CAPSTONE PROJECT

E-Commerce Customer Churn Prediction

Anurag VRS [DSBA Oct,2020]
vrsanurag@gmail.com

Contents

List of Figures

List of Tables

1. Introduction:

a) Business Problem:

Managing customer churn is one major challenge companies are facing especially those who are in E-Commerce industry. An E-Commerce company is facing a lot of competition in current market & are facing a major challenge to retain existing customers in the current situation. Account churn is a key thing in E-Commerce industry as one account can have multiple customers & losing one account might lead to losing more than one customer. The data set compromises details of various account IDs along customer demographics, revenue earned & ratings/scoring etc. It consists of over 11,000 accounts and around 19 variables.

b) Need of the Project:

The main objective of this project is to build/develop a model through which company can do churn prediction of the accounts and provide some segmented offers to the potential churners based on the business recommendations given through this project. Recommendations can be given based on the insights from the dataset by doing some descriptive & exploratory data analysis like how the data is distributed, whether there is correlation between the variables and by doing some visualization of the variables etc.

c) Data Dictionary & Understanding Data:

Variable	Description
AccountID	Account unique identifier
Churn	Account churn flag (Target)
Tenure	Tenure of account
City_Tier	Tier of primary customer's city
CC_Contacted_L12m	How many times all the customers of the account has contacted customer care in last 12months
Payment	Preferred Payment mode of the customers in the account
Gender	Gender of the primary customer of the account
Service_Score	Satisfaction score given by customers of the account on service provided by company
Account_user_count	Number of customers tagged with this account
Account_segment	Account segmentation on the basis of spend
CC_Agent_Score	Satisfaction score given by customers of the account on customer care service provided by company
Marital_Status	Marital status of the primary customer of the account
Rev_per_month	Monthly average revenue generated by account in last 12 months
Complain_l12m	Any complaints has been raised by account in last 12 months
Rev_growth_yoy	Revenue growth percentage of the account (last 12 months vs last 24 to 13 month)
coupon_used_l12m	How many times customers have used coupons to do the payment in last 12 months
Day_Since_CC_connect	Number of days since no customers in the account has contacted the customer care
Cashback_l12m	Monthly average cashback generated by account in last 12 months
Login_device	Preferred login device of the customers in the account

Table 1: Data Dictionary of the Given Dataset

(i) Checking the head of the Dataset:

AccountID	Churn	Tenure	City_Tier	CC_Contacted_LY	Payment	Gender	Service_Score	Account_user_count	account_segment	CC_Agent_Score	Marital_Status	rev_per_month	Complain_ly	rev_growth_yoy	coupon_used_for_payment	Day_Since_CC_connect	cashback	Login_device
20000	1	4	3	6	Debit Card	Female	3	3	Super	2	Single	9	1	11	1	5	159.93	Mobile
20001	1	0	1	8	UPI	Male	3	4	Regular Plus	3	Single	7	1	15	0	0	120.9	Mobile
20002	1	0	1	30	Debit Card	Male	2	4	Regular Plus	3	Single	6	1	14	0	3	NaN	Mobile
20003	1	0	3	15	Debit Card	Male	2	4	Super	5	Single	8	0	23	0	3	134.07	Mobile
20004	1	0	1	12	Credit Card	Male	2	3	Regular Plus	5	Single	3	0	11	1	3	129.6	Mobile

Table 2: Top five (5) Rows of the Dataset

- From the above table we can have a view of the top 5 rows of the dataset. Given dataset consists of 11,260 observations and 19 variables. ‘Churn’ is our target/independent variable which is of binary where ‘1’ implies that customer has churned and ‘0’ implies that customer has not churned.
- Few variables are of categorical like Payment, Gender, Martial Status, Account Segment etc.
- City Tier is segregated into three segments i.e., Tier 1, Tier 2 & Tier 3 cities.
- According to the data dictionary we are considering tenure in months, revenue per month in thousands and revenue growth in percentage. All monetary values will be considered in INR.

(ii) Checking the Info of the Dataset:

S.No	Column	Non-Null Count	Data Type
0	AccountID	11260	Continuous
1	Churn	11260	Continuous
2	Tenure	11158	Categorical
3	City_Tier	11148	Continuous
4	CC_Contacted_L12m	11158	Continuous
5	Payment	11151	Categorical
6	Gender	11152	Categorical
7	Service_Score	11162	Continuous
8	Account_user_count	11148	Categorical
9	account_segment	11163	Categorical
10	CC_Agent_Score	11144	Continuous
11	Marital_Status	11048	Categorical
12	rev_per_month	11158	Categorical
13	Complain_l12m	10903	Continuous
14	rev_growth_yoy	11260	Categorical
15	coupon_used_l12m	11260	Categorical
16	Day_Since_CC_connect	10903	Categorical
17	cashback_l12m	10789	Categorical
18	Login_device	11039	Categorical

