CHURN PREDICTION MODEL

CAPSTONE PROJECT -FINAL SUBMISSION

BATCH: PGPDSBA.O. SEP22.A

Yaresh Vijayasundaram

TEXAS McCombs
The University of Texas at Austin
McCombs School of Business

Post Graduate Program in Data Science and Business Analytics
September 2022 - September 2023

Table of Contents

LIST OF FIGURES:

LIST OF TABLES:

ACKNOWLEDGMENT:

I extend my heartfelt appreciation to Mr Akshay for his patient guidance in helping me grasp the fundamentals. A special thank you to my capstone project mentor, Mr Sharath Srivasta, for his invaluable guidance and support. I am also grateful to Ms Megha and Mr Eshan, the Program Managers, for their continuous assistance over the past year. My sincere thanks go out to my colleagues as well for their support.

1. Introduction:

a. Problem statement & Business Objective:

The E-commerce company is facing tough competition and finding it challenging to keep its current customers. They want to create a special system to predict when a customer might stop using their services (churn prediction) and offer attractive deals to those customers to encourage them to stay. It's important to note that when one account leaves, multiple customers might be lost. The goal is to create a churn prediction model that can accurately identify accounts that are likely to stop using their services.

Once the model predicts potential churners, the company will design unique and targeted offers to keep those customers from leaving. However, they need to be careful not to offer too many free or heavily discounted deals that could lead to financial losses for the company.

The business report should explain how the churn prediction model works and how well it performs. Additionally, it should provide creative and effective campaign recommendations that will entice customers to stay without causing financial problems for the company. The recommendations should be based on the specific needs and preferences of potential churners, aiming for higher customer retention and overall profitability.

b. Project Rationale:

- The study is needed because the E-commerce company is facing tough competition and struggling to keep its current customers.
- Churn prediction is important to identify customers who might leave, so the company can try to keep them.
- The company wants to understand why customers are leaving and offer them special deals to encourage them to stay.
- By predicting churn, the company can focus on keeping its most valuable customers and improving customer loyalty.
- The study will provide valuable insights into customer behaviour and preferences, helping the company design targeted and effective marketing campaigns to increase customer retention and overall revenue.
- A successful churn prediction model and effective campaign recommendations can lead to increased customer satisfaction, positive brand perception, and improved financial performance for the company.

c. Understanding business/social opportunity

- The E-commerce company is facing intense competition, and customer retention has become a significant challenge in the current market environment.
- Developing a churn prediction model will enable the company to identify potential churners in advance and take proactive measures to retain them.
- Targeted offers and incentives can be provided to potential churners, tailored to their specific needs and preferences, to increase the chances of retaining them.
- By reducing customer churn, the company can maintain a stable revenue stream and achieve long-term sustainability, as retaining existing customers is more cost-effective than acquiring new ones.

- Data-driven decision-making based on the churn prediction model will help the company make informed business choices, resulting in enhanced customer experience and a competitive advantage over rivals.

In summary, using churn prediction and targeted offers allows the E-commerce company to better understand its business and social opportunities. This helps the company build stronger customer relationships, increase revenue, and stay ahead of the competition. By making informed decisions, the company can achieve long-term success in a challenging market.

d. About the Data.

