = 'Churn')
plt.title('Count Plot of Marital Status VS Churned Customers', fontsize=14)
plt.xlabel('Marital Status')
plt.show()
plt.figure(figsize = (12, 7))
sns.countplot(data = df,x = 'Complain_ly',hue = 'Churn')
plt.title('Count Plot of Complains Last Year VS Churned Customers', fontsize=14)
plt.xlabel('Complains in Last Year')
plt.show()
```

```
plt.figure(figsize = (12, 7))
sns.countplot(data = df,x = 'Login_device',hue = 'Churn')
plt.title('Count Plot of Login Device VS Churned Customers', fontsize=14)
plt.xlabel('Login Device')
plt.show()
```

```
# Function to check all values in dataset
for column in df.columns:
    if df[column].dtype == 'object':
        print(column.upper(),': ',df[column].nunique())
        print(df[column].value_counts().sort_values())
        print('\n')
```

```
# removing unwanted variables in Tenure
df['Tenure'] = df['Tenure'].replace('#',np.NaN)
df['Tenure'] = df['Tenure'].astype('Int64')
df["Tenure"].unique()
```

```
# replace null values with median
df['Tenure'] = df['Tenure'].fillna(df['Tenure'].median())
```

```
# removing unwanted variables in Gender
df['Gender'] = df['Gender'].replace('F','Female')
df['Gender'] = df['Gender'].replace('M','Male')
```

```
# removing unwanted variables in Account User Count
df['Account_user_count'] = df['Account_user_count'].replace('@',np.NaN)
df['Account_user_count'] = df['Account_user_count'].astype('float64')
```

```
df["Account_user_count"].unique()
```

```
# removing unwanted variables in Rev Per Month
df['rev_per_month'] = df['rev_per_month'].replace('+',np.NaN)
df['rev_per_month'] = df['rev_per_month'].astype('float64')
df["rev_per_month"].unique()
```

```
# removing unwanted variables in Rev Growth yoy'
df['rev_growth_yoy'] = df['rev_growth_yoy'].replace('$',np.NaN)
df['rev_growth_yoy'] = df['rev_growth_yoy'].astype('float64')
df["rev_growth_yoy"].unique()
```

```
# removing unwanted variables in Coupon Used For Payment
df['coupon_used_for_payment'] = df['coupon_used_for_payment'].replace('#',np.NaN)
df['coupon_used_for_payment'] = df['coupon_used_for_payment'].replace('$',np.NaN)
df['coupon_used_for_payment'] = df['coupon_used_for_payment'].replace('*',np.NaN)
df['coupon_used_for_payment'] = df['coupon_used_for_payment'].astype('float64')
df["coupon_used_for_payment"].unique()
```

```
# removing unwanted variables in Day Since CC Connect
df['Day_Since_CC_connect'] = df['Day_Since_CC_connect'].replace('$',np.NaN)
df['Day_Since_CC_connect'] = df['Day_Since_CC_connect'].astype('float64')
df["Day_Since_CC_connect"].unique()
```

```
# removing unwanted variables in Cashback
df['cashback'] = df['cashback'].replace('$',np.NaN)
df['cashback'] = df['cashback'].astype('float64')
df["cashback"].unique
```

```
# removing unwanted variables in Login Device
df['Login_device'] = df['Login_device'].replace('&&&&',np.NaN)
df["Login_device"].unique()
```

```
df.isnull().sum()
```

```
# treatment in City Tier
df['City_Tier'] = df['City_Tier'].fillna(df['City_Tier'].mode()[0])
```

```
# treatment in CC_Contacted_LY
df['CC_Contacted_LY'] = df['CC_Contacted_LY'].fillna(df['CC_Contacted_LY'].median())
```

```
# treatment in Payment
df['Payment'] = df['Payment'].fillna(df['Payment'].mode()[0])
```

```
# treatment in Gender
df['Gender'] = df['Gender'].fillna(df['Gender'].mode()[0])
```