Table 3: Info of the Dataset

From the table we can say the following,

- Missing values are present in almost all variables except for Account ID & Churn.
- By observing the data type, we can see that few variables which should be in continuous form but is in object/categorical type. It might be because few observations would have been entered with special characters like ‘@’, ‘$’, ‘*’ etc. This can be checked with unique values feature.
- Below are the variables which have been identified of wrong data type based on the data dictionary & purpose.
  - ✓ Tenure
  - ✓ Account user count
  - ✓ Revenue per month
  - ✓ Revenue growth YOY
  - ✓ Coupon used for payment
  - ✓ Day since cc connect
  - ✓ Cash back
  - ✓ Login Device

(iii) Removing & Renaming Unwanted attributes:

- By checking the unique values in the above-mentioned variables, we can observe that these variables consist of special characters like ‘@’, ‘&&&&’, ‘+’, etc. Hence, we will be replacing these values with null values.
- After replacing these values with null, the data type of these variables automatically changes to continuous type except for Login Device which basically is of categorical.
  - ✓ **Gender** – [Female, Male, F, nan, M]
  - ✓ **account_segment** – [Super, Regular Plus, Regular, HNI, Regular +, nan, Super Plus, Super +]
- By seeing the variables ‘Gender’ and ‘account_segment’ we can say that few attributes are similar but mentioned in different name, for example there is Female & F basically both means the same but are represented in different way. Hence, we will be replacing the attributes to unique name.
- After completing the above two steps we can see that most of the variables which were categorical earlier have been changed to continuous.
- In the given dataset we can see that account ID variable is not much of importance, hence we will be dropping this

Variable Name	Special Character Present	Count
Tenure	#	116
Account User Count	@	332
Revenue per month	+	689
Revenue Growth YOY	$	3
Coupon Used for Payment	#	1
	*	1
	$	1
Days_since_Cc connect	$	1
Cashback	$	2
Login Device	&&&&	539

Table 4: List of Variables having Special Characters

variable. After dropping the variable if we check for duplicates, around 264 observations are found to be duplicates. These duplicate observations will be removed.

2. Exploratory Data Analysis:

a) Summary of the Dataset:

(i) Checking the Summary for Continuous & Discrete Variables

	Churn	Tenure	City_Tier	CC_Contacted_LY	Service_Score	Account_user_count	CC_Agent_Score	rev_per_month	Complain_ly	rev_growth_yoy	coupon_used_for_payment	Day_Since_CC_connect	cashback
count	10996	10778	10884	10894	10898	10552	10880	10205	10639	10993	10993	10638	10523
mean	0.17	11.07	1.65	17.89	2.90	3.69	3.06	6.41	0.29	16.21	1.80	4.65	196.94
std	0.37	12.98	0.92	8.87	0.73	1.02	1.38	12.05	0.45	3.76	1.98	3.70	180.70
min	0	0	1	4	0	1	1	1	0	4	0	0	0
25%	0	2	1	11	2	3	2	3	0	13	1	2	147.36
50%	0	9	1	16	3	4	3	5	0	15	1	3	165.62
75%	0	16	3	23	3	4	4	7	1	19	2	8	200.86
max	1	99	3	132	5	6	5	140	1	28	16	47	1997

Table 5: Descriptive Statistics of Continuous & Discrete Variables

From the above summary table (for continuous variables) we can say the following,

- Mean and Median for most of the variables have significant differences, which indicates that the data might be skewed.
- Mean tenure of an account is around 11 months which tells us that most of the account's tenure might be between 0 to 25 months.
- For few variables like 'cc_contacted_Ly', 'rev_per_month', 'cashback' etc. there is significant difference between the 75% value and maximum value which indicates that there is presence of outliers which need to be checked further.

(ii) Checking the Summary for Categorical Variables:

From the above table we can say the following,

- Around 41% of accounts have used debit cards as preferred mode of transaction/payment.
- Most of the accounts primary customers are Males, i.e., around 60%
- Around 36% of accounts are of Regular Plus type account. Which is a mid-level kind of account.
- Marital status of around 52% account holder is married, there might be chances of multiple customers linked to single account.
- Around 71% of account holders logged through mobile phones which tells us that most of the customers prefer login through mobile.

	count	unique	top	freq
Payment	10887	5	Debit Card	4482
Gender	10888	2	Male	6545
account_segment	10899	5	Regular Plus	4012
Marital_Status	10785	3	Married	5708
Login_device	10236	2	Mobile	7306

Table 6: Descriptive Statistics of Categorical Variables

b) Univariate Analysis

(i) Univariate Analysis for Continuous Variables (Tenure, CC_Contacted_LY, rev_per_month)