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

Table 2: First Five rows of the dataset

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 11260 entries, 0 to 11259
Data columns (total 19 columns):
 #   Column                       Non-Null Count  Dtype
---  ------                       --------------  -----
 0   AccountID                    11260 non-null  int64
 1   Churn                        11260 non-null  int64
 2   Tenure                       11158 non-null  object
 3   City_Tier                    11148 non-null  float64
 4   CC_Contacted_LY              11158 non-null  float64
 5   Payment                      11151 non-null  object
 6   Gender                       11152 non-null  object
 7   Service_Score                11162 non-null  float64
 8   Account_user_count           11148 non-null  object
 9   account_segment              11163 non-null  object
 10  CC_Agent_Score               11144 non-null  float64
 11  Marital_Status               11048 non-null  object
 12  rev_per_month                11158 non-null  object
 13  Complain_ly                  10903 non-null  float64
 14  rev_growth_yoy               11260 non-null  object
 15  coupon_used_for_payment      11260 non-null  object
 16  Day_Since_CC_connect         10903 non-null  object
 17  cashback                     10789 non-null  object
 18  Login_device                 11039 non-null  object
dtypes: float64(5), int64(2), object(12)
memory usage: 1.6+ MB
```

Table 1: Data information

Data Shape: The dataset contains information on 11,260 accounts, with each account having 19 data columns.

Missing Data: It's essential to mention that some columns in the data have missing information, which means that certain details were not recorded for some accounts. For instance, variables like "Tenure," "City_Tier," and "CC_Contacted_LY" and few more entries with no data. This missing information could have an impact on our analysis and predictions.

Data Types: The dataset includes various data types, such as integers, floats, and objects (categorical data).

Churn Column: The "Churn" column is a significant attribute, representing whether an account has churned or not. This column will serve as the target variable for the churn prediction model.

Duplicate values:

There are no duplicate value present in the dataset.

2. EDA and Business Implication

Exploratory Data Analysis (EDA) helps in understanding data to find useful information and patterns. It helps us get a clear picture of the data, spot any unusual pattern and identify important relationships between different variables. EDA is essential for making informed decisions and guiding the next steps in data analysis.

Data Value Standardization and special characters before EDA enhances the reliability, accuracy, and interpretability of the insights gained during the analysis. It sets a solid foundation for meaningful exploration and helps in making informed business decisions based on reliable data patterns.

Data Value Standardization:

For the Gender column, we streamlined representations by unifying 'Female' and 'Male' as well as 'F' and 'M'. This resulted in two distinct categories: 'Female' and 'Male'. Similarly, we harmonized account segment names like 'Super', 'Regular Plus', etc., for consistency. The account_segment column now reflects: 'Super', 'Regular Plus', 'Regular', 'HNI', 'Super Plus', alongside some missing values.

Addressing Data Anomalies:

Certain variables (Tenure, Account_user_count, rev_per_month, rev_growth_yoy, coupon_used_for_payment, Day_Since_CC_connect, Login_device) contained special characters. To ensure data reliability, we replaced these characters with null values. This fortifies the groundwork for a dependable churn prediction model.

a. Univariate Analysis: Continuous Variable

1. Box plot for 'Tenure':

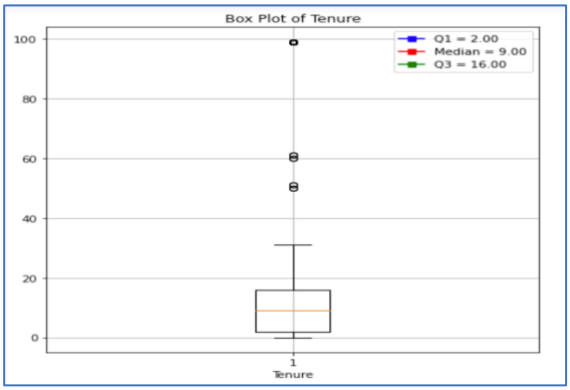

Figure 1: Box Plot of Tenure

The boxplot reveals that 50% of the tenure data is less than 9, showing the middle value of the distribution. Moreover, there are a few outliers in the dataset, which are data points lying far away from the majority of the data.

2. Histogram - CC_Contacted_LY:

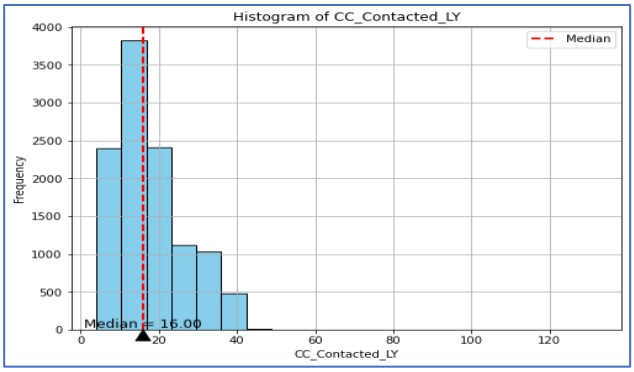

Figure 2: Histogram - CC_Contacted_LY

The "CC_Contacted_LY" histogram displays a median of 16, representing typical customer care contacts. Most interactions (0 to 50) highlight the importance of robust customer support to address their needs effectively.

3. Histogram - coupon_used_for_payment:

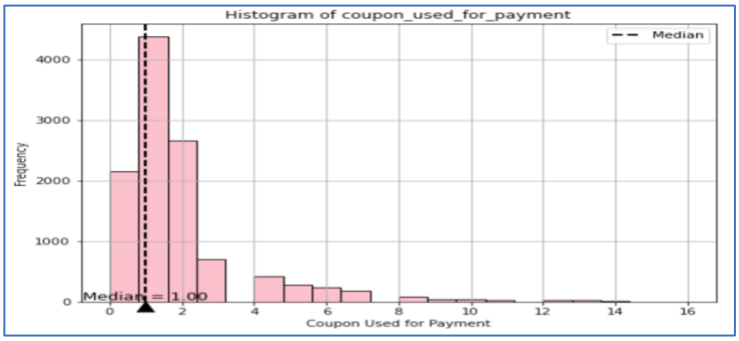

Figure 3: Histogram - coupon_used_for_payment:

The "coupon_used_for_payment" histogram unveils intriguing customer behaviour. A median of 1 shows most customers use coupons, while outliers, maxing at 14 uses, highlight a small group heavily reliant on coupons.

4. Box plot for rev_per_month

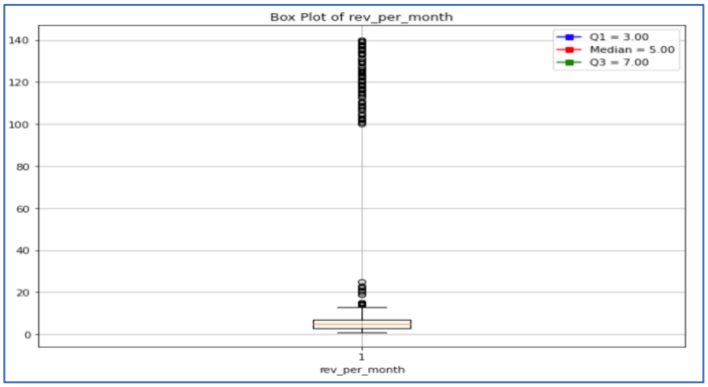

Figure 4: Box plot for rev_per_month

The "rev_per_month" box plot depicts average monthly revenue (in INR thousands) for the past year. It reveals that 75% of accounts generate less than 10 (INR 10,000), indicating a majority with moderate revenue. Some outliers, potentially from multi-account customers, exhibit higher values.

5. Day_Since_CC_connect- Box plot:

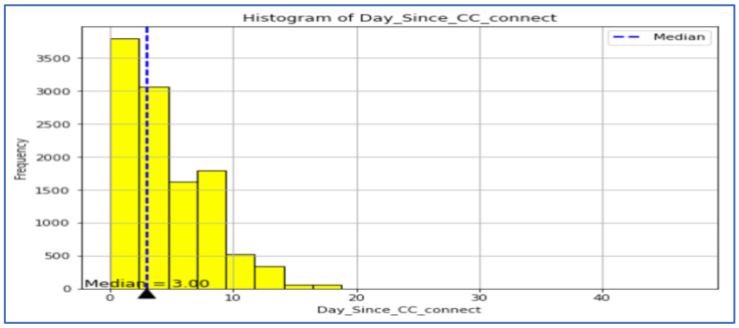

Figure 5:Boxplot - Day_since_CC_connect

The above box plot shows "Day_Since_CC_connect" reflecting time since last customer care contact. Median of 3 days suggests swift issue resolution for 50% of cases. Instances up to 47 days indicate occasional delayed interactions.

6. Boxplot of Cashback

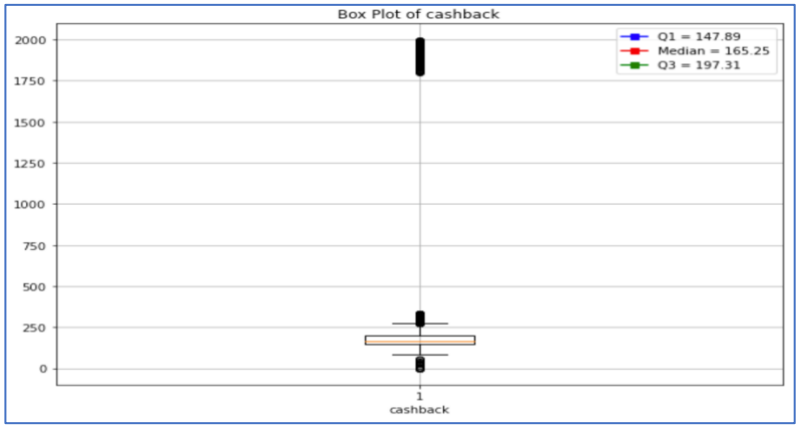

Figure 6: Boxplot of Cashback

The box plot displays "cashback," the monthly average given in the past year. Median of 165.25 shows typical cashback, but outliers, like 1997, indicate notably higher amounts for specific accounts.

7. Histogram - rev_growth_yoy:

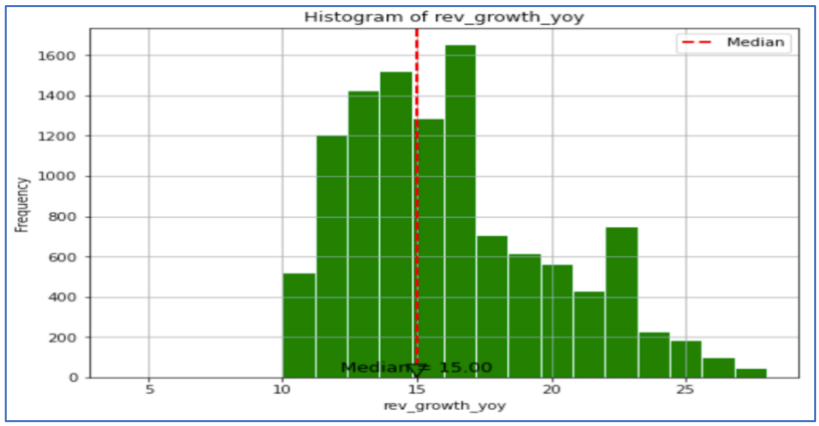

Figure 7:Histogram - rev_growth_yoy

The box plot shows "rev_growth_yoy," indicating revenue growth percentage. Median under 15% suggests moderate expansion for many accounts. Some exceptional cases hit 28% growth.

b. Univariate Analysis: Categorical Variable

8. Count plot of City Tier:

The below count plot shows customer distribution by city tiers. Most (7375) are in tier 1.0, followed by 3405 in tier 3.0. Tier 2.0 has the fewest (480). This guides customized strategies and offers.

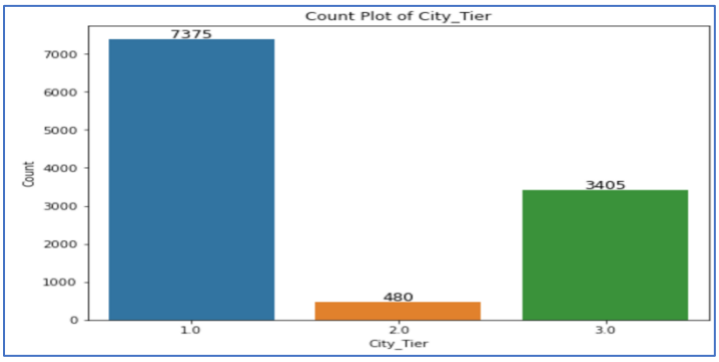

Figure 8:Count plot of City Tier

9. Donut Plot of Payment:

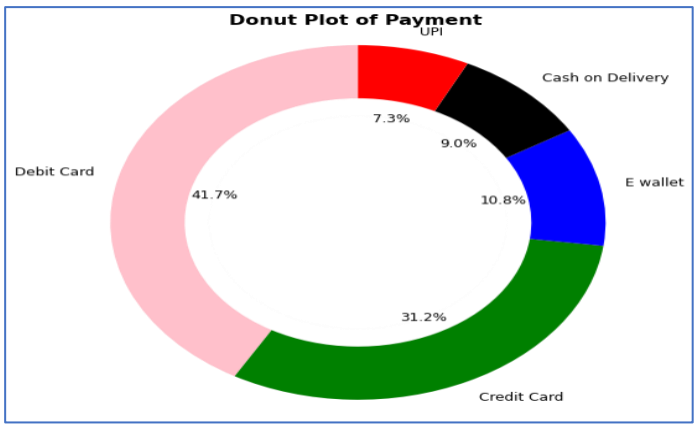

Figure 9:Donut Plot of Payment

The donut plot provides insights into the preferred payment modes of customers in the account. Notably, 41.7% of customers prefer using Debit Card, 31.2% opt for Credit Card, 10.8% use E-wallet, 9.0% choose Cash on Delivery, and 7.3% prefer UPI.

10. Pie chart of Gender:

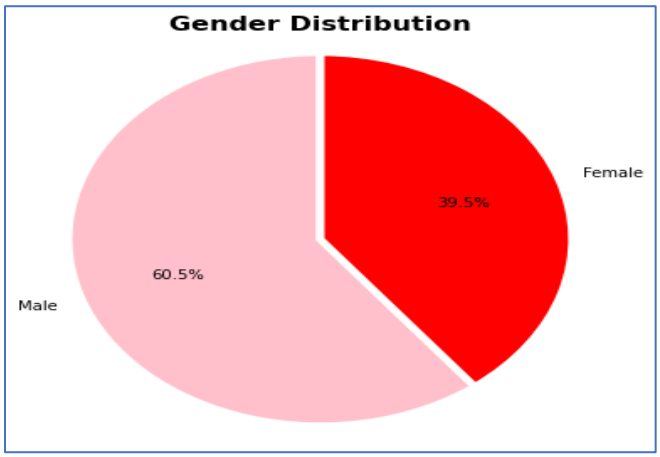

Figure 10:Pie chart of Gender:

The pie chart shows customer gender distribution: 60.5% male and 39.5% female. This insight aids tailored marketing strategies for diverse demographics.

11. Count Plot of Service score:

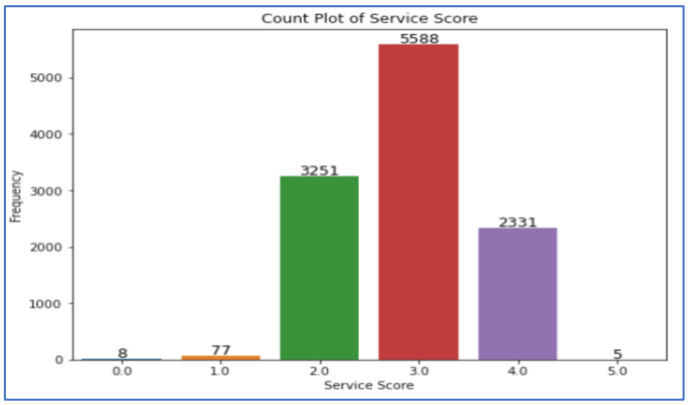

Figure 11:Count Plot of Service score

The count plot shows customer satisfaction scores. Most (5588) rate 3.0, moderately satisfied. 3251 gave 2.0, 2331 gave 4.0, indicating varying satisfaction. Few gave extreme scores: 77 at 1.0, 8 at 0.0, and 5 at 5.0. Analysing scores guides service improvements, enhancing loyalty and retention.

12. Count Plot of Account Segment:

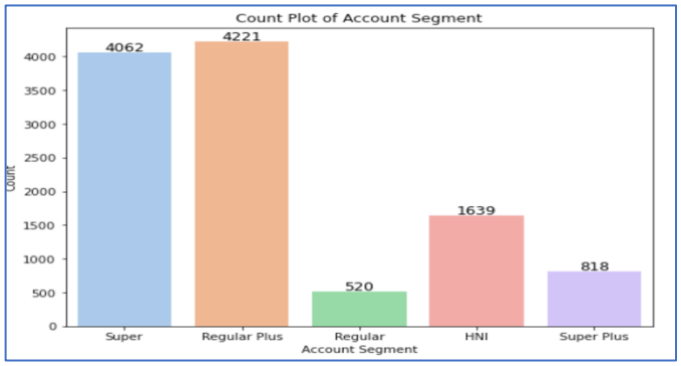

Figure 12:Count Plot of Account Segment

The count plot shows account segmentation by spending. "Regular Plus" has 4221 accounts, "Super" 4062, "HNI" 1639, "Super Plus" 818, and "Regular" 520. Segment analysis guides personalized offers and services, catering to varied spending patterns and preferences.

13. Count plot of CC_Agent_Score:

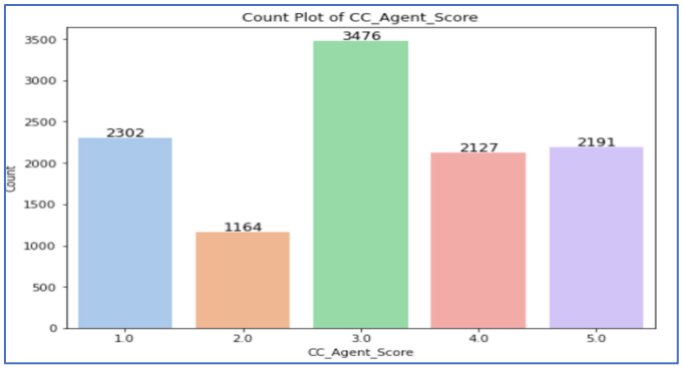

Figure 13:Count plot of CC_Agent_Score:

The count plot shows customer satisfaction scores for company's customer care service. The majority (3476) rated it 3.0, followed by 2302 at 1.0. Moreover, 2191 rated it 5.0, 2127 as 4.0, and 1164 as 2.0. This analysis guides service enhancements, promoting loyalty and retention.

14. Bar Plot of Marital Status:

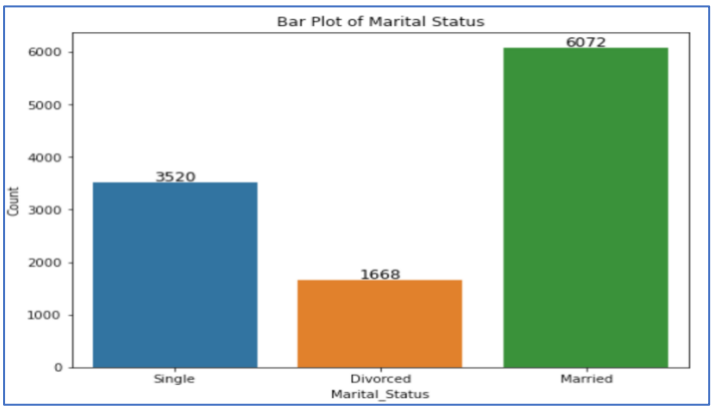

Figure 14: Bar Plot of Marital Status:

The bar plot shows customer marital status: most are married, followed by singles, and then divorced. This insight guides customized marketing and engagement approaches.

15. Pie Chart of Compliant_ly:

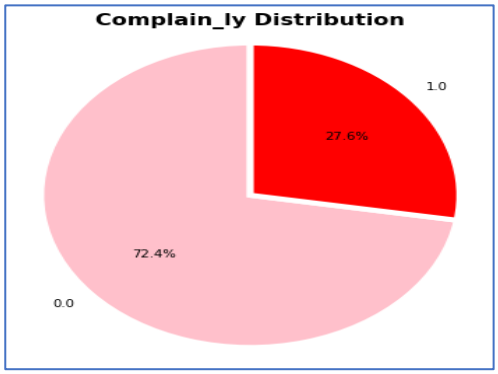

Figure 15: Pie Chart of Compliant_ly

The pie chart displays complaint occurrences in the last year. Most (72.4%) had no complaints, implying satisfactory service. However, 27.6% raised complaints, suggesting areas for improvement.

16. Login Device – Distribution:

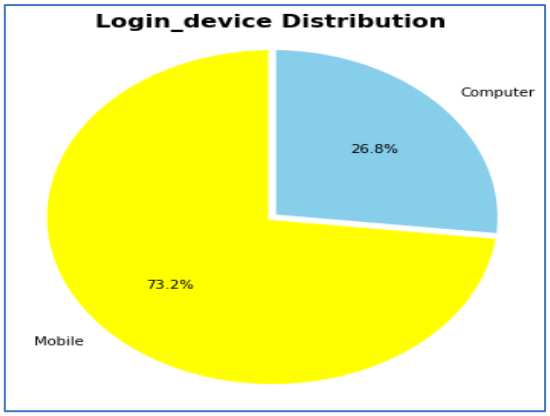

Figure 16:Login Device – Distribution

The distribution of login devices used by customers reveals that 73.2% prefer to access the platform through their mobile devices, while 26.8% opt for login via computers.

17. Bar Plot of Account_user_count:

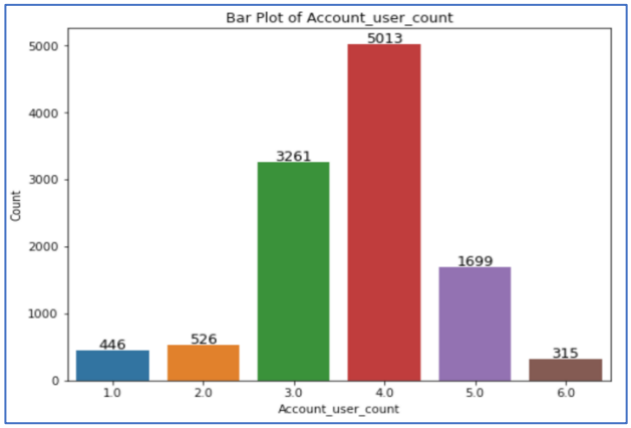

Figure 17:Bar Plot of Account_user_count:

The bar plot shows customer distribution per account. Most accounts have 4 customers (50.1%), followed by 3 (32.4%) and 5 (16.0%). Smaller proportions have 2 (5.0%), 1 (4.0%), and 6 (2.8%) customers.

18. Churn Distribution:

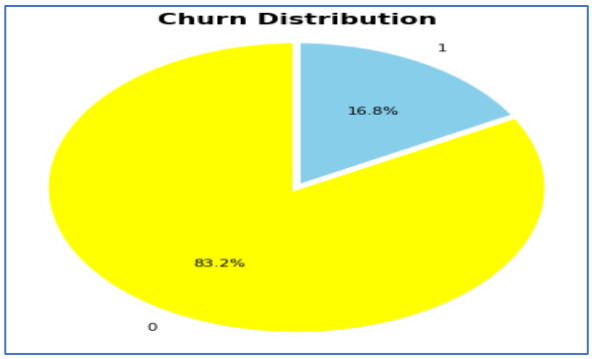

Figure 18:Churn Distribution

The "Churn" variable's distribution shows class imbalance: 83.2% non-churn (0), 16.8% churn (1). Imbalance affects model performance, favouring the majority class. Addressing this is crucial for accurate churn prediction.

c. Bivariate analysis (relationship between different variables, correlations)

19. Count Plot of City Tier Against Churn:

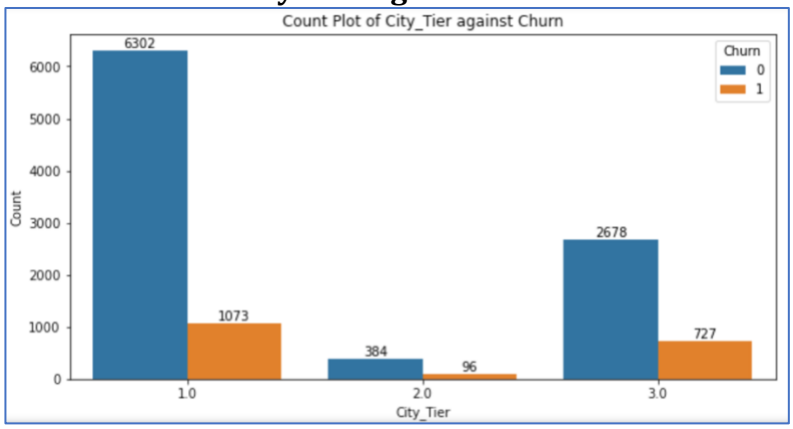

Figure 19:Count Plot of City Tier Against Churn

The above Count plot of City Tier vs. Churn shows consistent patterns across tiers. Churn rates don't vary significantly by city tier. This implies city tier might not strongly predict churn; other factors likely influence customer attrition.