```
# treatment in Service_Score
df['Service_Score'] = df['Service_Score'].fillna(df['Service_Score'].mode()[0])
```

```
# treatment in Account_user_count
df['Account_user_count'] = df['Account_user_count'].fillna(df['Account_user_count'].median())
```

```
# treatment in account_segment
df['account_segment'] = df['account_segment'].fillna(df['account_segment'].mode()[0])
```

```
# treatment in CC_Agent_Score
df['CC_Agent_Score'] = df['CC_Agent_Score'].fillna(df['CC_Agent_Score'].mode()[0])
```

```
# treatment in Marital_Status
df['Marital_Status'] = df['Marital_Status'].fillna(df['Marital_Status'].mode()[0])
```

```
# treatment in rev_per_month
df['rev_per_month'] = df['rev_per_month'].fillna(df['rev_per_month'].median())
```

```
# treatment in Complain_ly
df['Complain_ly'] = df['Complain_ly'].fillna(df['Complain_ly'].mode()[0])
```

```
# treatment in rev_growth_yoy
df['rev_growth_yoy'] = df['rev_growth_yoy'].fillna(df['rev_growth_yoy'].median())
```

```
# treatment in coupon_used_for_payment
df['coupon_used_for_payment'] = df['coupon_used_for_payment'].fillna(df['coupon_used_for_payment'].median())
```

```
# treatment in Day_Since_CC_connect
df['Day_Since_CC_connect'] = df['Day_Since_CC_connect'].fillna(df['Day_Since_CC_connect'].median())
```

```
# treatment in cashback
df['cashback'] = df['cashback'].fillna(df['cashback'].median())
```

```
# treatment in Login_device
df['Login_device'] = df['Login_device'].fillna(df['Login_device'].mode()[0])
```

```
df.isnull().sum()
```

```
# outlier detection using boxplot
numeric_columns = df.select_dtypes(include=np.number).columns.tolist()

plt.figure(figsize=(15, 12))

for i, variable in enumerate(numeric_columns):
    plt.subplot(4, 4, i + 1)
    plt.boxplot(df[variable], whis=1.5)
    plt.tight_layout()
    plt.title(variable)

plt.show()
```

```
# treating outliers
def outlier_treatment(col):
    sorted(col)
    Q1,Q3 = col.quantile([0.25,0.75])
    IQR = Q3-Q1
    lower_range = Q1 - (1.5 * IQR)
    upper_range = Q3 + (1.5 * IQR)
    return lower_range, upper_range
```

```
# List of columns to apply the outlier treatment
columns_to_process = [
    'Tenure', 'CC_Contacted_LY', 'Account_user_count', 'cashback', 'rev_per_month',
    'Day_Since_CC_connect', 'coupon_used_for_payment', 'rev_growth_yoy'
]

for col in columns_to_process:
    lw, up = outlier_treatment(df[col])
    df[col] = np.clip(df[col], lw, up)
```

```
# check outliers post treatment
numeric_columns = df.select_dtypes(include=np.number).columns.tolist()

plt.figure(figsize=(15, 12))

for i, variable in enumerate(numeric_columns):
    plt.subplot(4, 4, i + 1)
    plt.boxplot(df[variable], whis=1.5)
    plt.tight_layout()
    plt.title(variable)

plt.show()
```

```
# encoding payment variable
payment_mapping = {
    'Debit Card': 1,
    'UPI': 2,
    'Credit Card': 3,
    'Cash on Delivery': 4,
    'E wallet': 5
}

df['Payment'] = df['Payment'].replace(payment_mapping)
```

```
C:\Users\Ishaan Shakti\AppData\Local\Temp\ipykernel_11164\357238523.py:10: FutureWarning: Downcasting behavior in `replace` is deprecated and will be rem
oved in a future version. To retain the old behavior, explicitly call `result.infer_objects(copy=False)`. To opt-in to the future behavior, set `pd.set_o
ption('future.no_silent_downcasting', True)`
  df['Payment'] = df['Payment'].replace(payment_mapping)
```