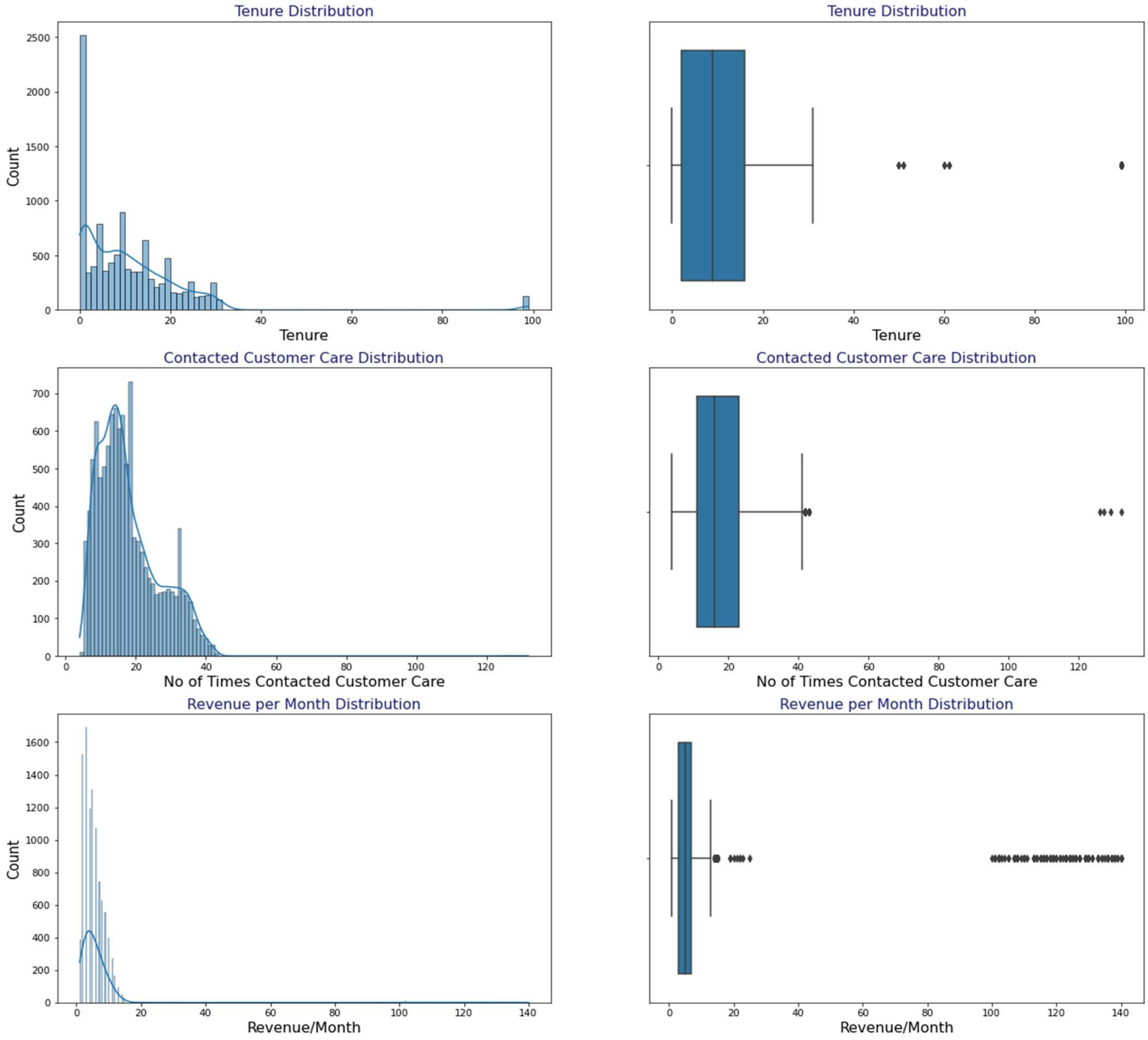


Figure 1: Distribution Plot of Continuous Variables (1/3)

From the above distribution plot, we can say the following

- Distribution of tenure seems to be skewed; Tenure basically tells us that how many months a customer was retained & we can see that most of the account's tenure ranged between 0 to 25 months. From the above graph we can also observe that they are few accounts whose tenure is in the range of 95 to 99 months which might be an extremely rare scenario, which might also indicate that this might be an outlier.
- Distribution of Cc_contacted_Ly seems to be skewed. The variable tells us that how many times a customer contacted customer during the last 1 year. Most of the customers have contacted the customer care between 5 to 20 times, where the average being 16 times. We can see few customers have contacted more than the 75th percentile, which might be a potential outlier & rare occurrence.
- Revenue per month also seems to be skewed, where most of the revenue from customers is ranging 0 to 18 thousand rupees, whereas the average revenue from the customer is around 5 thousand rupees. We can also observe that there are few accounts where the revenue is more than the 75th percentile of the data i.e., ranging from 100 to 140 thousand.