20. Count Plot of payment against Churn:

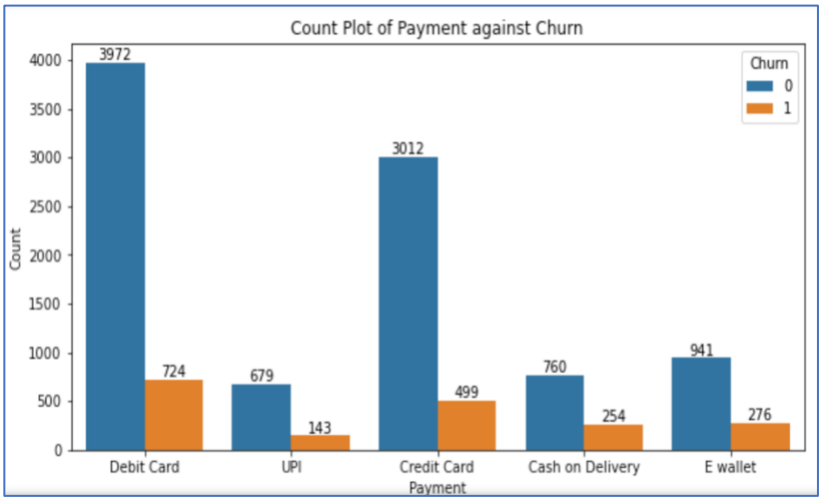

Figure 20: Count Plot of payment against Churn

The Count plot for payment method vs. churn displays similar patterns for churned and non-churned customers. Churn rates don't notably differ by payment mode. This implies payment method might not strongly impact customer churn in the E-commerce company.

21. Count plot of service score against Churn:

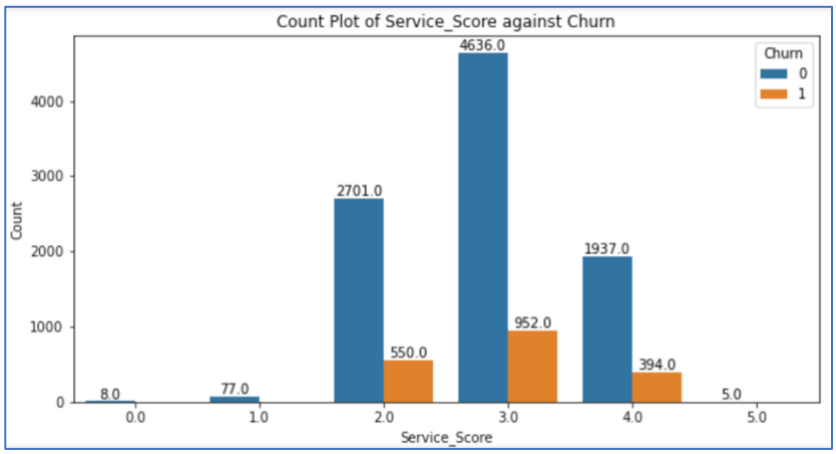

Figure 21:Count plot of service score against Churn

The Count plot for service score vs. churn shows consistent patterns for scores 2, 3, and 4, regardless of churn status. Interestingly, scores 0 and 1 didn't result in churn. This implies non-service factors might impact their decision to remain.

22. Count plot of Account segment against Churn:

The Count plot below for account segment vs. churn shows varying patterns. Regular Plus, Super, and HNI segments have higher churn, while Regular and Super Plus have lower churn. This indicates need for customized retention strategies per segment to boost loyalty and curb churn.

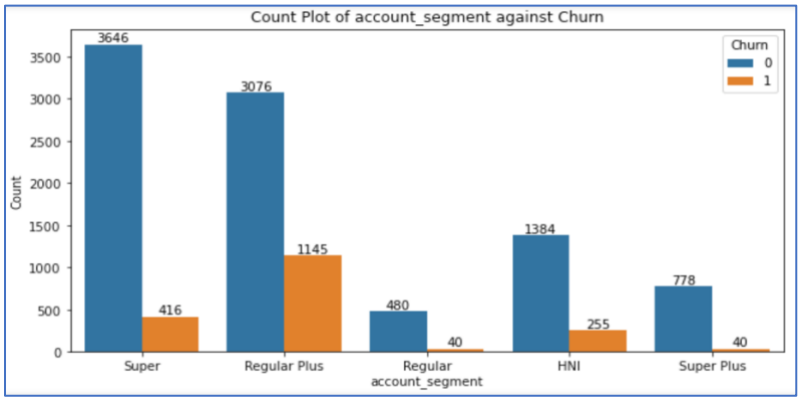

Figure 22: Count plot of Account segment against Churn

23. Count plot of Marital status against Churn:

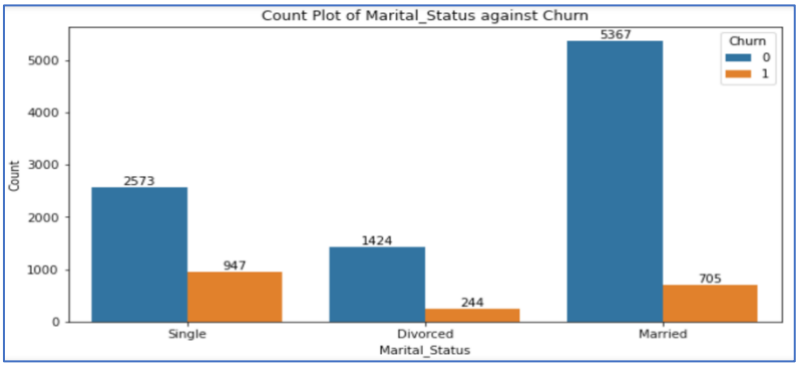

Figure 23: Count plot of Marital status against Churn

The bar chart for Marital Status vs. Churn reveals higher churn among singles than married or divorced. This implies single customers might be more influenced by factors causing churn.

24. Box plot of Tenure against Churn:

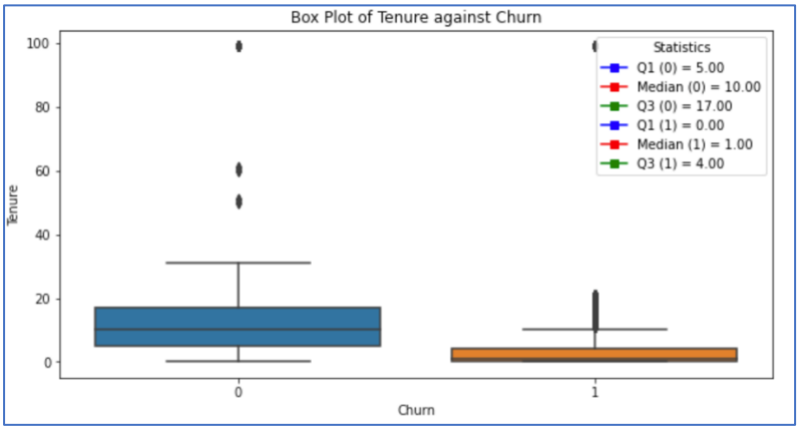

Figure 24: Box plot of Tenure against Churn

The boxplot of Account Tenure vs. Churn shows churning customers have shorter tenures. About 75% of churned customers have account tenures below 10. This highlights longer-term customers' loyalty and shorter-term customers' churn tendency. Addressing early churn factors can boost retention and satisfaction.

25. Box plot of CC_Contacted_LY against Churn:

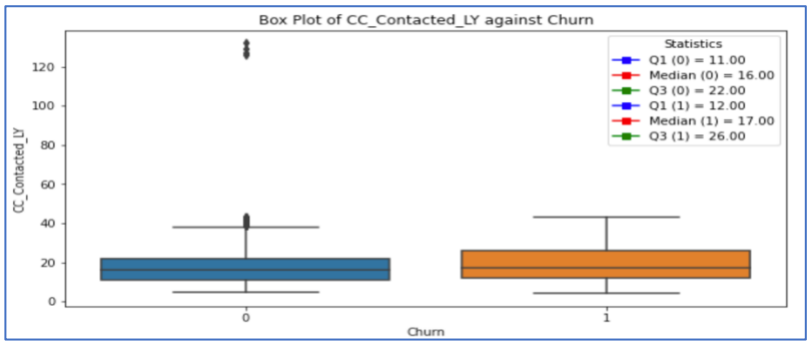

Figure 25: Box plot of CC_Contacted_LY against Churn

The Boxplot of CC_Contacted_LY vs. Churn shows similar patterns, but churned customers have slightly higher median contact values. This suggests churned customers contact customer care more often than non-churned customers.

d. Multivariate Analysis:

26. Pair plot of the continuous variables:

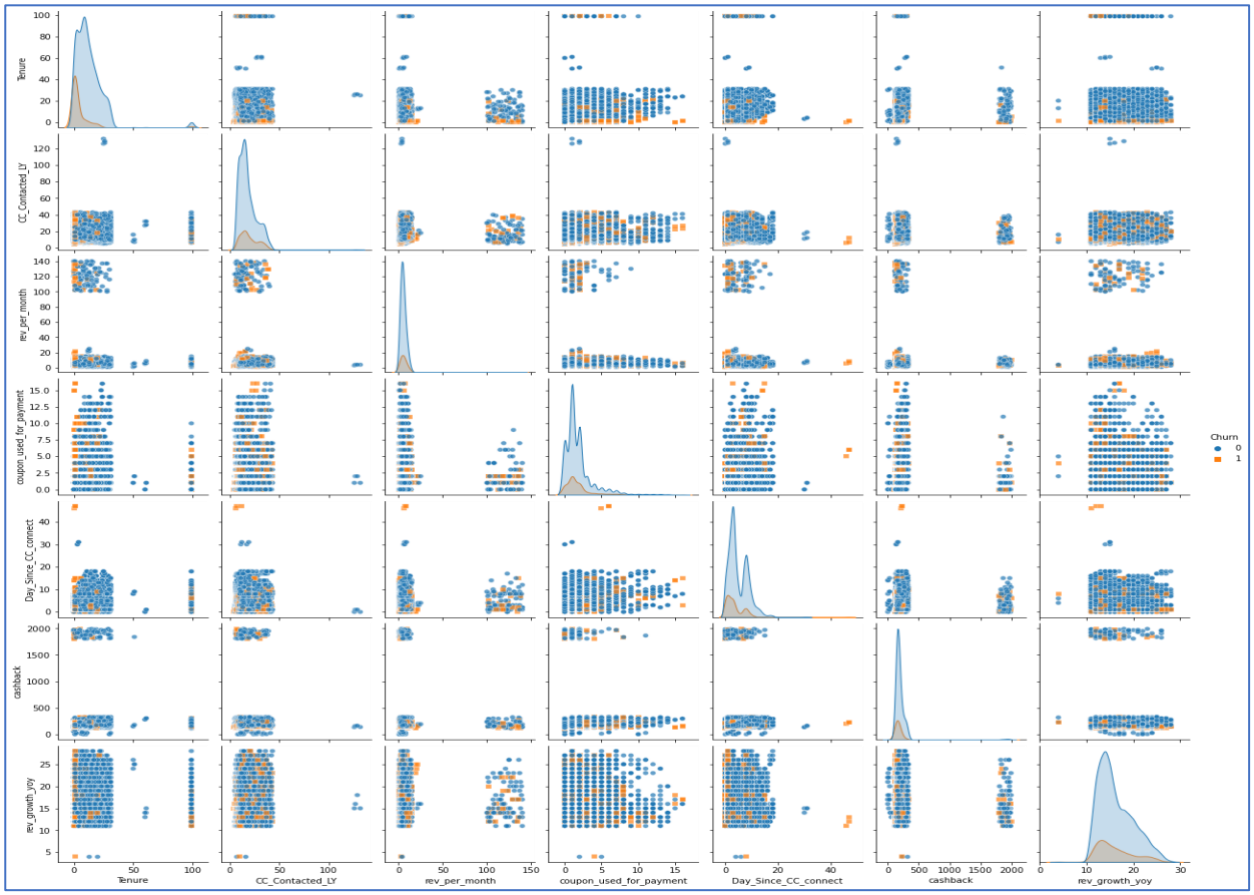

Figure 26:Pair plot of the continuous variables

From the above multi-variate pair plot analysis of continuous variables, a significant correlation was observed between churn and tenure. Churned customers had shorter account tenures compared to non-churn customers, highlighting the importance of account longevity in customer retention.

3. Data Cleaning and Pre-processing:

a. Approach used for identifying and treating missing values

Presence of Missing Values in Each Variable:

Churn	0
Tenure	218
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	473
Login_device	760
dtype: int64	

Table 4:Presence of Missing Values in Each Variable

After Missing Value treatment:

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	

Table 3:After Missing Value treatment

Imputing Missing Values:

To address missing values within the dataset, the following approaches were adopted:

For numerical columns:

'Tenure', 'CC_Contacted_LY', 'rev_per_month', 'rev_growth_yoy', 'coupon used for payment', 'Day_Since_CC_connect', 'cashback': The missing values were imputed with the respective column's **median.**

For categorical columns:

'City_Tier', 'Payment', 'Gender', 'Service_Score', 'account_segment', 'CC_Agent_Score', 'Marital_Status', 'Complain_ly', 'Login_device', 'Account_user_count': The missing values were imputed with the **mode** (most frequent value) of the corresponding column.

These imputation strategies were implemented to ensure dataset completeness while retaining valuable information for analysis.

b. Outlier Treatment.

Outlier treatment is applied only to numerical variables in the dataset. Outliers are extreme data points that can significantly influence statistical analyses and predictive models. Handling outliers is essential to ensure the accuracy and reliability of our analysis. By addressing outliers, we aimed to reduce bias and maintain the integrity of the data for meaningful insights and decision-making.

Log Transformation:

Initially, we applied the log transformation to the numerical variables in the dataset. The log transformation is a common method used to reduce the impact of skewness in the data and make it more normally distributed.

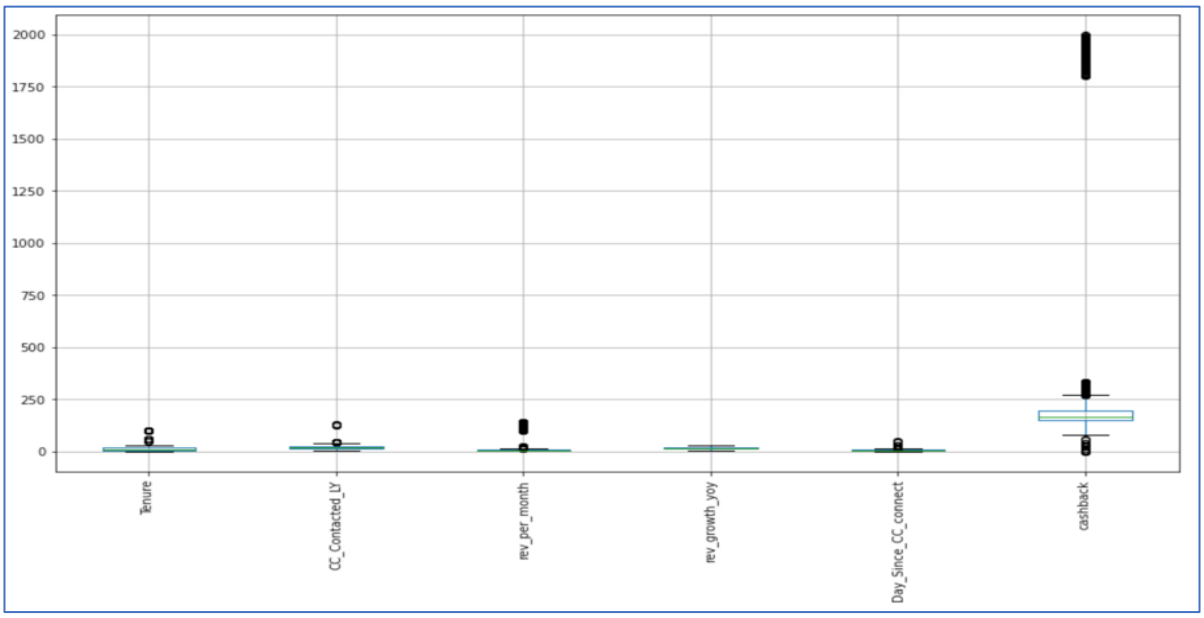

Figure 27: Before Outlier Treatment - Numerical Variable

After applying the log transformation, we noticed a slight decrease in the number of outliers in most numerical variables. Log transformation is effective in reducing the influence of extreme values and creating a more standardized distribution. However, we observed that the variable 'Coupon_used_for_payment' continued to exhibit high outliers even after the transformation. As a result, we decided to exclude 'Coupon_used_for_payment' and all categorical variables from the log transformation process to ensure its effectiveness on the remaining numerical variables.

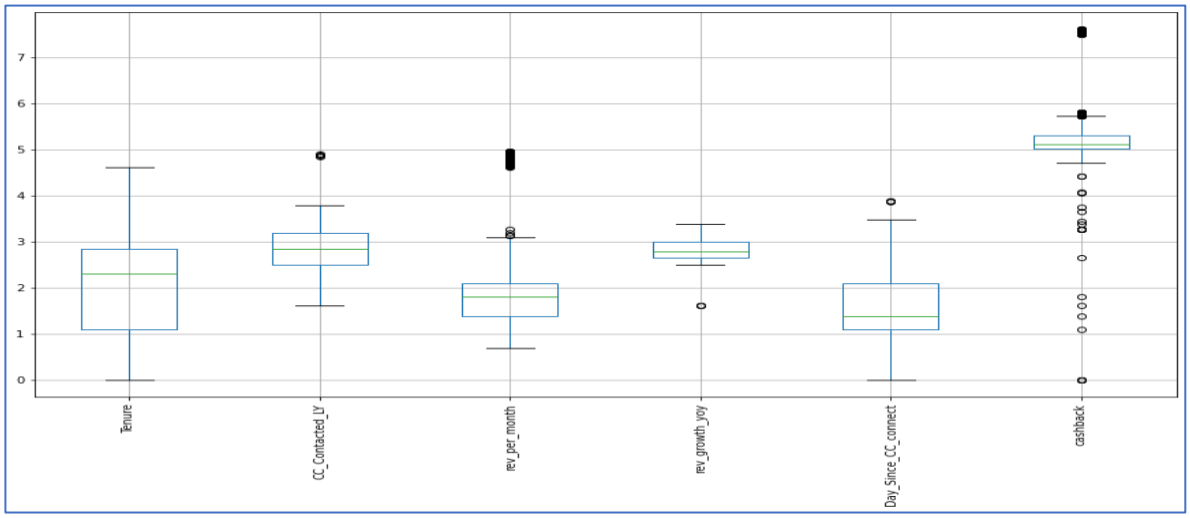

Figure 28: Applying Log transformation - Numerical Variable

Interquartile Range (IQR) method:

After applying the log transformation, we will implement the Interquartile Range (IQR) method to handle outliers. To handle outliers in the data, we used a method called the Interquartile Range (IQR) method. This method calculates the lower and upper limits based on the first and third quartiles of the data. Any data points that fall below the lower limit or above the upper limit are considered outliers and removed from the dataset. This helps to keep the data reliable by excluding extreme values that could affect the analysis and model accuracy. However, during the outlier removal process, we included the. 'Coupon_used_for_payment' variable along with the other numerical variables.

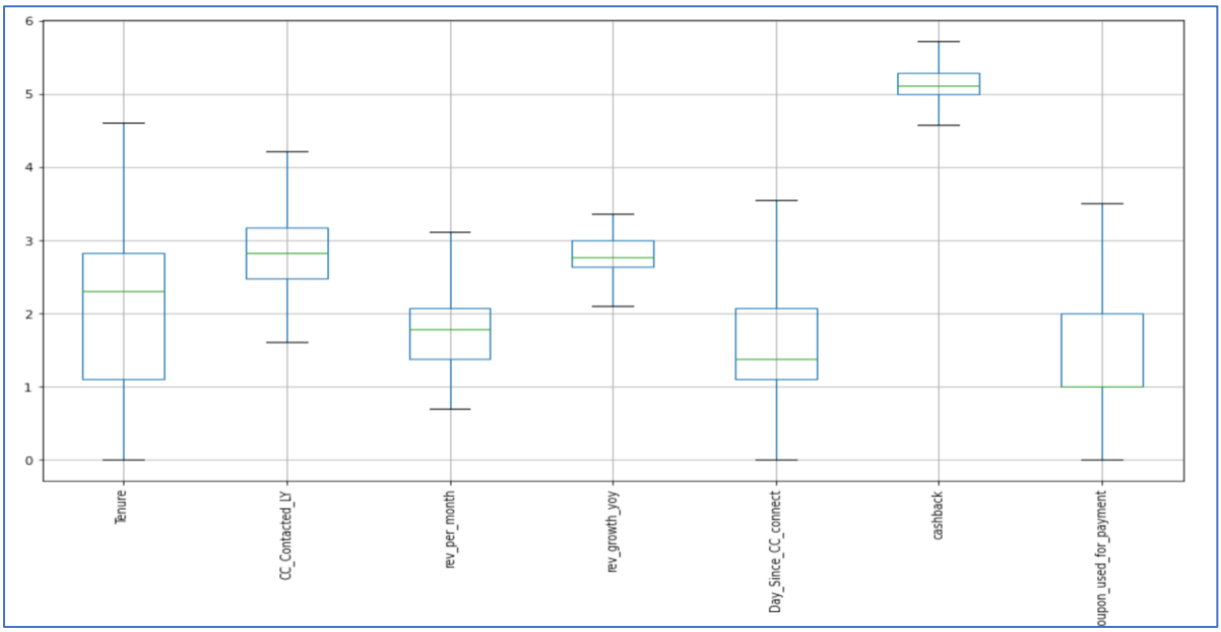

Figure 29: After Outlier treatment using IQR Method

c. Variable transformation

1) Encoding the categorical Variables

In the data pre-processing stage, ordinal encoding and one-hot encoding techniques were applied to handle categorical variables in the dataset.

Ordinal Encoding for 'Account Segment':
The 'Account Segment' variable, reflecting spending-based segments, underwent ordinal encoding. Categories ('Regular', 'Regular Plus', 'Super', 'Super Plus', 'HNI') were assigned numerical values (1 to 5) to establish a hierarchy based on spending patterns. This empowers the model to capture varying segment significance linked to spending behaviours.

One-Hot Encoding for Categorical Variables:
For categorical variables such as 'Payment,' 'Gender,' 'Marital Status,' and 'Login Device,' one-hot encoding was performed. One-hot encoding creates binary columns for each category within a categorical variable.

Payment_Cash on Delivery	Payment_Credit Card	Payment_Debit Card	Payment_E wallet	Payment_UPI	Gender_Female	Gender_Male	Marital_Status_Divorced	Marital_Status_Married	Marit
0	0	1	0	0	1	0	0	0	
0	0	0	0	1	0	1	0	0	
0	0	1	0	0	0	1	0	0	
0	0	1	0	0	0	1	0	0	
0	1	0	0	0	0	1	0	0	

Table 5: First Five rows after encoding

2) Data Type Standardization: Uniform Conversion to float64

This step involves the uniform conversion of columns with varying data types, such as float64 and uint8, to a consistent float64 data type. Achieved through the astype() function, this process ensures compatibility for seamless data processing and model utilization. Refer to Table No. [7] for specific column details.

Datatype before Variable Transformation:

Tenure	float64
CC_Contacted_LY	float64
rev_per_month	float64
rev_growth_yoy	float64
Day_Since_CC_connect	float64
cashback	float64
coupon_used_for_payment	float64
Churn	int64
City_Tier	float64
Service_Score	float64
account_segment	object
CC_Agent_Score	float64
Complain_ly	float64
Account_user_count	float64
Payment_Cash on Delivery	uint8
Payment_Credit Card	uint8
Payment_Debit Card	uint8
Payment_E wallet	uint8
Payment_UPI	uint8

Table 6:Datatype before Variable Transformation

Datatype after Variable Transformation:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 11260 entries, 0 to 11259
Data columns (total 26 columns):
 #   Column                       Non-Null Count  Dtype
---  ------                       --------------  -----
 0   Tenure                       11260 non-null  float64
 1   CC_Contacted_LY              11260 non-null  float64
 2   rev_per_month                11260 non-null  float64
 3   rev_growth_yoy               11260 non-null  float64
 4   Day_Since_CC_connect         11260 non-null  float64
 5   cashback                     11260 non-null  float64
 6   coupon_used_for_payment      11260 non-null  float64
 7   Churn                        11260 non-null  float64
 8   City_Tier                    11260 non-null  float64
 9   Service_Score                11260 non-null  float64
 10  account_segment              11260 non-null  float64
 11  CC_Agent_Score               11260 non-null  float64
 12  Complain_ly                  11260 non-null  float64
 13  Account_user_count           11260 non-null  float64
 14  Payment_Cash on Delivery     11260 non-null  float64
 15  Payment_Credit Card          11260 non-null  float64
 16  Payment_Debit Card           11260 non-null  float64
 17  Payment_E wallet             11260 non-null  float64
 18  Payment_UPI                  11260 non-null  float64
 19  Gender_Female                11260 non-null  float64
 20  Gender_Male                  11260 non-null  float64
 21  Marital_Status_Divorced      11260 non-null  float64
 22  Marital_Status_Married       11260 non-null  float64
 23  Marital_Status_Single        11260 non-null  float64
 24  Login_device_Computer        11260 non-null  float64
 25  Login_device_Mobile          11260 non-null  float64
dtypes: float64(26)
memory usage: 2.2 MB
```