```
# encoding gender variable
gender_mapping = {
    'Female': 1, 'F': 1,
    'Male': 2, 'M': 2
}

df['Gender'] = df['Gender'].replace(gender_mapping)
```

```
C:\Users\Ishaan Shakti\AppData\Local\Temp\ipykernel_11164\3322179126.py:7: FutureWarning: Downcasting behavior in `replace` is deprecated and will be rem
oved in a future version. To retain the old behavior, explicitly call `result.infer_objects(copy=False)`. To opt-in to the future behavior, set `pd.set_o
ption('future.no_silent_downcasting', True)`
  df['Gender'] = df['Gender'].replace(gender_mapping)
```

```
# encoding account segment variable
account_segment_mapping = {
    'Super': 1,
    'Regular Plus': 2, 'Regular +': 2,
    'Regular': 3,
    'HNI': 4,
    'Super Plus': 5, 'Super +': 5
}

df['account_segment'] = df['account_segment'].replace(account_segment_mapping)
```

```
# encoding marital status variable
marital_status_mapping = {
    'Single': 1,
    'Divorced': 2,
    'Married': 3
}

df['Marital_Status'] = df['Marital_Status'].replace(marital_status_mapping)
```

```
C:\Users\Ishaan Shakti\AppData\Local\Temp\ipykernel_11164\2153673498.py:8: FutureWarning: Downcasting behavior in `replace` is deprecated and will be rem
oved in a future version. To retain the old behavior, explicitly call `result.infer_objects(copy=False)`. To opt-in to the future behavior, set `pd.set_o
ption('future.no_silent_downcasting', True)`
  df['Marital_Status'] = df['Marital_Status'].replace(marital_status_mapping)
```

```
# encoding login device variable
login_device_mapping = {
    'Mobile': 1,
    'Computer': 2
}

df['Login_device'] = df['Login_device'].replace(login_device_mapping)
```

```
# List of columns to scale
columns_to_scale = [
    'Churn', 'Tenure', 'City_Tier', 'CC_Contacted_LY', 'Payment',
    'Gender', 'Service_Score', 'Account_user_count', 'account_segment',
    'CC_Agent_Score', 'Marital_Status', 'rev_per_month', 'Complain_ly',
    'rev_growth_yoy', 'coupon_used_for_payment', 'Day_Since_CC_connect',
    'cashback', 'Login_device'
]

# Initialize the MinMaxScaler
scaler = MinMaxScaler()

# Apply scaling to each column in the list
for col in columns_to_scale:
    df[f'Scaled_{col}'] = scaler.fit_transform(df[[col]])
```

```
# creating new dataframe with scaled data
scaled_columns = [col for col in df.columns if col.startswith('Scaled_')]
df_scaled = df[scaled_columns]
df_scaled
```

```
df_scaled.info()
```

```
df['Churn'].value_counts()
```

```
# to perform k-means clustering and compute silhouette scores
from sklearn.cluster import KMeans
# Elbow Method to determine the optimal number of clusters
distortions = []
for i in range(1, 11):  # Check for cluster sizes from 1 to 10
    kmeans = KMeans(n_clusters=i, random_state=42)
    kmeans.fit(df_scaled)
    distortions.append(kmeans.inertia_)  # Inertia is the distortion value

# Plotting the Elbow graph
plt.figure(figsize=(8, 6))
plt.plot(range(1, 11), distortions, marker='o')
plt.title('Elbow Method for Optimal Number of Clusters')
plt.xlabel('Number of Clusters')
plt.ylabel('Distortion (Inertia)')
plt.grid(True)
plt.show()
```

```
# Applying KMeans with the optimal number of clusters
optimal_clusters = 4
kmeans_final = KMeans(n_clusters=optimal_clusters, random_state=42)
df_scaled.loc[:, 'Cluster'] = kmeans_final.fit_predict(df_scaled)
df_scaled.head()
```

```
df_scaled.groupby('Cluster').mean()
```