(ii) Univariate Analysis for Continuous Variables (rev_growth_yoy, coupon_used_for_payment, Day_Since_CC_connect)

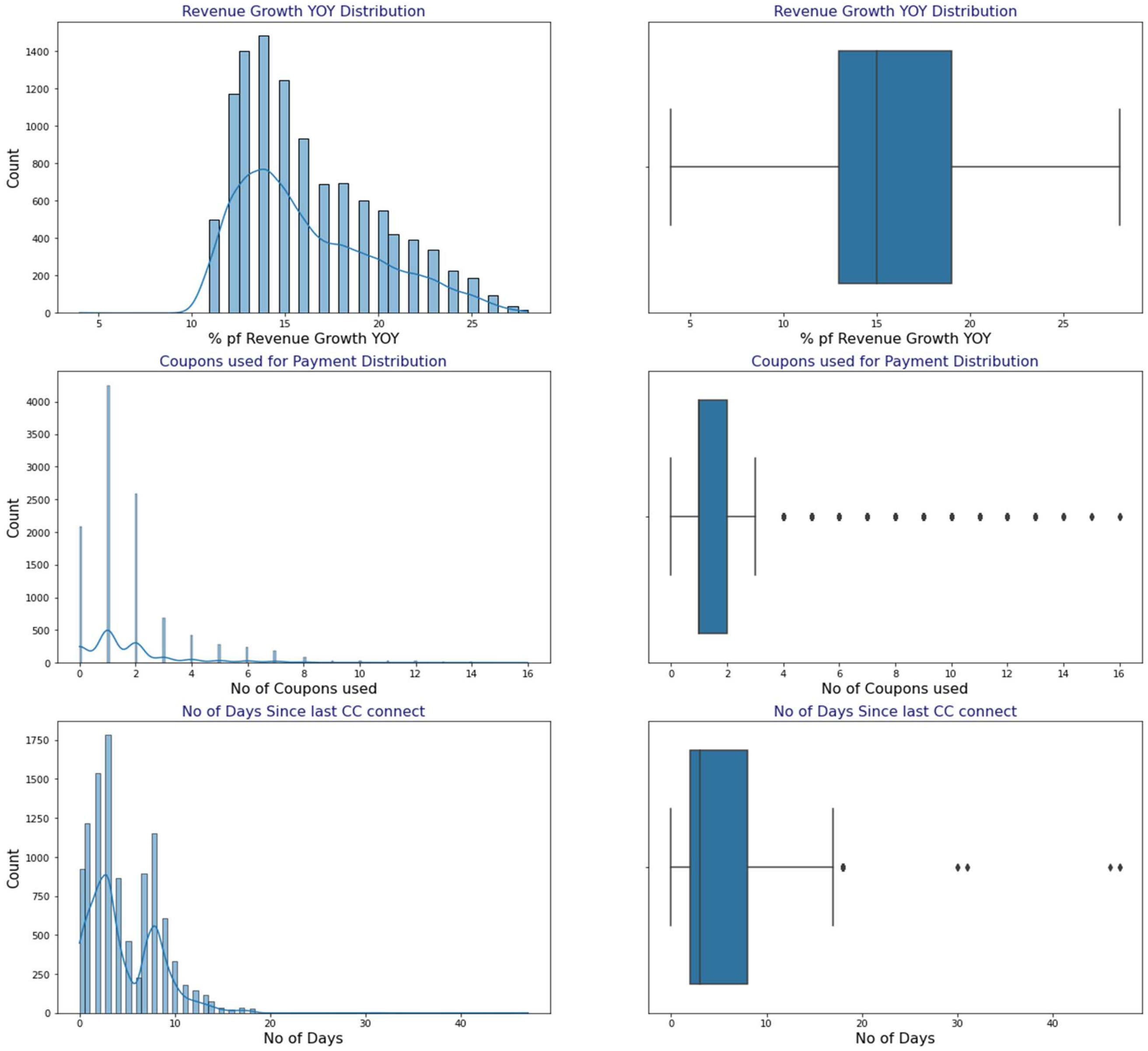


Figure 2: Distribution Plot for Continuous Variables (2/3)

From the above distribution plot, we can say the following

- Distribution of revenue growth is partly symmetrical; it tells us that most of the account's revenue growth is between 12 to 20%. Average percentage of revenue growth is 15% and the maximum being 28%.
- Distribution of Day_Since_CC_Connect seems to be skewed with multiple peaks. It basically tells us the number of days since the customer contacted customer care last. We can see that most of the customers are ranging between 0 to 10 days. On an average after 3 days customer gets connected to the customer care. We can also see that few customers didn't connect to customer care for more than 30 days.