Table 7:Datatype after Variable Transformation

After completing the data processing steps, it is evident that the dataset is now free from any null values or duplicate entries. The cleaning process has effectively prepared the data for further analysis, ensuring that all necessary information is available.

d. VIF (Variance Inflation Factor):

VIF (Variance Inflation Factor) is a metric used to identify potential multicollinearity in a regression model, where predictor variables are highly correlated with each other. We calculated the VIF values for each variable in the dataset using the statsmodels library. Higher VIF values indicate stronger correlations between variables, which can lead to instability in the regression model's coefficients. By examining the VIF values, we can detect and address any multicollinearity issues that might affect the accuracy and interpretability of the regression analysis.

	variables	VIF
24	Login_device_Mobile	inf
14	Payment_Credit Card	inf
23	Login_device_Computer	inf
22	Marital_Status_Single	inf
21	Marital_Status_Married	inf
20	Marital_Status_Divorced	inf
19	Gender_Male	inf
18	Gender_Female	inf
17	Payment_UPI	inf
16	Payment_E wallet	inf
15	Payment_Debit Card	inf
13	Payment_Cash on Delivery	inf
6	cashback	1.555314
7	City_Tier	1.395960
0	Tenure	1.309279
5	Day_Since_CC_connect	1.271953
4	coupon_used_for_payment	1.270187
9	account_segment	1.257874
8	Service_Score	1.222777
12	Account_user_count	1.154175
2	rev_per_month	1.116758
1	CC_Contacted_LY	1.034333
3	rev_growth_yoy	1.028537
10	CC_Agent_Score	1.018709

Table 9:Variance Inflation Factor

	variables	VIF
7	City_Tier	2.634244
11	Payment_Debit Card	2.027588
10	Payment_Credit Card	1.713547
13	Gender_Female	1.631843
15	Marital_Status_Single	1.595491
6	cashback	1.393362
8	Complain_ly	1.372614
16	Login_device_Computer	1.362332
0	Tenure	1.285722
14	Marital_Status_Divorced	1.259128
5	Day_Since_CC_connect	1.256007
9	Payment_Cash on Delivery	1.220723
4	coupon_used_for_payment	1.191023
12	Payment_UPI	1.166026
2	rev_per_month	1.098425
1	CC_Contacted_LY	1.025961
3	rev_growth_yoy	1.017203

Table 8:Variance Inflation Factor after removing value above 5

We utilized the VIF (Variance Inflation Factor) values to identify highly correlated variables in the dataset. Variables with higher VIF values indicate stronger correlations. To maintain the stability of the regression model and avoid multicollinearity issues, we decided to remove variables with VIF values above 5. This careful selection process resulted in a set of variables with minimal correlation, ensuring the reliability and accuracy of our regression analysis.

e. Scaling & Clustering:

Scaling the data is essential before clustering to ensure equal treatment of features and prevent bias caused by variable scales. By using standardization, we have transformed the data. The target variable 'Churn' was excluded from scaling to maintain its original interpretation.

	Tenure	CC_Contacted_LY	rev_per_month	rev_growth_yoy	coupon_used_for_payment	Day_Since_CC_connect	cashback	Churn	City_Tier	Service_Score
0	-0.361753	-1.969009	1.189540	-1.609808	-0.431239	0.421668	-0.337449	1.0	3.0	3.0
1	-1.854829	-1.411937	0.724124	-0.236250	-1.338515	-2.081514	-1.473097	1.0	1.0	3.0
2	-1.854829	1.329515	0.445614	-0.544393	-1.338515	-0.144788	-0.204479	1.0	1.0	2.0
3	-1.854829	-0.136564	0.969787	1.699671	-1.338515	-0.144788	-1.053652	1.0	3.0	2.0
4	-1.854829	-0.596825	-0.721590	-1.609808	-0.431239	-0.144788	-1.191245	1.0	1.0	2.0

Table 10: Dataset after Scaling

The table above displays the dataset with numerical variables scaled using standardization, which transforms them to have a mean of 0 and a standard deviation of 1. This ensures that all numerical features are on the same scale, allowing for fair and unbiased clustering analysis. Additionally, the categorical variables have been one-hot encoded to convert them into a numerical format suitable for clustering. By performing these transformations, we create a level for all features, enabling us to identify meaningful patterns and groupings in the data without being influenced by variations in the scales of different variables.

Elbow Plot :

We used the elbow plot method to find the best number of clusters for the clustering analysis. The plot shows how the inertia (a measure of cluster quality) changes with the number of clusters. This helps us determine the right number of clusters to use for our analysis in an easy and effective way.

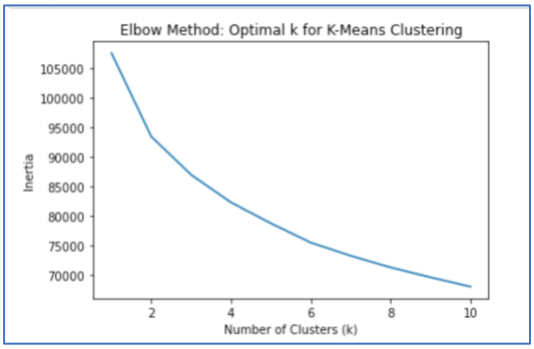

Figure 30: Elbow Plot

Although the elbow plot suggests that two clusters may be the optimal choice based on inertia, we have decided to use **3 clusters** for our analysis. The reason behind this decision is to explore potential patterns and subgroups within the data that might not be evident with just two clusters. By utilizing 3 clusters, we aim to gain a more Business insights.

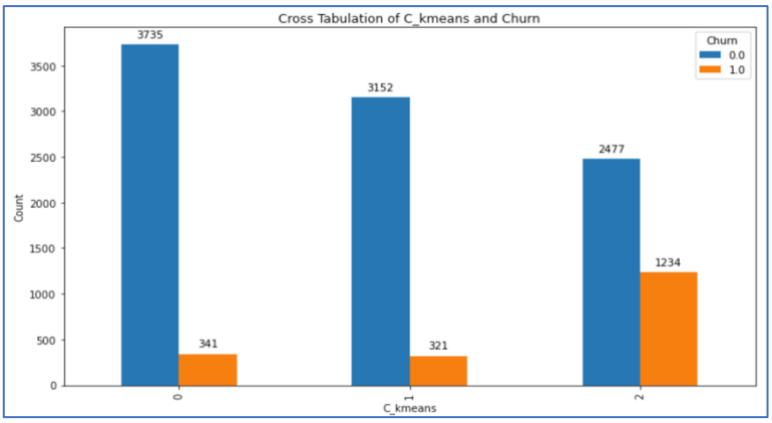

Figure 31: Cross tabulation of C_kmeans & churn

The "Cluster Churn Analysis Bar Chart" tells us interesting things about how customers behave. It shows that Cluster 2 customers might stop using our services more often, which is a concern. But for Clusters 0 and 1, customers are not likely to stop using our services; they're loyal to us. So, we need to pay more attention to Cluster 2 and find ways to keep those customers. At the same time, we should keep treating customers from Clusters 0 and 1 well to make sure they stay happy with us. This helps us prevent customers from leaving and keeps them loyal.

4. Model building: (Clear on why was a particular model(s) chosen. - Effort to improve model performance.)

a. Train and Split and its importance in model building

The train-test split is a simple but crucial step that helps us build better predictive models by ensuring it can handle unseen data effectively.

The train-test split is a vital process where data is divided into training and test sets. The training set (70%) teaches the model patterns, while the test set (30%) evaluates its performance on new data.

```
Shape of X_train: (7882, 17)
Shape of X_test: (3378, 17)
Shape of y_train: (7882,)
Shape of y_test: (3378,)
```

- X_train (7882, 17): Training dataset with 7,882 rows and 17 features for each, used to teach the model patterns.
- X_test (3378, 17): Test dataset with 3,378 rows and 17 features, unseen by the model, for evaluating its performance.
- y_train (7882,): Target variable for training samples, guiding the model's learning during training.
- y_test (3378,): Target variable for test samples, used to measure how well the model's predictions match actual outcomes on unseen data.

b. Performance Various Models: (Classification Report and AUC score)

In order to gauge the effectiveness of various classification models, we have conducted a comprehensive evaluation using classification reports and Area Under the Curve (AUC) scores. These metrics provide valuable insights into the models' capabilities in distinguishing between different classes and their overall predictive power. By meticulously analysing precision, recall, F1-scores, and AUC scores, we gain a clearer understanding of each model's strengths and weaknesses. This assessment aids us in making informed decisions on which models are best suited for addressing the specific business problem at hand.

- "0" indicates a prediction of non-churn.
- "1" indicates a prediction of churn.

Models	Training Dataset (70%)								Testing Dataset (30)%							
Pefromance Metrics	Precision (0)	Recall (0)	F1-score (0)	Precision (1)	Recall (1)	F1-score (1)	Accuracy	AUC Score	Precision (0)	Recall (0)	F1-score (0)	Precision (1)	Recall (1)	F1-score (1)	Accuracy	AUC Score
Logistic Regression (Basic)	0.9	0.96	0.93	0.73	0.48	0.58	0.88	0.88	0.91	0.96	0.93	0.74	0.51	0.61	0.89	0.87
Logistic Regression (with hyper-tuning)	0.9	0.97	0.93	0.74	0.48	0.58	0.89	0.88	0.91	0.97	0.94	0.75	0.51	0.61	0.89	0.87
Logistic Regression (SMOTE)	0.83	0.82	0.83	0.82	0.84	0.83	0.83	0.89	0.95	0.83	0.89	0.49	0.8	0.6	0.82	0.87
LDA (Basic)	0.91	0.96	0.93	0.71	0.53	0.61	0.88	0.88	0.91	0.96	0.93	0.72	0.56	0.63	0.89	0.87
LDA (with Hyper-tuning)	0.91	0.96	0.93	0.71	0.53	0.61	0.88	0.88	0.91	0.96	0.93	0.71	0.55	0.62	0.89	0.87
LDA (SMOTE)	0.84	0.82	0.83	0.82	0.84	0.83	0.83	0.89	0.95	0.82	0.88	0.48	0.8	0.6	0.82	0.87
KNN Model (Basic)	0.97	0.99	0.98	0.95	0.87	0.91	0.97	0.99	0.95	0.98	0.97	0.9	0.74	0.81	0.94	0.96
KNN Model (with Hyper-tuning)	1	1	1	1	1	1	1	1	0.97	0.99	0.98	0.92	0.82	0.87	0.96	0.96
KNN Model (SMOTE)	1	0.95	0.97	0.95	1	0.97	0.97	1	0.99	0.93	0.95	0.72	0.93	0.81	0.93	0.97
Naïve Bayes Model (Basic)	0.91	0.92	0.92	0.59	0.55	0.57	0.86	0.84	0.91	0.92	0.91	0.58	0.54	0.56	0.86	0.83
Naïve Bayes Model (with Hyper-tuning)	0.91	0.92	0.92	0.59	0.55	0.57	0.86	0.84	0.91	0.92	0.91	0.58	0.54	0.56	0.86	0.83
Naïve Bayes Model (SMOTE)	0.79	0.78	0.79	0.78	0.79	0.79	0.79	0.84	0.94	0.78	0.85	0.41	0.74	0.53	0.78	0.83
RandomForestClassifier (Basic)	1	1	1	1	1	1	1	1	0.96	0.99	0.98	0.95	0.81	0.87	0.96	0.98
RandomForestClassifier (with Hyper-tuning)	0.95	1	0.97	0.98	0.75	0.85	0.96	0.99	0.93	0.99	0.96	0.92	0.65	0.76	0.93	0.96
RandomForestClassifier (SMOTE)	1	1	1	1	1	1	1	1	0.97	0.98	0.98	0.9	0.85	0.88	0.96	0.98
Bagging (Basic)	1	1	1	1	0.98	0.99	1	0.99	0.95	0.99	0.97	0.92	0.76	0.84	0.95	0.88
Bagging (with Hyper Tuning)	1	1	1	1	0.98	0.99	1	0.99	0.94	0.99	0.97	0.95	0.68	0.8	0.94	0.84
Bagging (SMOTE)	1	1	1	1	0.99	0.99	1	0.99	0.96	0.97	0.97	0.86	0.82	0.84	0.95	0.9
Ada Boost (Basic)	0.91	0.96	0.93	0.72	0.54	0.62	0.89	0.91	0.91	0.96	0.93	0.73	0.54	0.62	0.89	0.9
Ada Boost (with Hyper Tuning)	0.91	0.96	0.94	0.73	0.55	0.63	0.89	0.91	0.91	0.96	0.94	0.74	0.55	0.63	0.89	0.9
Ada Boost (SMOTE)	0.88	0.88	0.88	0.88	0.88	0.88	0.88	0.95	0.94	0.89	0.92	0.58	0.74	0.65	0.87	0.89
Gradient Boosting (Basic)	0.92	0.97	0.95	0.81	0.6	0.69	0.91	0.94	0.92	0.97	0.94	0.79	0.58	0.67	0.9	0.92
Gradient Boosting (with Hyper-tuning)	1	1	1	1	0.98	0.99	1	1	0.96	0.99	0.97	0.93	0.79	0.85	0.95	0.98
Gradient Boosting (SMOTE)	0.92	0.92	0.92	0.92	0.92	0.92	0.92	0.97	0.94	0.93	0.94	0.67	0.71	0.69	0.89	0.91
Support vector Machine (Basic)	0.92	0.98	0.95	0.86	0.6	0.71	0.92	0.94	0.92	0.98	0.95	0.85	0.58	0.69	0.91	0.91
Support vector Machine (with Hyper-tuning)	0.96	0.99	0.98	0.96	0.81	0.88	0.96	0.99	0.94	0.98	0.96	0.9	0.71	0.79	0.94	0.95
Support vector Machine (SMOTE)	0.94	0.9	0.92	0.91	0.94	0.92	0.92	0.97	0.96	0.9	0.93	0.62	0.83	0.71	0.88	0.94

Table 11: Performance Various Models: (Classification Report and AUC score)

c. Confusion Matrix Analysis for Model Comparison:

The confusion matrix offers a deeper understanding of model performance by presenting a comprehensive breakdown of prediction outcomes. This matrix allows us to discern true positives, true negatives, false positives, and false negatives for each model. By interpreting these values, we can identify patterns of correct and incorrect predictions made by the models. This analysis assists us in not only quantifying the effectiveness of different models but also in identifying potential areas of improvement. Through this examination, we can make more informed decisions regarding the selection and refinement of models for addressing our business objectives.

Models	Training Dataset (70%)				Testing Dataset (30%)			
Confusion Matrix - Metrics	True Negative (TN)	Type II Error (False Negative)	Type I Error (False Positive)	True Positive (TP)	True Negative (TN)	Type II Error (False Negative)	Type I Error (False Positive)	True Positive (TP)
Logistic Regression (Basic)	6324	232	686	640	2705	103	278	292
Logistic Regression (with hyper-tuning)	6336	220	690	636	2712	96	281	289
Logistic Regression (SMOTE)	5366	1190	1081	5475	2325	483	114	456
LDA (Basic)	6270	286	626	700	2682	126	252	318
LDA (with Hyper-tuning)	6270	286	625	701	2682	126	254	316
LDA (SMOTE)	5383	1173	1058	5498	2307	501	113	457
KNN Model (Basic)	6494	62	171	1155	2764	44	151	419
KNN Model (with Hyper-tuning)	6556	0	0	1326	2769	39	100	470
KNN Model (SMOTE)	6214	342	8	6548	2601	207	39	531
Naïve Bayes Model (Basic)	6048	508	596	730	2586	222	260	310
Naïve Bayes Model (with Hyper-tuning)	6048	508	596	730	2586	222	260	310
Naïve Bayes Model (SMOTE)	5133	1423	1380	5176	2198	610	146	424
RandomForestClassifier (Basic)	6556	0	0	1326	2786	22	119	451
RandomForestClassifier (with Hyper-tuning)	6538	18	331	995	2774	34	201	369
RandomForestClassifier (SMOTE)	6556	0	0	6556	2757	51	86	484
Bagging (Basic)	6554	2	29	1297	2765	43	133	437
Bagging (with Hyper Tuning)	6556	0	20	1306	2789	19	181	389
Bagging (SMOTE)	6549	7	12	1314	2720	88	109	461
Ada Boost (Basic)	6285	271	612	714	2694	114	265	305
Ada Boost (with Hyper Tuning)	6291	265	601	725	2699	109	255	315
Ada Boost (SMOTE)	5766	790	810	5746	2502	306	150	420
Gradient Boosting (Basic)	6368	188	531	795	2722	86	238	332
Gradient Boosting (with Hyper-tuning)	6555	1	22	1304	2776	32	121	449
Gradient Boosting (SMOTE)	6060	496	543	6013	2612	196	165	405
Support vector Machine (Basic)	6428	128	525	801	2749	59	241	329
Support vector Machine (with Hyper-tuning)	6510	46	249	1077	2761	47	166	404
Support vector Machine (SMOTE)	5925	631	408	6148	2514	294	99	471

Table 12: Confusion Matrix Analysis for Model Comparison:

d. Interpretation of performance of metrics of Various model.

i. Interpretation of the Logistic Regression Model: Basic

The model's recall for predicting churn (class 1) is 48% in training and slightly better at 51% in the test data, indicating its performance in correctly identifying churn cases. Precision, reflecting accurate positive predictions, is 73% in training and 74% in testing, showing good alignment between predicted and actual churn. The F1-score, balancing precision and recall, is 0.58 for training and 0.61 for testing, indicating a fair overall balance.

Examining confusion matrices, false negatives and false positives are present, highlighting areas for improvement. In training, 686 actual churners were wrongly predicted as non-churners (false negatives), and 232 were inaccurately predicted to churn (false positives). Similarly, testing had 278 false negatives and 103 false positives.

AUC values of 0.88 (training) and 0.87 (testing) affirm the model's ability to identify potential churners. The model pinpoints three important factors: recent complaints, tenure, and single marital status, influencing customer outcomes.