(iii) Univariate Analysis for Continuous Variables (Cashback)

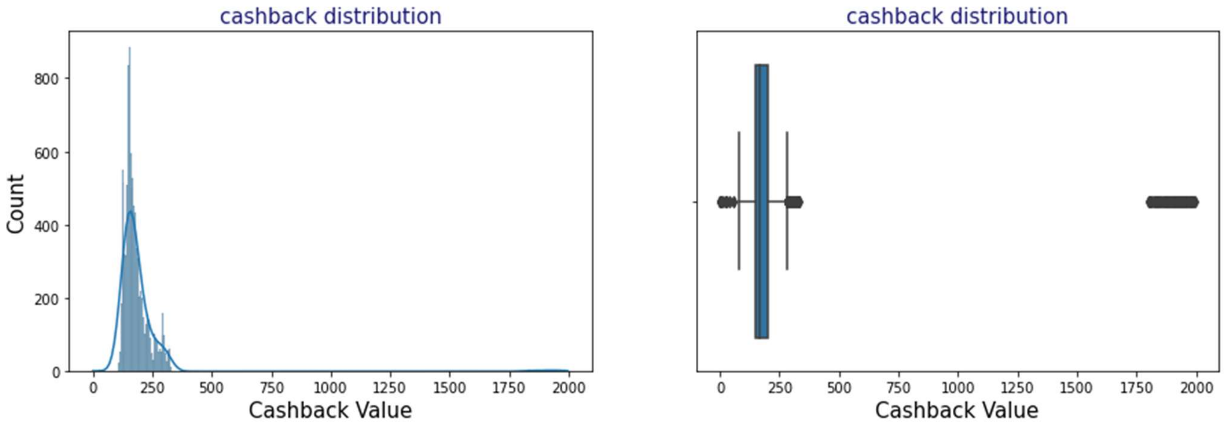
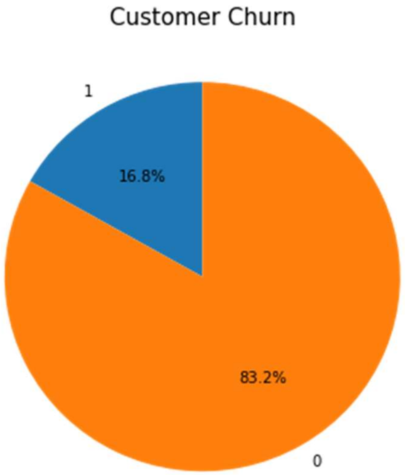


Figure 3: Distribution Plot for Continuous Variables (3/3)

From the above plot we can see that the variable cashback has a symmetrical distribution. It is the monthly average cashback generated by the account for last 12 months. We can see that most of the accounts has generated cashback ranging from 100 to 300 rupees. Average cashback generated by the account is around 160 rupees. We can also observe that there are few accounts which has generated more than 1800 rupees which might be a rare occurrence and also tell us that these values might be a potential outlier.

(iv) Univariate Analysis for Target Variable (Customer Churn)



Customer churn variable basically is a binary variable where ‘0’ implies that the customer has not churned and ‘1’ implies that customer has churned. From the pie-chart plotted for the customer churn tells us that around 83% of accounts did not churn and around 17% of the accounts have churned or did not retain. This also indicates that the data might be skewed towards the accounts which have not churned. This need to be studied further.

Figure 4: Pie Chart for Customer Churn

(v) Univariate Analysis for Categorical Variables

1. City Tier

Following inferences can be drawn by seeing the plot for city tier variable.

- City Tier basically tells us the tier of primary customer’s city.
- Around 64% of the primary customer’s is from Tier – 1 city which tell us that a greater number of users are from this city tier.
- Around 30% of the primary customer’s is from Tier – 3 city. It indicates that more than 90% of primary customers are from Tier - 1 and Tier – 3 cities.

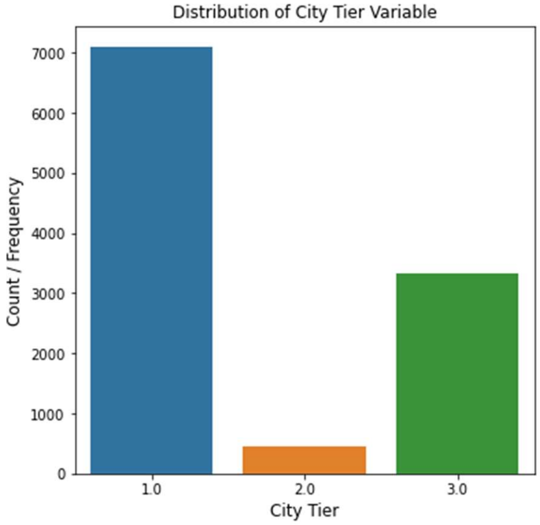


Figure 5: Distribution Plot for City Tier

2. Payment

Following inferences can be drawn by seeing the plot for payment variable.