In summary, the model shows consistent performance with decent precision, but its lower recall suggests room for improvement, making it a good, but not exceptional, model.

ii. Interpretation of the Logistic Regression Model: SMOTE

The Logistic Regression Model with SMOTE demonstrates balanced performance on both training and test data. In the training set, precision and recall are well-matched for both classes, resulting in an 83% F1-score and 83% accuracy. Similarly, the test set shows high precision (95%) and recall (83%) for class 0, yielding an 89% F1-score. Class 1 in the test set has 49% precision, 80% recall, and a 60% F1-score, with an overall test accuracy of 82%.

Confusion matrices highlight accurate predictions for true positives and true negatives, but also misclassifications. Training data misclassifies 1081 non-churners as churners (false positives) and 1190 churners as non-churners (false negatives). In the test data, 114 non-churners are falsely identified as churners (false positives), and 483 churners as non-churners (false negatives).

AUC scores are 0.89 for the test and 0.87 for training, showcasing the model's ability to identify potential churners. The model emphasizes the importance of recent complaints, tenure, single marital status, and monthly revenue in influencing customer outcomes.

In summary, the Logistic Regression Model with SMOTE achieves balanced performance, though slightly lower precision for class 1 predictions in the test data suggests room for improvement.

iii. Interpretation of Linear Discriminant Analysis: Basic

The LDA model's performance is evident from the classification reports on training and test data. For churned customers (class 1), the model achieves precision of 71% in training and 72% in testing, reflecting accurate positive predictions. Recall for class 1 is 53% in training and 56% in testing, with F1-scores of 0.61 and 0.63 respectively. For non-churned customers (class 0), the model's precision and recall are 91% and 96% in both training and testing, resulting in F1-scores of 0.93.

The confusion matrices show misclassification of churners (629 false negatives) and non-churners (286 false positives) in training. In testing, 252 churners and 126 non-churners are misclassified.

AUC scores of 0.88 for training and 0.87 for testing confirm the model's ability to identify potential churners. Customer complaints, tenure, single marital status, and monthly revenue are key factors influencing outcomes.

In conclusion, the LDA model performs reasonably well in identifying potential churners and retaining customers, yet there's room for improvement.

iv. Interpretation of Linear Discriminant Analysis: SMOTE

The LDA model with SMOTE's performance is captured by the classification reports on training and test data. For churned customers (class 1), the model achieves precision of 82% in training and 48% in testing, with recall at 84% and 80% respectively, yielding F1-scores of 0.83 and 0.60. For non-churned customers (class 0), the model's precision and recall are 84% in both training and testing, resulting in F1-scores of 0.83.

The confusion matrices indicate misclassifications of 1058 churners and 1173 non-churners in training, and 113 churners and 501 non-churners in testing. AUC scores of 0.89 for training and 0.87 for testing showcase the model's effectiveness in identifying potential churners. Key features influencing outcomes are customer tenure and complaints.

In summary, the model adeptly differentiates between churn and non-churn cases. It accurately captures both customer categories, although there's room for improvement in pinpointing potential churners more precisely.

v. Interpretation of KNN Model: Basic

The KNN model shows strong performance in identifying customers who stay (class 0) with high precision (97%) and satisfactory recall (87%) for potential churners (class 1) in the training data. The impressive overall accuracy of 97% and weighted F1-score of 0.97 underline its balance between precision and recall. On the test data, the model maintains high precision (95%) for class 0 and respectable recall (74%) for class 1, resulting in a strong overall accuracy of 94% and a weighted F1-score of 0.94.

Analysing the confusion matrices, the model misclassified 171 churners as non-churners (false negatives) and 62 non-churners as churners (false positives) in the training data. In the test data, it misclassified 151 churners and 44 non-churners. AUC scores of 0.99 (training) and 0.96 (testing) emphasize its ability to distinguish between classes.

In summary, the KNN model performs well in identifying staying customers and detecting potential churners. While recall for class 1 could improve, the model's balanced precision and recall, along with strong AUC scores, suggest solid overall performance.

vi. Interpretation of KNN Model: SMOTE

The model's performance evaluation on both training and test data reveals exceptional precision for both classes. For class 0 (customers who stay), precision is perfect at 100%, while for class 1 (potential churners), it's high at 95% in training and 72% in testing.

Although precision is impressive, class 1 recall in the test data is slightly lower at 93%, leaving room for improvement in identifying potential churners. The model maintains strong accuracy of 97% in training and 93% in testing, with well-balanced weighted F1-scores.

Looking at confusion matrices, few cases of misclassification exist, with slightly more class 0 predictions being mistaken for class 1. AUC scores of 1.00 (training) and 0.97 (testing) highlight the model's ability to distinguish between classes.

In summary, the model demonstrates remarkable precision and overall performance. Despite slightly lower class 1 recall in the test data, the model remains effective and balanced, without signs of overfitting or underfitting.

vii. Interpretation of the Naïve Bayes Model: Basic

The model's evaluation on both training and test data balances precision and recall for different classes. For class 0 (customers who stay), precision is strong at 91% in both training and testing, indicating accurate identification of non-churn cases. Precision for class 1 (potential churners) drops to 59% in training and 58% in testing, with notable false positive predictions. Recall for class 1 is 55% in training and 54% in testing, showing potential for improvement in identifying customers who might leave.

Overall accuracy remains consistent at 86% for both training and testing. Weighted F1-scores blend precision and recall. Confusion matrices reveal misclassification between classes. In training, 508 class 0 cases are incorrectly predicted as class 1, and 596 class 1 cases are predicted as class 0. In testing, 222 class 0 cases are mistakenly predicted as class 1, and 260 class 1 cases as class 0. AUC scores of 0.84 for training and 0.83 for testing reflect the model's ability to distinguish between classes, with room for improvement.

In summary, the Naïve Bayes Model showcases balanced precision and recall, with scope to enhance class 1 precision. The model's accuracy is fair, with potential to better capture actual churn cases. AUC scores suggest moderate discriminatory ability without clear signs of overfitting or underfitting. The model's performance underscores the importance of improving its ability to classify churn cases accurately while minimizing false positives.

viii. Interpretation of Naïve Bayes Model: SMOTE

The classification reports assess the model's performance on training and test datasets. For class 0 (customers who remain), precision is consistent, with 79% in training and a higher 94% in testing, indicating accurate identification. Class 1 (potential churners) has balanced precision of 78% in training and 41% in testing. Recall, measuring actual class 1 capture, is around 79% in training and 74% in testing, with room for enhancement. Overall accuracy is 79% in training and 78% in testing. Weighted F1-scores balance precision and recall.

Confusion matrices show specific misclassifications. In training, 1423 class 0 cases were wrongly predicted as class 1, and 1380 class 1 cases as class 0. In testing, 610 class 0 cases were misclassified as class 1 and 146 class 1 cases as class 0.

AUC scores of 0.84 for training and 0.83 for testing reveal class distinction ability.

In summary, the model consistently identifies customers who stay (class 0), with higher testing precision. Enhancements are needed for recognizing potential churners (class 1) and balanced precision-recall. The model's accuracy is reasonable, demanding further regularisation.

ix. Interpretation of the Random Forrest Model: Basic

The assessment of the Random Forest Model's performance on training and test datasets reveals exceptional precision scores for both classes. For class 0 (customers who stay), precision remains consistently high at 100% in training and 96% in testing. Similarly, for class 1 (potential churners), the precision scores are impressive, reaching 100% in training and 95% in testing. However, recall values for class 1 are slightly lower, measuring 76% in testing and a perfect 100% in training. This suggests room for improvement in identifying all potential churners.

The overall accuracy of the model is robust, achieving 100% on training and 96% on testing. The weighted F1-scores balance precision and recall. AUC scores of 1.00 for testing and 0.99 for training underscore the model's effectiveness.

The matrix accurately distinguishes classes but with a few churn prediction errors: 22 class 0 instances mislabelled as class 1 and 119 class 1 instances misidentified as class 0.

In summary, the Random Forest Model demonstrates remarkable precision for both classes, with scope to enhance class 1 recall. The model's accuracy, AUC scores, and influential features (Tenure, Cashback) contribute to its strong performance.

x. Interpretation of the Random Forrest Model: SMOTE

The Random Forest Model with SMOTE excels in precision and recall for both classes. In training, it achieves 100% precision, recall, and F1-scores for both classes, which raises concerns about overfitting due to perfect scores on the training data. Moving to testing, precision remains high at 97% for class 0 and 90% for class 1. Class 1 recall slightly decreases to 85%, still capturing actual churn cases well. The test accuracy is an impressive 96%.

In confusion matrices, training data has perfect classification, while test data shows minor misclassifications: 51 class 0 instances predicted as class 1 and 86 class 1 instances as class 0. AUC scores are 1.00 for test data and 0.98 for training data, highlighting strong discriminatory ability. Key features: Tenure and Complaints Ly play pivotal roles.

To sum up, the Random Forest Model with SMOTE demonstrates remarkable predictive capabilities, but the perfect scores on training data suggest potential overfitting.

xi. Interpretation of the Support Vector Machines – SVM Model: Basic

The SVM Model shows balanced performance on training and test datasets. For class 0, it maintains strong precision and recall (92% and 98% respectively) on both. Class 1's precision (86%) and recall (60%) suggest some missed churn cases. Accuracy is solid at 92%.On the test set, class 0 precision (92%) and recall (98%) remain robust. Class 1's precision (85%) and recall (58%) indicate room for improvement. Test accuracy is 91%.

In training, the model correctly classifies 801 class 1 instances but misclassifies 525 class 0 instances as class 1 and 128 class 1 instances as class 0. In testing, it identifies 329 class 1 cases correctly but misclassifies 241 class 0 cases as class 1 and 59 class 1 cases as class 0. SVM's AUC scores are 0.94 for training and 0.91 for testing, demonstrating effective class separation.

In summary, the SVM Model showcases balanced performance and notable accuracy on both datasets. While effective, there's potential to enhance identifying potential churn cases. Its AUC scores suggest its capability, and it avoids overfitting or underfitting indications.

xii. Interpretation of the Support Vector Machines – SVM Model: SMOTE

The SVM model with SMOTE demonstrates balanced performance on training and test data. For class 0, precision and recall are strong at 94% and 90% in training, and 96% and 90% in testing. For class 1, precision and recall are 91% and 94% in training, and 62% and 83% in testing, indicating improved identification of churn cases.

In the confusion matrix for training, it correctly predicts 5925 class 0 and 6148 class 1 instances. It misclassifies 631 class 1 instances as class 0 and 408 class 0 instances as class 1. In the test set, it predicts 2514 class 0 and 471 class 1 instances accurately. However, it wrongly predicts 294 class 1 instances as class 0 and 99 class 0 instances as class 1.

Overall accuracy remains consistent at 92% in training and 88% in testing. Weighted F1-scores reflect the balance between precision and recall. Strong AUC scores of 0.97 for training and 0.94 for testing confirm its ability to distinguish between classes.

In summary, the SVM model with SMOTE achieves balanced performance, addressing data imbalance for improved recall. The model's accuracy and AUC scores showcase its strong performance in classification tasks.

Model Tuning:

Model tuning, also known as hyperparameter tuning, is a critical step in model building that involves adjusting various parameters of a machine learning algorithm to optimize its performance and achieve the best possible results. Different hyperparameter values can significantly impact a model's performance. Tuning allows you to fine-tune these parameters to achieve higher accuracy.

xiii. Interpretation of the Logistic Regression with Hyper – Tuning:

Optimizing a logistic regression model involves crucial hyperparameters such as 'penalty' (L1 or L2), 'C' (regularization strength), 'solver' (optimization algorithm), 'class_weight' (for imbalance), 'max_iter' (iterations), and 'dual' (problem type). Tuning these enhances performance, controlling overfitting and addressing class imbalances.

```
GridSearchCV(cv=5, estimator=LogisticRegression(random_state=1),
             param_grid={'C': [0.01, 0.1, 1.0, 10.0],
                         'class_weight': [None, 'balanced'],
                         'dual': [False, True], 'max_iter': [100, 200, 500],
                         'penalty': ['l1', 'l2'],
                         'solver': ['liblinear', 'lbfgs', 'newton-cg', 'saga']},
             scoring='accuracy')
```

Table 13: Hyper-tuning parameter of Logistic Regression

The below shows the best-performing logistic regression model based on these parameters:

```
LogisticRegression(C=0.1, penalty='l1', random_state=1, solver='saga')
```

The hyper-tuned Logistic Regression model excels at classification, demonstrating balanced precision but differing recall rates. In training, it achieves 90% precision for class 0 and 73% for class 1, while recall is 96% for class 0 and 48% for class 1. Similar patterns emerge in the test data. The confusion matrix reveals accurate predictions for class 0 (6324 instances in training, 2705 instances in testing), but misclassifications for class 1 (232 instances in training, 103 instances in testing). AUC scores of 0.88 (training) and 0.87 (testing) imply moderate discriminatory capacity. "Tenure" and "Complain Ly" prove significant features. The model's strengths and weaknesses suggest room for refining churn prediction.

xiv. Interpretation of the LDA Model with Hyper – Tuning:

GridSearchCV optimizes the Linear Discriminant Analysis (LDA) model by tuning 'shrinkage' (covariance matrix regularization) and 'solver' (solution method) hyperparameters. It systematically explores combinations to enhance accuracy and classification efficiency.

```
GridSearchCV(cv=5, estimator=LinearDiscriminantAnalysis(),
             param_grid={'shrinkage': [None, 'auto', 0.0, 0.5, 1.0],
                         'solver': ['svd', 'lsqr', 'eigen']},
             scoring='accuracy')
```

Table 14: Hyper tuning Parameter of LDA Model

The below shows the best-performing LDA model based on these parameters:

```
LinearDiscriminantAnalysis(shrinkage='auto', solver='lsqr')
```

The Hyper-Tuned LDA model's performance is assessed using precision, recall, and metrics. In the training set, it achieves 91% precision for class 0 and 71% for class 1, with class 1 recall at 53%. Class 0 has a higher recall at 96%. Similarly, in the test set, the LDA model maintains a balanced performance with 91% precision for class 0 and 71% for class 1. The improved recall of 53% for class 1 suggests the model's capability to recognize these instances. Class 0, on the other hand, demonstrates a recall of 96%.

In the training data, the confusion matrix reveals 6270 true negatives and 701 true positives. However, there were 286 false positives and 625 false negatives. Similarly, in the test data, the model correctly predicted 2682 instances as class 0 and 316 instances as class 1, but misclassified 126 instances of class 1 as class 0 and 254 instances of class 0 as class 1.

In summary, the Hyper-Tuned LDA model demonstrates satisfactory precision and recall for both classes, while its performance on class 1 could benefit from further improvement.

xv. Interpretation of the KNN Model with Hyper Tuning:

GridSearchCV optimizes the K-Nearest Neighbors (KNN) Classifier through systematic testing of hyperparameter combinations. This includes algorithm, Neighbors count, and weighting method, aiming to boost accuracy. Cross-validation with 5 subsets is employed to fine-tune the model. The process identifies optimal hyperparameter values for improved predictions.

```
GridSearchCV(cv=5, estimator=KNeighborsClassifier(),
             param_grid={'algorithm': ['auto', 'ball_tree', 'kd_tree', 'brute'],
                         'n_neighbors': [3, 5, 7, 9, 11, 13, 15],
                         'weights': ['uniform', 'distance']},
             scoring='accuracy')
```

Table 15: Hyper tuning parameter of KNN Model

The below shows the best-performing KNN model based on these parameters:

```
KNeighborsClassifier(n_neighbors=3, weights='distance')
```

The Hyper-Tuned K-Nearest Neighbors (KNN) model demonstrates outstanding performance on both training and test data. During training, it achieves flawless 100% precision and recall for both classes. In the test phase, the model sustains impressive precision rates of 97% for class 0 and 92% for class 1, with a notable 82% recall for class 1 and 99% for class 0.

The confusion matrix for training data shows perfect predictions, with 6556 true negatives and 1326 true positives. In the test data, it accurately predicts 2769 true negatives and 470 true positives, but misclassifies 39 instances as false positives and 100 instances as false negatives. A high AUC score of 1.00 for training and 0.96 for testing emphasizes its strong classification ability and generalization.

However, it's important to note that the model's perfect performance on training data suggests potential overfitting concerns, requiring further investigation for practical deployment.

xvi. Inference of Naive Bayes with Hyper tuning parameter:

Hyperparameter tuning is usually not extensively done for Naive Bayes models as compared to other algorithms. This is because Naive Bayes has only a few hyperparameters, and its performance is generally consistent. Whether using Naive Bayes with or without hyper tuning, the metric outcomes remain similar. Even when incorporating the "var_smoothing" parameter, the results remain unchanged. As a result, hyperparameter tuning is not considered for Naive Bayes models.

```
GridSearchCV(cv=5, estimator=GaussianNB(),
             param_grid={'var_smoothing': [1e-09, 1e-08, 1e-07, 1e-06, 1e-05]},
             scoring='accuracy')
```

Table 16: Hyper tuning parameter of Naive Bayes Model

xvii. Inference of Random forest with Hyper-tuning.

GridSearchCV was utilized to optimize the Random Forest Classifier for business purposes. It systematically explored hyperparameters such as criteria for node splitting ('gini' or 'entropy'), tree depth (5 to 10), minimum samples split (8 to 10), number of estimators (100 to 300), and fixed random state (1). This approach enhances model accuracy through cross-validation and efficient CPU utilization. The goal is to fine-tune the model for precise and reliable classification, addressing business needs effectively.

```
GridSearchCV(cv=5, estimator=RandomForestClassifier(), n_jobs=-1,
             param_grid={'criterion': ['gini', 'entropy'],
                         'max_depth': [5, 6, 7, 8, 9, 10],
                         'min_samples_split': [8, 9, 10],
                         'n_estimators': [100, 200, 300], 'random_state': [1]},
             scoring='accuracy')
```

Table 17: Hyper Tuning Parameter of RF Model

The below shows the best-performing Random Forrest Model based on these parameters:

```
Best Estimator: RandomForestClassifier(max_depth=10, min_samples_split=8, n_estimators=300,
                       random_state=1)
```

The Hyper-Tuned Random Forest model demonstrates strong precision and accuracy in both training and test datasets. It achieves 95% precision for class 0 and an impressive 98% for class 1 in training, while class 1 recall is 75%, suggesting potential for improvement in capturing churn cases. In the test set, precision is balanced at 93% for class 0 and 92% for class 1, with class 1 recall at 65%. Class 0 recall remains high at 99%.