- Around 5 different types of payment mode were used i.e., Debit Card, UPI, Credit Card, Cash on Delivery & E-Wallet.
- Around 41% of customers transaction was made through Debit card and around 30% of customers transaction was made through Credit card.
- They are only few customers whose transactions were made through UPI, Cash on delivery & E-wallet

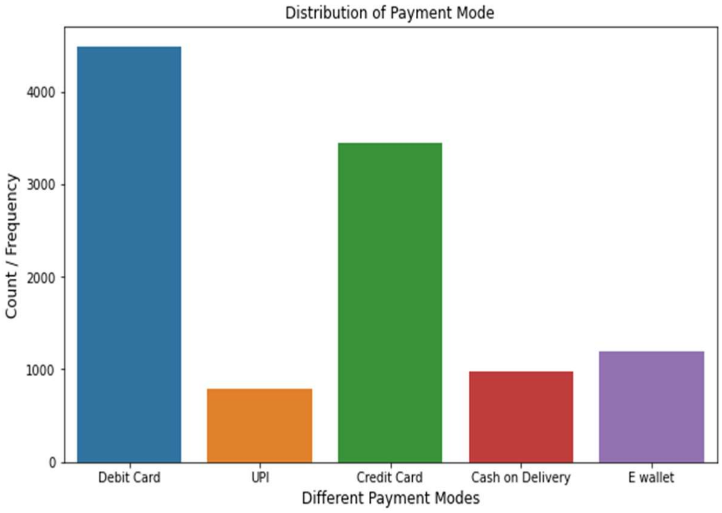


Figure 6: Distribution of Payment Mode

3. Gender

Following inferences can be drawn by seeing the plot for Gender variable.

- Around 60% of the primary customers of the account are owned or used by males.
- Around 40% of the primary customers of the account are owned or used by females.
- Difference between the male and female is less.

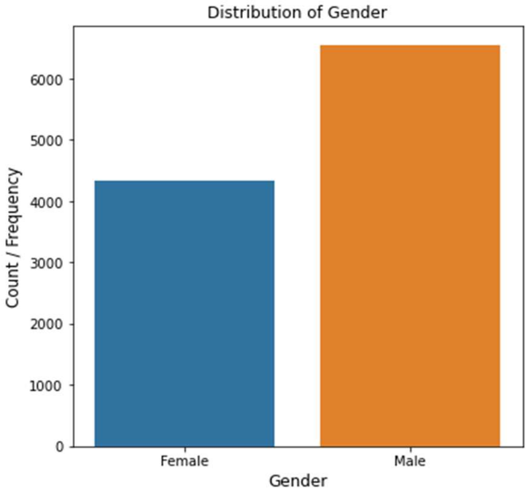


Figure 7: Distribution of Gender

4. Service Score

Following inferences can be drawn by seeing the plot for Service Score.

- Around 70% of customers have given a satisfaction score of 3 and above on the service provided by the company. Which tells us that most of the customers are satisfied by the service provided by the customer
- Around 30% of customers have given a satisfaction score of 2 and below on the service provided by the company, that means only few customers are not satisfied by the service provided by the customer.

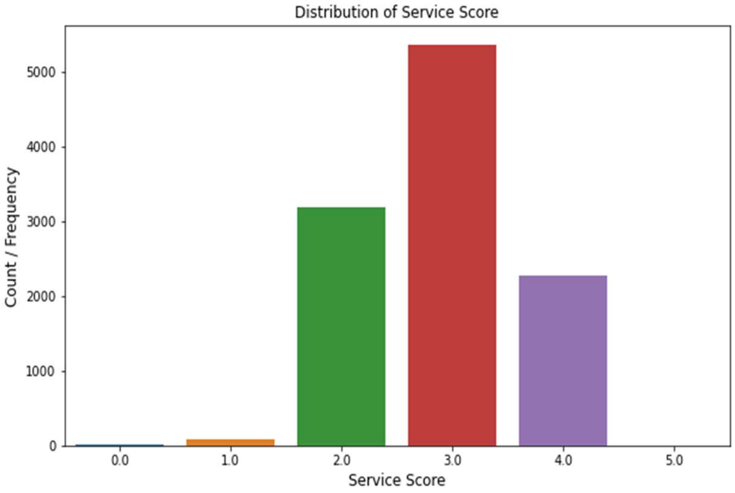


Figure 8: Distribution for Service Score

5. Account User count

Following inferences can be drawn by seeing the plot for Account user count.

- Around 72% of accounts are having 3 & 4 number of users per account and most of the accounts are having more than 3 users per account, which means that most of the accounts are having multiple users. Maximum users per account being 6.
- Only few accounts are having 2 and less than users per account.

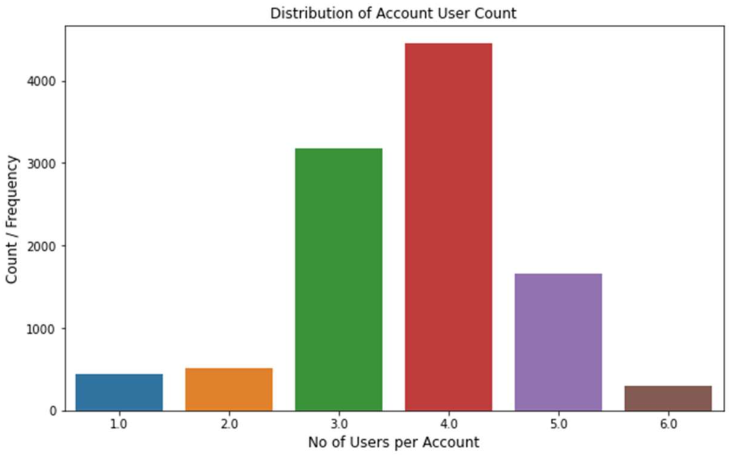


Figure 9: Distribution plot for No of Users per Account

6. Account Segment

Following inferences can be drawn by seeing the plot for Account Segment.

- Account segment consists of five classes, Regular, Regular Plus, Super, Super Plus & HNI which are divided on basis of spend.
- Around 41% of accounts are of Regular Plus and Regular type of accounts, which is a basic kind of account.
- Around 44% of accounts are of Super & Super Plus type of accounts, which might be a premium account.
- Around 15% of accounts are of HNI type, which means that they are high net worth individual or the highest segment of the account type

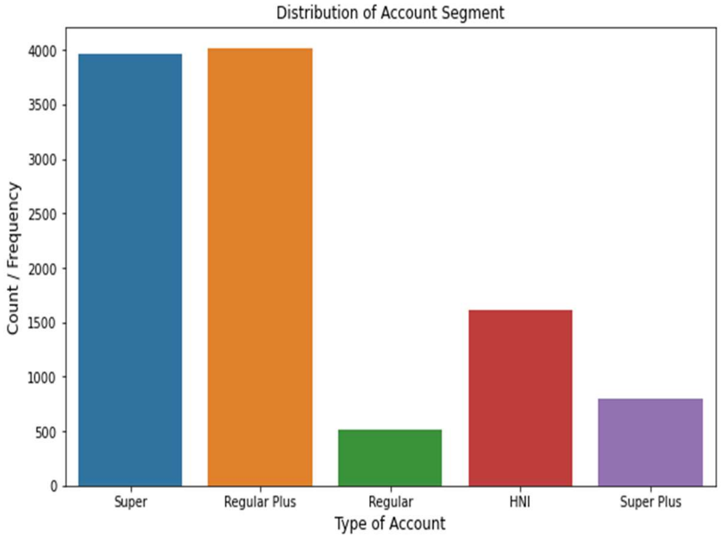


Figure 10: Distribution plot for Account Segment

7. CC Agent Score

Following inferences can be drawn by seeing the plot for CC Agent Score

- Around 69% of primary customers have given a satisfaction score of 3.0 and above for the customer care service provided by the company. Most of the customers are satisfied by the customer care service.
- Around 30% of primary customers have given a satisfaction score of 2.0 and below for the customer care service provided by the company. Most of the 30% have given a score of 1.0 which tells us these customers are not satisfied by the customer care service

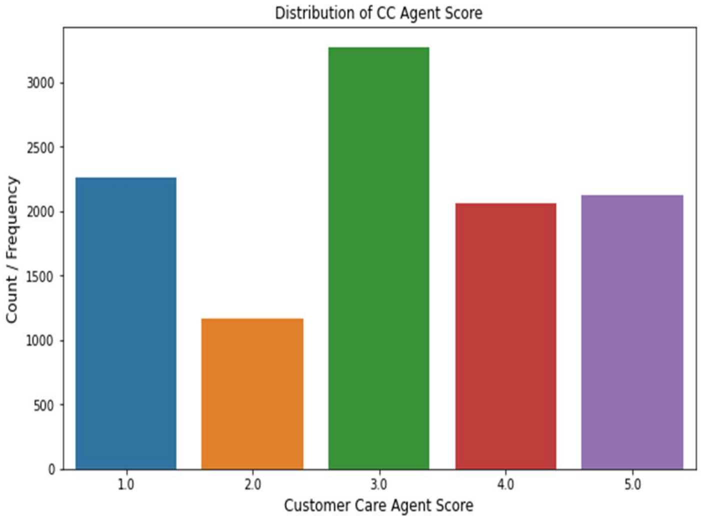


Figure 11: Distribution Plot for Customer Care Agent Score

8. Marital Status

Following inferences can be drawn by seeing the plot for Marital Status

- Around 53% of primary users are married, which tells us that majority of the users are married and there might be chances that each account can have multiple users.
- Around 33 % of primary users are single.
- 13% of primary users are divorced.