Confusion matrices reveal accurate predictions but also misclassifications. In training, it predicts 6538 class 0 instances and 995 class 1 instances correctly, but misclassifies 18 as class 1 and 331 as class 0. In testing, it accurately predicts 2774 class 0 instances and 369 class 1 instances, with 34 false positives and 201 false negatives.

The AUC scores of 0.99 for training and 0.96 for testing highlight the model's strong classification ability. Key features like Tenure and Cashback contribute to its performance. Despite its strengths, overfitting may be a concern due to higher performance on training data.

In conclusion, the Hyper-Tuned Random Forest model excels in precision and accuracy, with potential for improvement in class 1 recall. Its significant features and strong performance position it well for churn prediction.

xviii. Interpretation of the SVM Model - Hyper tuning:

To optimize the Support Vector Machines (SVM) model, a parameter grid was defined, encompassing 'C' values of 0.1, 1, and 10, 'kernel' options of 'linear', 'rbf', and 'poly' with degrees 2, 3, and 4, and 'gamma' variations of 'scale' and 'auto'. This methodical approach aimed to improve the SVM model's churn prediction accuracy.

```
GridSearchCV(cv=5, estimator=SVC(),
             param_grid={'C': [0.1, 1, 10], 'degree': [2, 3, 4],
                         'gamma': ['scale', 'auto'],
                         'kernel': ['linear', 'rbf', 'poly']},
             scoring='accuracy')
```

Table 18: Hyper Tuning Parameter - SVM Model

The below shows the best-performing SVM Model based on these parameters:

```
{'C': 10, 'degree': 2, 'gamma': 'scale', 'kernel': 'rbf'}
```

The Hyper-Tuned SVM model's performance is evaluated using precision, recall, and metrics. In training, it achieves 96% precision for class 0 and 96% for class 1, with class 1 recall at 81%. Transitioning to the test set, precision remains balanced at 94% for class 0 and 90% for class 1, and class 1 recall is 71%.

The confusion matrices reveal that in the training data, the model accurately predicted 6510 instances as class 0 and 1077 instances as class 1. However, it made 46 false predictions for class 1 and 249 false predictions for class 0.Similarly, in the test data, the model accurately predicted 2761 instances as class 0 and 404 instances as class 1. Nonetheless, there were 47 instances wrongly classified as class 1 and 166 instances inaccurately predicted as class 0.

The AUC scores of 0.99 for training and 0.95 for testing underscore the model's effectiveness in classifying instances, reflecting its strong performance.

In summary, the Hyper-Tuned SVM model demonstrates robust precision and accuracy in classifying instances. Despite its strong performance, addressing the relatively lower recall for class 1 could further enhance its predictive capabilities, making it a promising model for churn prediction.

Ensemble techniques:

Ensemble techniques in machine learning involve combining multiple individual models (base or weak learners) to create a more powerful predictive model. The aim is to enhance accuracy, stability, and generalization by leveraging the strengths of different models while minimizing their weaknesses. Examples of ensemble techniques include Bagging, AdaBoost, and Gradient Boosting.

xix. Interpretation of the Bagging Model:

The classification report provides a comprehensive overview of our model's performance on both the training and test datasets. In the training data, our model achieves exceptional precision and recall for both classes: 100% precision for class 0 and 100% precision for class 1. The recall for class 0 is also perfect at 100%, while class 1 recall is strong at 98%. Transitioning to the test data, the model maintains high-quality performance with 95% precision for class 0 and 91% for class 1. Notably, class 1 recall is 77%, highlighting its ability to capture actual churn instances, while class 0 recall remains high at 98%.

The confusion matrices provide insights. In training, the model accurately predicted 6554 instances as class 0 and 1297 instances as class 1. However, there were 2 instances wrongly classified as class 1 and 29 instances inaccurately predicted as class 0. In the test data, it accurately predicted 2765 instances as class 0 and 437 instances as class 1, with 43 instances misclassified as class 1 and 133 instances inaccurately predicted as class 0.

AUC scores of 0.99 for training and 0.88 for testing underscore its effectiveness in classifying instances, with potential for test set generalization improvement. Key features contributing to performance include Tenure and Cashback, reinforcing their significance in churn prediction.

In summary, the model shows strong performance overall, with high precision and recall for class 0. However, its recall for class 1 at 77% indicates potential for improvement in capturing churn cases while addressing the overfitting observed in the model.

xx. Interpretation of the Bagging Model with Hyper Tuning:

Bagging ensemble technique is fine-tuned using hyperparameters. We explore different combinations of sampling ratios, feature proportions, and the number of estimators (base models). These parameters aim to enhance the predictive performance of our model. The 'random_state' parameter ensures consistent results. This process allows us to create an optimized Bagging model that could potentially improve our predictions for better decision-making.

The below shows the best-performing Bagging model based on these parameters:

```
Best Parameters: {'max_features': 0.6, 'max_samples': 0.7, 'n_estimators': 30, 'random_state': 1}
Best Estimator: BaggingClassifier(base_estimator=DecisionTreeClassifier(), max_features=0.6,
                  max_samples=0.7, n_estimators=30, random_state=1)
```

The Bagging model, fine-tuned using hyperparameters, demonstrates strong performance. In the training set, it achieves impeccable precision for both classes: 100% for class 0 and 100% for class 1. Class 1 recall is notably high at 98%, and class 0 recall remains perfect at 100%.

Transitioning to the test set, the model maintains robustness with 94% precision for class 0 and 95% for class 1. However, class 1 recall decreases to 68%, indicating room for improvement in identifying actual churn cases. Class 0 recall remains strong at 99%.

In the confusion matrices for training data, the model accurately predicted 6556 instances as class 0 (true negatives) and 1306 instances as class 1 (true positives). In the test data, it accurately predicted 2789 instances as class 0 and 389 instances as class 1. However, there were 20 false predictions for class 1 and 181 false predictions for class 0 in the test data.

AUC scores of 0.99 for training and 0.84 for testing underscore the model's classification ability, suggesting potential for refinement.

In summary, the Bagging model, with hyperparameter tuning, showcases strong precision and accuracy. While performing well in predicting class 0 instances, there is room for enhancement in capturing class 1 cases, as reflected by the comparatively lower recall.

xxi. Interpretation of the Bagging Model with SMOTE:

The Bagging Model with Synthetic Minority Over-sampling Technique (SMOTE) showcases robust performance in both the training and test datasets. In the training set, it demonstrates 100% precision for class 0 and 99% precision for class 1, with perfect recall for class 0 and a strong 99% recall for class 1. Transitioning to the test set, the model maintains high performance with 96% precision for class 0 and 84% for class 1. The recall for class 0 is 97%, while class 1 recall is at 81%, capturing a significant portion of actual churn cases.

In the confusion matrices, the training data shows 6549 instances correctly predicted as class 0 and 1314 instances as class 1. In the test data, the model accurately predicts 2720 instances as class 0 and 461 instances as class 1. However, there are 88 instances wrongly classified as class 1 and 109 instances inaccurately predicted as class 0. The high AUC score of 0.99 for training and 0.89 for testing reflects the model's strong classification ability.

In summary, the Bagging Model with SMOTE achieves impressive precision and recall. While class 0 performance is near perfect, the model's recall for class 1 in the test set is slightly lower at 81%, indicating potential for capturing churn.

xxii. Interpretation of the Ada Boost Model:

The Ada Boost model's performance is evaluated using precision, recall, and other metrics. In the training set, it achieves 91% precision for class 0 and 72% for class 1. Class 0 recall is 96%, while class 1 recall is 54%, indicating potential for improvement in capturing actual churn cases.

Transitioning to the test set, the pattern continues with 91% precision for class 0 and 73% for class 1. Class 0 recall is 96%, and class 1 recall is 54%. The confusion matrix for training shows 6285 true negatives and 714 true positives, with 271 false positives and 612 false negatives. Similarly, in the test data, 2694 true negatives and 305 true positives, with 114 false positives and 265 false negatives.

AUC scores of 0.91 for training and 0.90 for testing underscore the model's effectiveness. Key features like Tenure and Cashback contribute to the model's performance in predicting churn.

The Ada Boost model's performance is mixed, with high precision for class 0 but relatively lower precision for class 1. It maintains a balanced F1-score and reasonable AUC scores, highlighting effectiveness. However, there's room for improvement, particularly in enhancing recall for class 1.

xxiii. Interpretation of the Ada Boost Model with Hyper Tuning:

The Ada Boost model with hyperparameter tuning involves optimizing its performance by adjusting key parameters. The parameters being tuned are the number of estimators (decision trees) used for boosting, which can be 50, 100, or 200, and the learning rate, which influences the contribution of each model to the final prediction and can be 0.01, 0.1, or 1.0. Hyperparameter tuning aims to enhance the model's predictive capability and achieve better results in churn prediction.

The below shows the best-performing Ada Boost model based on these parameters:

```
Best Parameters: {'learning_rate': 1.0, 'n_estimators': 100}
Best Estimator: AdaBoostClassifier(n_estimators=100, random_state=1)
```

The hyper-tuned Ada Boost model's performance is evaluated using precision, recall, and other metrics. In the training set, it achieves higher precision for class 0 (91%) and relatively lower precision for class 1 (73%). The recall for class 0 is 96%, while for class 1, it is 55%, indicating potential improvement in capturing actual churn cases.

Similar trends continue in the test set, with 91% precision for class 0 and 74% for class 1. The recall for class 0 is 96%, and for class 1, it's 55%. The confusion matrix for training shows 6291 true negatives and 725 true positives, with 265 false positives and 601 false negatives. In the test data, 2699 true negatives and 315 true positives, with 109 false positives and 255 false negatives.

AUC scores of 0.91 for training and 0.90 for testing underscore the model's effectiveness in classifying instances. Key features contributing to the model's performance are Cashback and Tenure, highlighting their importance in predicting churn. Overall, the hyper-tuned Ada Boost model demonstrates decent precision and recall, with potential for improvement in capturing class 1.

xxiv. Interpretation of the Ada Boost Model with SMOTE:

The Ada Boost model with Synthetic Minority Over-sampling Technique (SMOTE) showcases its performance through balanced precision and recall of 88% for both class 0 (customers who stay) and class 1 (potential churners) in the training set, along with commendable accuracy.

Moving to the test set, the model maintains respectable precision of 94% for class 0 and 58% for class 1, with corresponding recall rates of 89% and 74%. The confusion matrix for training data shows 5766 true negatives and 5746 true positives, with 790 false positives and 810 false negatives. In the test data, 2502 true negatives and 420 true positives, along with 306 false positives and 150 false negatives.

AUC scores of 0.95 for training and 0.89 for testing underscore the model's effectiveness in classifying instances, particularly in the training data. Key features contributing to the model's performance include "Coupon used for payment" and "Tenure."

Overall, the Ada Boost model with SMOTE demonstrates a balanced ability to predict both classes, with an accuracy rate of 88% and 87%. However, there's room for improvement in predicting churn (Class 1).

xxv. Interpretation of the Gradient Boosting Model:

The Gradient Boosting model's performance is evaluated using precision, recall, F1-score, accuracy, and other metrics. In the training set, it achieves high precision of 92% for class 0 and 81% for class 1. However, recall for class 0 is 97%, while for class 1, it's 60%, indicating potential for improvement in capturing actual churn cases. These metrics collectively provide insights into the model's performance.

Transitioning to the test set, the pattern continues, with 92% precision for class 0 and 79% for class 1. Recall for class 0 is 97%, and for class 1, it's 58%. The model's accuracy, which measures overall correctness, is notable in both the training and test datasets.

The confusion matrix for the training data shows that it accurately predicted 6368 instances as class 0 (true negatives) and 795 instances as class 1 (true positives). However, it made 188 false predictions for class 1 and 531 false predictions for class 0.

Similarly, in the test data, the model accurately predicted 2722 instances as class 0 and 332 instances as class 1. Nonetheless, it made 86 false positives for class 1 and 238 false negatives for class 0. AUC scores of 0.94 for training and 0.92 for testing underscore the model's effectiveness in classifying instances, particularly in the training data.

Key features contributing to the model's performance include "Tenure" and "Complain_ly," highlighting their importance in predicting churn. Overall, the Gradient Boosting model demonstrates strong predictive ability, with potential room for improvement in capturing class 1.

xxvi. Interpretation of the Gradient Boosting Model: Hyper Tuning

These parameter settings define variations for fine-tuning the Gradient Boosting model in our business report. They control aspects like step size, tree depth, node splitting, number of stages, and randomness. By exploring these combinations, we aim to optimize the model's predictive performance for our business requirements.

The below shows the best-performing Gradient Boosting model based on these parameters:

```
Best Parameters: {'random_state': 1, 'n_estimators': 50, 'min_samples_split': 10, 'max_depth': 7, 'learning_rate':
0.2}
Best Estimator: GradientBoostingClassifier(learning_rate=0.2, max_depth=7, min_samples_split=10,
                           n_estimators=50, random_state=1)
```

The Gradient Boosting model, after hyper-tuning, achieves exceptional performance metrics. In the training set, it attains perfect precision and high recall for both classes: 100% precision for class 0 and 100% precision for class 1. Class 0 recall is 100%, and class 1 recall remains strong at 98%. Transitioning to the test set, the model sustains high precision rates of 96% for class 0 and 93% for class 1. Class 0 recall is 99%, while for class 1, it is 79%, capturing a significant portion of actual churn cases.

The confusion matrix for training data shows that the model accurately predicted 6555 instances as class 0 and 1304 instances as class 1. However, it made one false prediction for class 1 and 22 false predictions for class 0, indicating a potential for overfitting. Similarly, in the test data, the model accurately predicted 2776 instances as class 0 and 449 instances as class 1. Nonetheless, there were 32 instances wrongly classified as class 1 and 121 instances inaccurately predicted as class 0.

AUC scores of 1.00 for training and 0.98 for testing highlight the model's exceptional classification ability, particularly in the training data. Key features contributing to the model's performance, as identified through hyper-tuning, are Tenure and Cashback, indicating their significance in predicting churn. The model demonstrates outstanding precision and accuracy, making it a robust contender for effective churn prediction. However, the recall in the test set is 79%, indicating potential for improvement to achieve a more accurate prediction of churning customers. Additionally, the perfect precision and recall achieved in the training set suggest the possibility of overfitting, which should be carefully addressed.

xxvii. Interpretation of the Gradient Boosting Model: with SMOTE

The Gradient Boosting Model with Synthetic Minority Over-sampling Technique (SMOTE) is assessed through precision, recall, accuracy, and other metrics. In the training set, the model demonstrates balanced precision and recall of 92% for both class 0 (customers who stay) and class 1 (potential churners), resulting in an overall accuracy of 92%. Transitioning to the test set, the model maintains a precision of 94% for class 0 and 67% for class 1, with corresponding recall rates of 93% and 71%. The F1-score, which balances precision and recall, provides an overall measure of the model's performance.

The confusion matrix for the model's training data shows that it accurately predicted 6060 instances as class 0 (true negatives) and 6013 instances as class 1 (true positives). However, it misclassified 496 instances as class 1 when they were actually class 0 (false positives) and 543 instances as class 0 when they were actually class 1 (false negatives). Similarly, in the test data, the model accurately predicted 2612 instances as class 0 and 405 instances as class 1. Nonetheless, it made 196 instances false positives for class 1 and 165 instances false negatives for class 0. The AUC scores of 0.97 for training and 0.91 for testing underline the model's effectiveness in classifying instances, particularly in the training data.

Key features contributing to the model's performance include "Tenure" and "Complain_ly," highlighting their importance in predicting churn. In the training set, the model demonstrates balanced precision and recall of 92% for both class 0 and class 1, resulting in an impressive overall accuracy of 92%. Similarly, in the test set, the model maintains a solid precision of 94% for class 0 and 67% for class 1, accompanied by high accuracy rates of 89%. These results indicate the model's ability to effectively predict both classes, but there remains room for improvement in predicting class 1.

e. Feature Importance of Various Model.

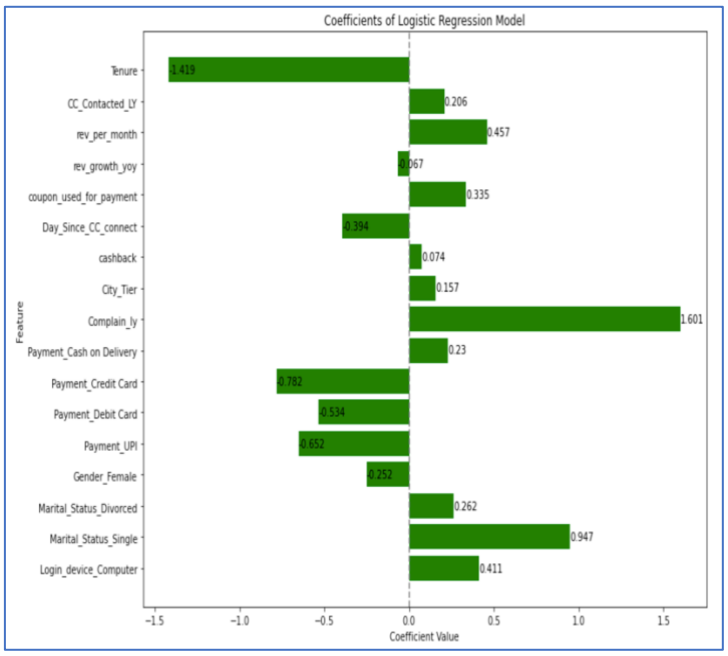

Figure 32: Logistic Regression - Feature Importance

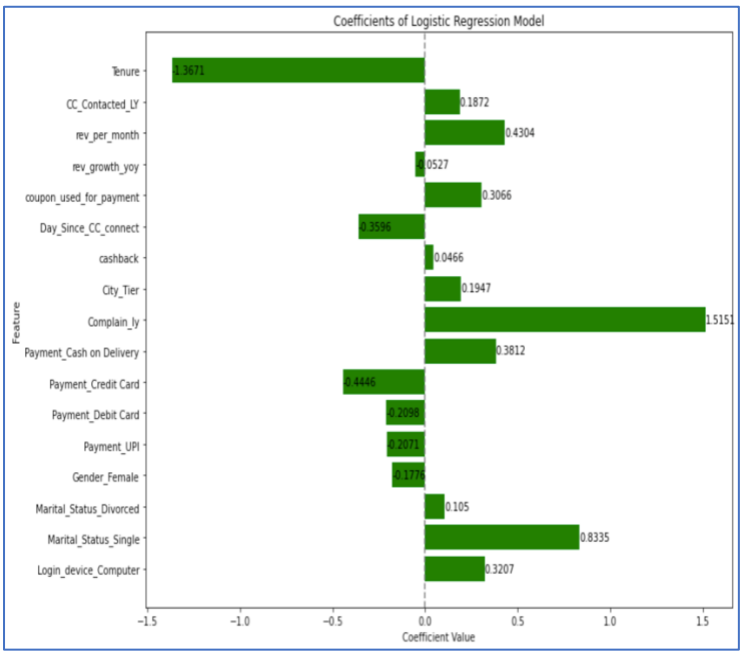

Figure 33:Logistic Regression - Hyper Tuning - Feature Importance

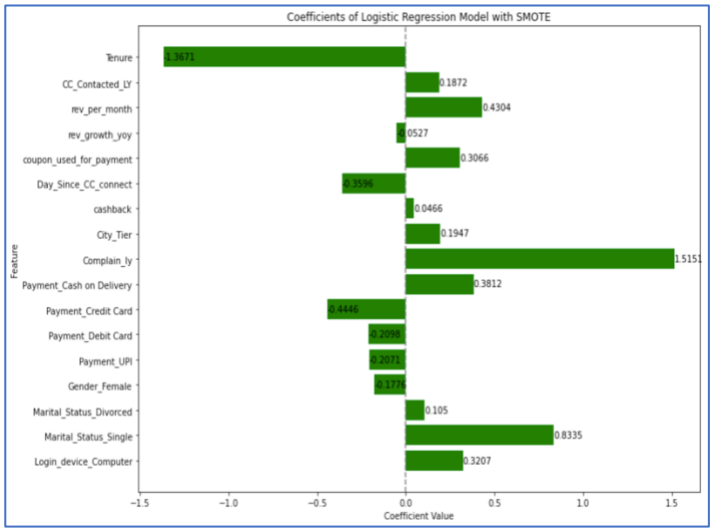

Figure 35:Logistic Regression with SMOTE - Logistic Regression Model

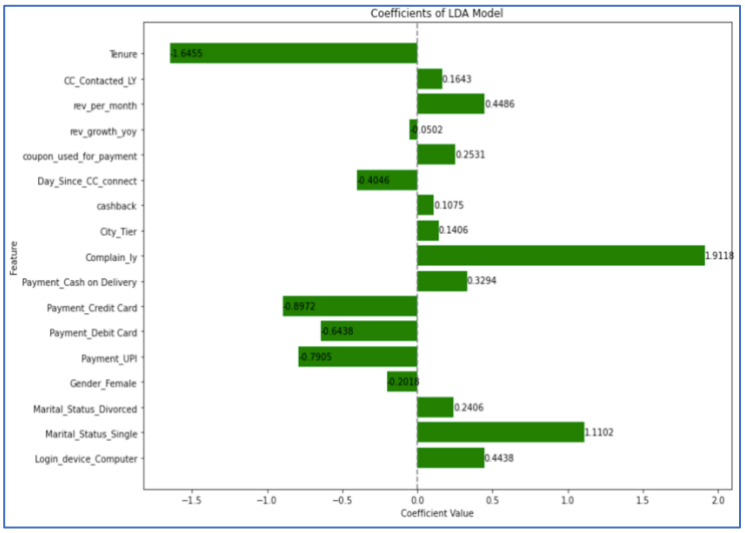

Figure 34: LDA Model - Feature Importance

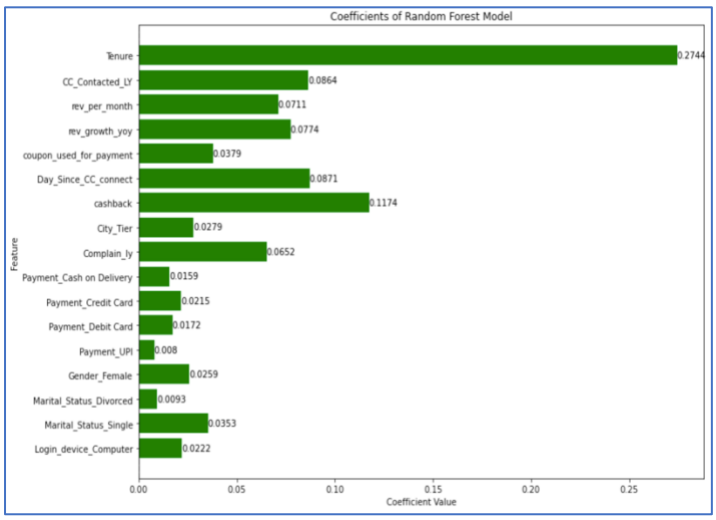

Figure 37:RF Model - Feature Importance

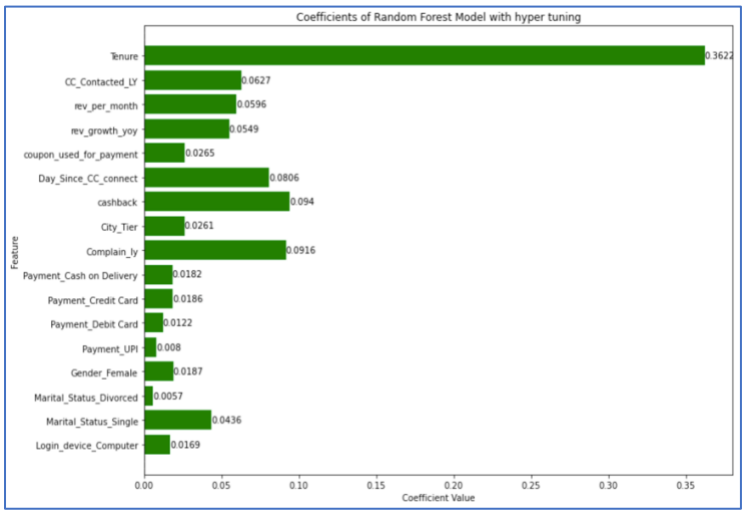

Figure 36: RF Model with Hyper tuning - Feature importance

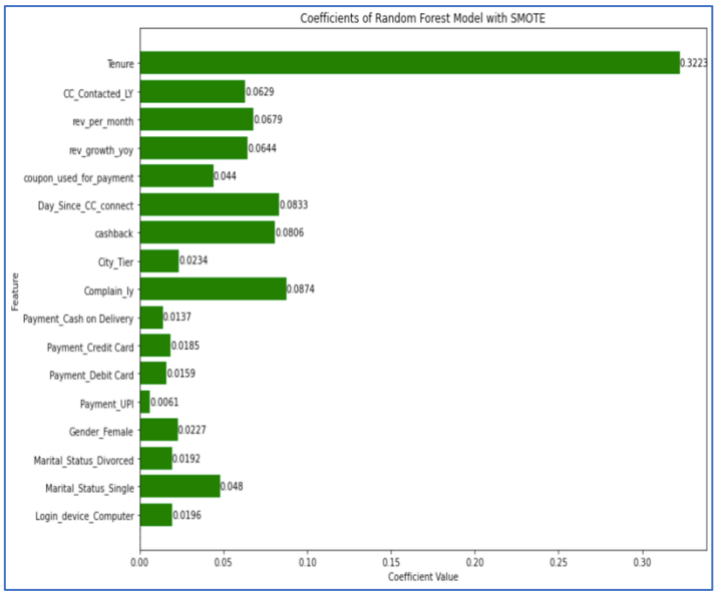

Figure 38:RF Model with SMOTE - Feature importance

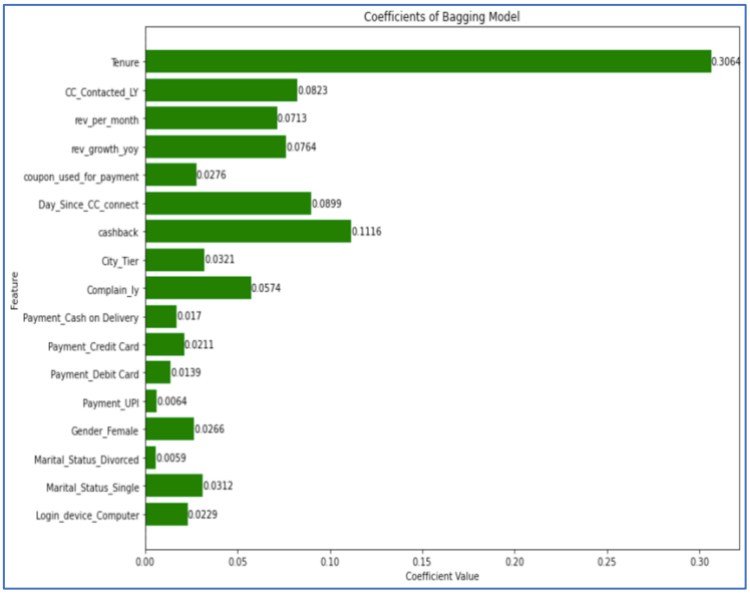

Figure 39:Bagging Model - Feature Importance

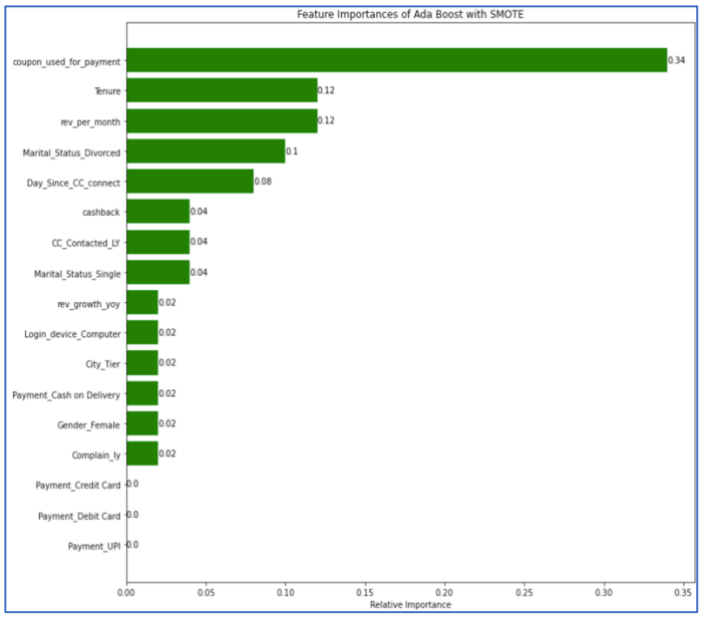

Figure 41:Ada Boost with SMOTE - Feature importance

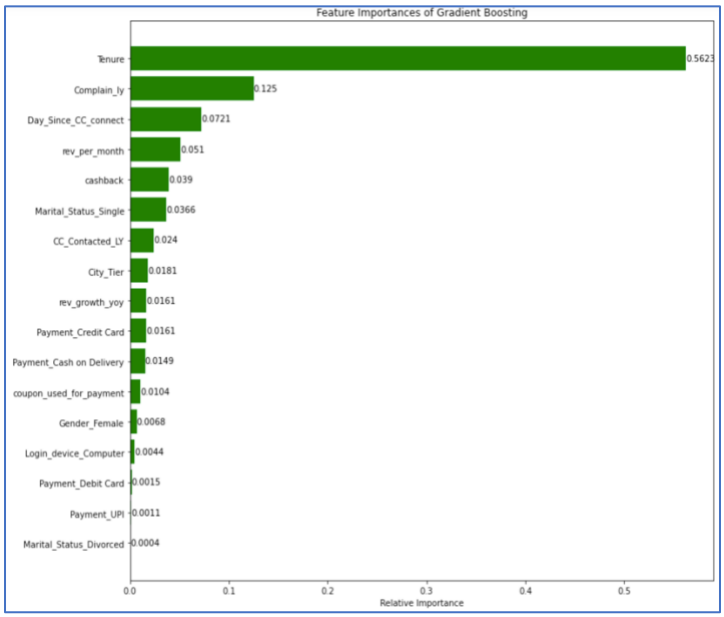

Figure 40: Gradient Boosting - Feature importance

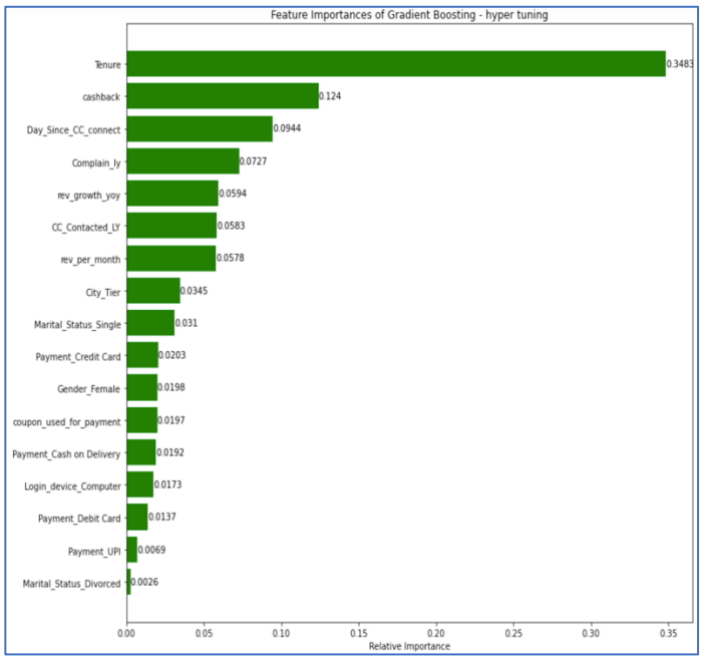

Figure 43: Gradient Boosting with Hyper tuning - Feature importance

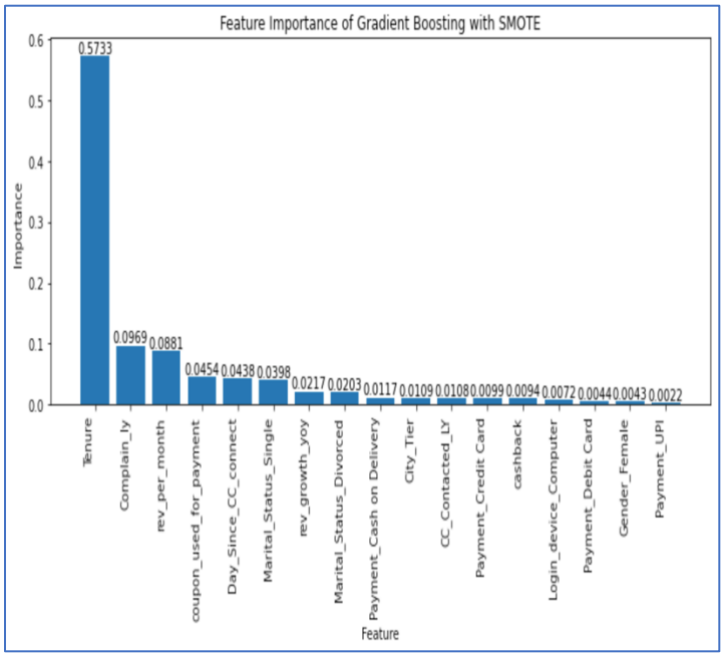

Figure 42: Gradient Boosting with SMOTE - Feature importance

f. Interpretation of the important feature of various model:

In our exploration of different models, we consistently observed certain features that stood out as crucial drivers of prediction accuracy. These features include "Tenure," "Cash back," "Days since CC Connect," "CC contact last year," and "Complaints last year." Their recurrent appearance as important features across multiple models underscores their significance in influencing customer behaviour and churn likelihood.

1. **Tenure:** The length of time a customer has remained with our service appears to be a significant indicator. Long-term customers might have developed a stronger loyalty, making them less likely to churn.
2. **Cash back:** The presence of cash back offers seems to play a notable role. Customers who have benefited from cash back incentives might perceive higher value in our services, increasing their retention.
3. **Days since CC Connect:** This indicates the recency of customer service interactions. A shorter duration between interactions suggests proactive engagement, which could foster a positive customer experience and reduce churn chances.
4. **CC contact last year:** Customers who have engaged with customer service in the past year might have ongoing concerns or needs. Addressing these proactively could enhance customer satisfaction and retention.
5. **Complaints last year:** The occurrence of complaints in the previous year serves as a red flag. Addressing and resolving customer grievances promptly can significantly impact customer satisfaction and ultimately reduce churn risk.

It's interesting to note that the Support Vector Machine (SVM) model didn't give these specific features as much importance. This could be because SVM looks at things in a different way compared to other models. Instead of focusing on individual features, SVM pays more attention to how well it can separate different groups. Even though SVM is accurate, it's unique in how it picks out important features. This could explain why it didn't highlight the same things as the other models did.

Knowing how important these features are can help us take smart actions. By working on things like making customer service better, improving loyalty programs, and dealing with complaints quickly, we can stop customers from leaving and make our business better overall.

g. Interpretation of the most optimum Model : Insight from the Analysis:

Most optimum models:

After a comprehensive analysis of all 27 models, it is evident that the following four models have exhibited exceptional performance across various metrics.

Models	Training Dataset (70%)								Testing Dataset (30)%							
Pefromance Metrics	Precision (0)	Recall (0)	F1-score (0)	Precision (1)	Recall (1)	F1-score (1)	Accuracy	AUC Score	Precision (0)	Recall (0)	F1-score (0)	Precision (1)	Recall (1)	F1-score (1)	Accuracy	AUC Score
KNN Model (with Hyper-tuning)	1	1	1	1	1	1	1	1	0.97	0.99	0.98	0.92	0.82	0.87	0.96	0.96
RandomForestClassifier (SMOTE)	1	1	1	1	1	1	1	1	0.97	0.98	0.98	0.9	0.85	0.88	0.96	0.98
Bagging (SMOTE)	1	1	1	1	0.99	0.99	1	0.99	0.96	0.97	0.97	0.86	0.82	0.84	0.95	0.9
Support vector Machine (with Hyper-tuning)	0.96	0.99	0.98	0.96	0.81	0.88	0.96	0.99	0.94	0.98	0.96	0.9	0.71	0.79	0.94	0.95

Table 19: Top 4 - Most optimum models

In a more detailed examination of the remaining models, a pattern of overfitting emerges, prominently observed in the **K-Nearest Neighbors (KNN), RandomForestClassifier (with SMOTE), and Bagging (with SMOTE) models**. These models, while showcasing remarkable prowess on the training dataset, appear to falter in their effectiveness on the testing dataset, particularly evidenced by their diminished recall values for class 1 during the testing phase. This recurrent trend suggests a potential issue of over-adaptation to the training data, subsequently hampering their ability to generalize adeptly to novel and unseen data.

h. The Optimum Model: Support Vector Machine (SVM) model with hyperparameter

The Hyper-Tuned SVM model stands out as the optimum choice for churn prediction due to several compelling reasons. Its selection is based on a thorough evaluation of its performance metrics and its ability to address the specific challenges posed by the churn prediction task.

a. **Balanced Precision and Recall:** The model's precision values of 96% for both classes in training and 94% for class 0 and 90% for class 1 in testing signify its balanced predictive accuracy. This balance is crucial in preventing an overly optimistic view of performance by ensuring both positive and negative predictions are reliable.

b. **Effective Churn Detection:** The model's recall of 81% for class 1 in training and 71% in testing demonstrates its capability to correctly identify a significant portion of actual churn cases. This is pivotal for the primary goal of churn prediction - capturing customers who are likely to leave the service.

c. **Low False Predictions:** The confusion matrices reveal that the model's misclassification of instances is minimal. In both training and testing, the model shows a low number of false predictions for both classes, indicating its robustness in distinguishing between non-churn and churn instances.

d. **Consistent Performance:** The model's ability to maintain its performance levels across training and testing datasets suggests that it is not overfitting or memorizing the training data. This consistency is a strong indicator of its generalization capability to unseen data.

e. **Strong Discriminatory Power:** The high AUC scores of 0.99 for training and 0.95 for testing demonstrate the model's ability to effectively separate the two classes, reinforcing its capacity to make informed predictions.

f. **Optimized Hyperparameters:** The hyperparameter tuning process ensures that the model's parameters are fine-tuned for the specific problem at hand. This optimization enhances its overall performance and predictive accuracy.

Considering these factors collectively, the Hyper-Tuned SVM model emerges as the optimum choice for churn prediction. Its balanced precision and recall, low false predictions, consistent performance, strong discriminatory power, and optimized hyperparameters make it a reliable and effective tool for identifying potential churn instances.

i. Effort to improve model performance:

Enhancing Support Vector Machine (SVM) with Hyperparameter Tuning:

The **SVM model with hyperparameter** tuning demonstrates solid performance across various metrics. It exhibits a balanced precision and recall for both classes, reflecting its ability to correctly identify customers who will stay (class 0) and those at risk of churning (class 1). However, there is room for improvement in the recall for class 1, indicating the potential to capture more instances of churn accurately.

1. **Adjusting Hyperparameters:** Fine-tuning the hyperparameters further might lead to better generalization and improved recall. Exploring different combinations of parameters such as the regularization parameter (C) and the choice of kernel can help the model better capture the patterns present in the data.
2. **Feature Engineering:** Carefully selecting or engineering relevant features can provide the model with more discriminatory information, potentially aiding in the identification of class 1 instances.
3. **Ensemble Methods:** Combining the predictions of multiple SVM models with slight variations in hyperparameters or training data can often lead to better overall performance and recall.
4. **Model Interpretation**: Gaining a deeper understanding of the features and their impact on predictions could provide insights into why certain instances are misclassified as class 0 instead of class 1. This understanding could guide adjustments to improve recall.

   By iteratively applying these strategies and monitoring their impact on recall, the SVM model's performance can potentially be enhanced, resulting in more accurate predictions of churn instances.

5. Model validation - How was the model validated ? Just accuracy, or anything else too ?

a. Model validation

The evaluation method depends on whether the model is unbalanced or balanced. Unbalanced models, dealing with uneven class distributions, are assessed using the F1 score, which considers both precision and recall. This is particularly useful for smaller classes, like potential churners. In contrast, balanced models are evaluated using accuracy, which gauges overall correctness.

For model validation, is didn't rely solely on accuracy. Instead, we employed a comprehensive approach involving precision, recall, F1-score, and accuracy. This multifaceted analysis allowed us to understand the model's performance from various angles, identifying strengths and areas needing improvement in predicting churn more effectively.

Classification Report for SVM Model with hyper-tuning:

Classification Report for Training Set:

	precision	recall	f1-score	support
0.0	0.96	0.99	0.98	6556
1.0	0.96	0.81	0.88	1326
accuracy			0.96	7882
macro avg	0.96	0.90	0.93	7882
weighted avg	0.96	0.96	0.96	7882

Classification Report for Test Set:

	precision	recall	f1-score	support
0.0	0.94	0.98	0.96	2808
1.0	0.90	0.71	0.79	570
accuracy			0.94	3378
macro avg	0.92	0.85	0.88	3378
weighted avg	0.94	0.94	0.93	3378

Table 20:Classification Report for SVM Model with hyper-tuning

Confusion Matrix : SVM Model with Hyper- tuning

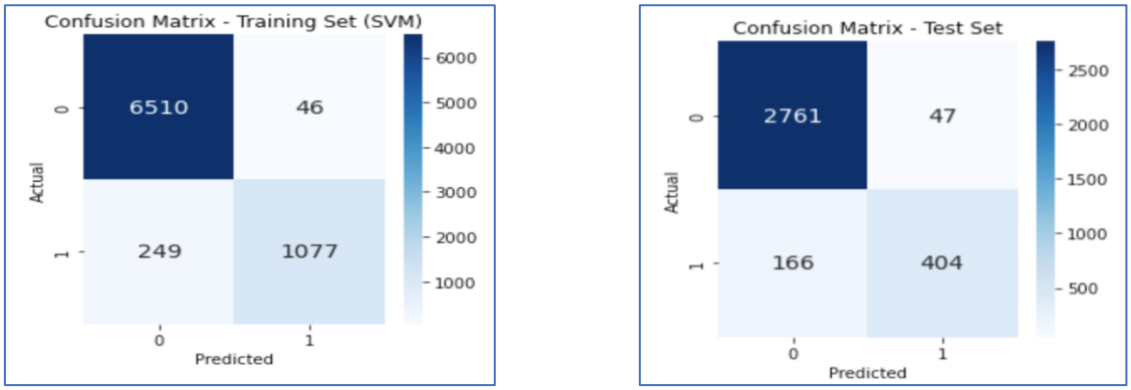

Figure 44:Confusion Matrix : SVM Model with Hyper- tuning

AUC & ROC Score: SVM Model with Hyper- tuning

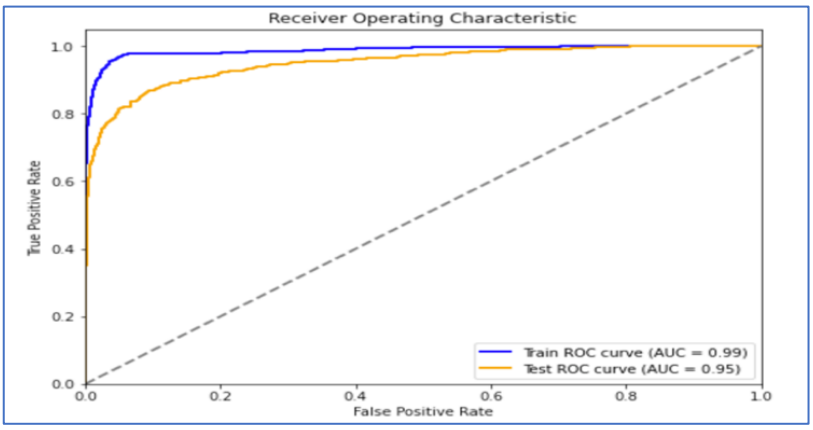

Figure 45:AUC & ROC Score: SVM Model with Hyper- tuning

Train AUC Score: 0.99
Test AUC Score. : 0.95

b. Performance metrics of SVM Model:

The classification reports reveal the model's performance on both the training and test datasets. In the training set, the model achieves strong precision and recall for both classes. Specifically, it attains a precision of 0.96 for class 0 (customers who stay) and 0.96 for class 1 (potential churners). The corresponding recall values are 0.99 for class 0 and 0.81 for class 1. These metrics culminate in an overall accuracy of 0.96.

Transitioning to the test set, the model maintains commendable precision rates of 0.94 for class 0 and 0.90 for class 1. However, the recall for class 1 drops to 0.71, influencing the F1-score for this class to be 0.79. The macro-average F1-score for the test set is 0.88, reflecting the overall balance between precision and recall. Additionally, the AUC scores for the model's training and test data are 0.99 and 0.95 respectively, demonstrating its ability to distinguish between the two classes.

c. Model validation of SVM Model:

In terms of model validation, our evaluation went beyond simply looking at accuracy as a sole metric. Instead, we opted for a more thorough and comprehensive approach. The assessment encompassed a range of performance indicators, including precision, recall, F1-score, and accuracy. By considering these multiple metrics, we gained a deeper and more nuanced understanding of the model's overall performance and its specific strengths and weaknesses.

This analysis allowed us to uncover not only where the model excelled, but also where it might benefit from improvements. For instance, while accuracy provides a general overview of correctness, metrics like precision, recall, and F1-score provide insights into how well the model performs for individual classes, especially in scenarios where class distribution is imbalanced.

Significantly, the SVM model with hyperparameter tuning, which emerged as the optimal choice, is characterized as an unbalanced model. This aspect highlights the importance of our comprehensive validation strategy. In unbalanced settings, where one class significantly outweighs the other, relying solely on accuracy can be misleading. The approach, which considers various performance metrics, provides a more robust evaluation of the model's ability to handle such imbalances and make accurate predictions for both classes.

In summary, our thorough validation approach, which looked at different measures and tackled the issue of imbalanced data, has made our findings more trustworthy and important. This is especially true when it comes to understanding how well the SVM model with hyperparameter tuning can predict churn accurately.

6. Final interpretation / recommendation:

a. Business Insight from Exploratory data Analysis:

1. **Tenure Impact:** Customers with shorter account tenures tend to churn more frequently, highlighting the importance of building long-term customer relationships for better retention.

2. **Customer Care Engagement:** Churned customers show a slightly higher median in customer care contacts, suggesting that addressing customer concerns and queries effectively can influence retention.

3. **Segment-specific Strategies:** Different account segments exhibit varying churn rates. Customized retention strategies for segments like "Regular Plus," "Super," and "HNI" can enhance loyalty, while "Regular" and "Super Plus" segments have lower churn, indicating effective strategies already in place.

4. **Marital Status Matters:** Singles tend to have higher churn rates compared to married or divorced customers. Understanding the factors influencing singles' churn decisions can aid in tailored retention efforts.

5. **Early Churn Indicators:** Churned customers typically have account tenures below 10. Addressing factors leading to early churn can boost customer satisfaction and loyalty.

6. **Preferred Payment Modes:** Most customers prefer Debit Cards and Credit Cards for payments. Recognizing and catering to these preferences can enhance customer convenience and satisfaction.

7. **Service Satisfaction:** Moderately satisfied customers rate services at 3.0, making up the majority. Analysing lower and higher scores can guide service improvements and loyalty-building strategies.

8. **City Tier and Churn:** City tier doesn't strongly predict churn, indicating that factors beyond geographic location play a more significant role in customer attrition.

9. **Account Tenure and Churn:** Account tenure shows a clear correlation with churn, highlighting the need to focus on retaining newer customers through personalized engagement strategies.

10. **Cluster insight:** The "Cluster Churn Analysis Bar Chart" highlights a concerning trend in Cluster 2 with a higher churn rate, urging attention to retention efforts. In contrast, Clusters 0 and 1 exhibit loyal customers, offering a chance to enhance retention and satisfaction through tailored strategies. This analysis guides the implementation of personalized approaches, bolstering customer loyalty and minimizing churn.

b. Business Insight: Unveiling Key Churn Indicators

Our exploration of various models has consistently highlighted specific features that significantly influence the accuracy of churn prediction. These features include "Tenure," "Cash back," "Days since CC Connect," "CC contact last year," and "Complaints last year." Their consistent presence as crucial indicators across multiple models emphasizes their role in shaping customer behaviour and predicting churn likelihood.

1) **Tenure:** The duration a customer remains with our service emerges as a crucial indicator. Long-term customers likely exhibit higher loyalty, reducing their likelihood to churn.

2) **Cash back:** The presence of cash back offers plays a notable role. Customers benefiting from cash back incentives may perceive greater value in our services, enhancing their retention.

3) **Days since CC Connect:** This reflects the recency of customer service interactions. A shorter interval between interactions suggests proactive engagement, fostering positive customer experiences and lowering churn risks.

4) **CC contact last year:** Customers engaging with customer service within the past year could have ongoing concerns or needs. Addressing these promptly can elevate customer satisfaction and retention rates.

5) **Complaints last year:** Incidents of complaints in the previous year act as a warning sign. Swiftly resolving customer grievances can significantly impact satisfaction and mitigate churn risks.

It's intriguing that the Support Vector Machine (SVM) model didn't assign the same importance to these features. SVM takes a distinct approach, focusing more on effective group separation than individual features. While SVM's accuracy is noteworthy, its unique feature selection methodology could explain the difference in highlighted factors compared to other models. Understanding the significance of these features empowers strategic action. Enhancing customer service, refining loyalty programs, and addressing complaints promptly can prevent churn, ultimately enhancing our overall business performance.

c. Recommendation:

In today's competitive business landscape, retaining customers is a top priority for sustained success. Customer churn, the loss of valuable customers, can have a profound impact on revenue and growth prospects. Unveiling key indicators that influence churn through analysis offers businesses a unique opportunity to take proactive steps in retaining their customer base. By harnessing the power of data insights, businesses can strategically address the factors that contribute to churn and implement effective measures to curb it.

Important Recommendations to Reduce Customer Churn:

Leverage Tenure for Personalized Engagement: Recognize the value of long-term customers and implement targeted retention efforts for newer customers. Offer exclusive benefits, discounts, or loyalty programs that encourage customers to stay and grow their loyalty over time.

- **Maximize Cash Back Programs:** Capitalize on the positive impact of cash back offers. Enhance and promote these programs to highlight the tangible value customers gain from continued engagement, reinforcing their motivation to remain loyal.

- **Prioritize Proactive Customer Engagement:** Utilize the "Days since CC Connect" indicator to enhance proactive customer engagement. Develop outreach strategies that initiate interactions before issues arise, showcasing your commitment to addressing customer needs.

- **Elevate Customer Service:** Recognize the significance of "CC contact last year" as a retention factor. Invest in training and resources for customer service teams to swiftly and effectively address customer queries, fostering positive experiences.

- **Swiftly Address Complaints:** Use the "Complaints last year" insight as a prompt for rapid issue resolution. Implement streamlined processes for handling complaints, ensuring customers' concerns are acknowledged and resolved promptly.

- **Fine-Tune SVM Model Interpretation:** Understand the SVM model's distinct approach to feature importance. While it highlights different factors, combining its insights with those of other models can offer a more comprehensive understanding of churn predictors.

- **Customize Retention Campaigns:** Tailor retention campaigns based on the specific insights from these key indicators. Craft messages and offers that directly address customers' tenure, engagement frequency, and potential concerns.

- **Enhance Loyalty Programs:** Incorporate findings about cash back's impact into loyalty program enhancements. Consider offering tiered rewards that align with different levels of engagement, encouraging customers to stay and engage more.

- **Proactively Address Customer Needs:** Leverage the recency of customer service interactions to anticipate customer needs. Implement automated follow-ups after interactions to ensure satisfaction and offer assistance if required.

- **Promote Positive Customer Experiences:** Utilize complaint resolution as an opportunity to showcase excellent customer service. Communicate transparently about resolutions, emphasizing your commitment to customer satisfaction.

By aligning strategies with the insights derived from these key churn indicators, businesses can take targeted actions to reduce customer churn, enhance loyalty, and ultimately improve their overall business performance. The combination of data-driven understanding and strategic implementation holds the key to long-lasting customer relationships.

d. Final Interpretation/conclusion

In summary, through data analysis and identifying churn indicators has illuminated strategies to enhance customer retention. By probing into customer behaviour and churn predictors, we've gained insights beyond mere numbers, guiding businesses in navigating customer relationships. From valuing long-term customers and proactive customer care to understanding different customer groups, our analysis underscores tailored approaches. Discovering the importance of elements like **cash back offers, recent interactions, and quick complaint** resolution further emphasizes the value of these insights.

Despite varying model perspectives, the consistent presence of these churn indicators and the unique viewpoint of the Support Vector Machine (SVM) with hyper tuning model highlight their significance. Transforming these insights into practical recommendations **(mentioned above)** empowers Ecommerce business to address churn, gain loyalty, and improve operations.

In the era of data-driven decisions, the synergy of data analysis and churn indicators offers a strong foundation for customer-centric businesses, enhancing retention and fostering growth. Utilizing these insights isn't just a strategy. it's a commitment to delivering value and securing lasting success.