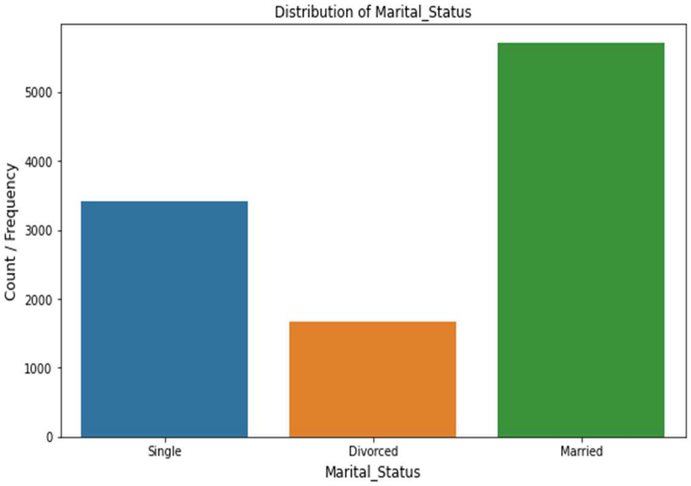


Figure 12: Distribution plot for Marital Status

9. Complain_ly

Following inferences can be drawn by seeing the plot for Complain_ly

- It is a binary variable where '0' implies that customer has not complained and '1' implies that customer has complained.
- No complaints were raised by 71% of the accounts during the last 12 months/ 1 year.
- Around 28% of accounts raised complaints during the last 12 months / 1 year.
- Shows that majority of the accounts didn't lodge any complaint which means there might be no issues with the service.

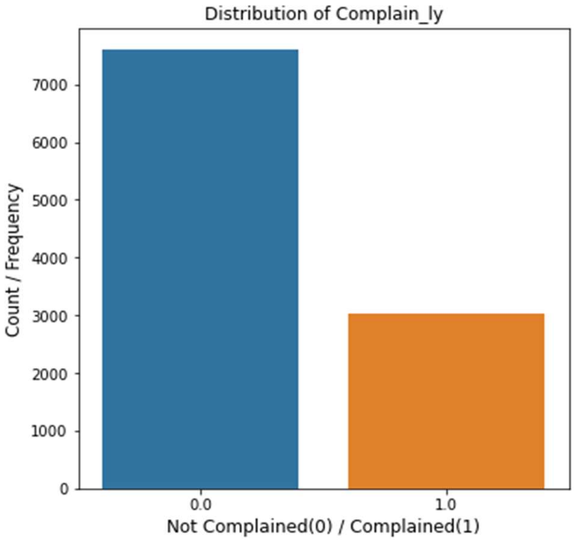


Figure 13: Distribution plot for Complain Status

10. Login Device

Following inferences can be drawn by seeing the plot for Login Device

- Most of the customers preferred 'Mobile' as preferred login device in the account. That is around 71% of the customers uses mobile phones as preferred login device.
- Around 21% of the customers preferred 'Computer' as preferred login device in the account.
- Since nowadays most of the people in the country are having smart phones, hence the use of mobile phones for login is high.

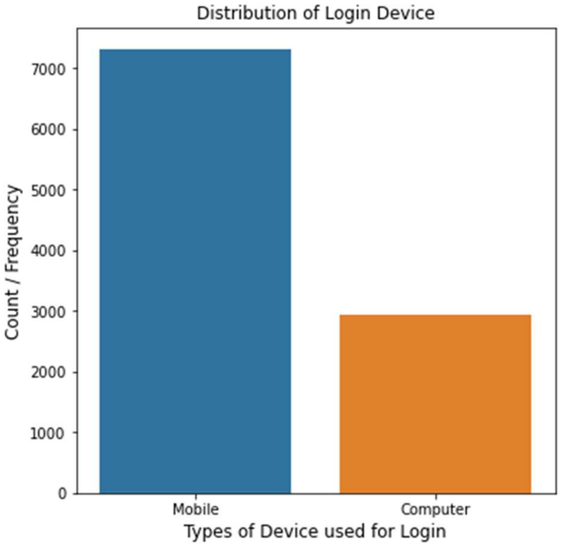


Figure 14: Distribution plot for Login Device

c) Bi-Variate Analysis

(i) Categorical / Discrete Variables

1. Payment V/s Churn

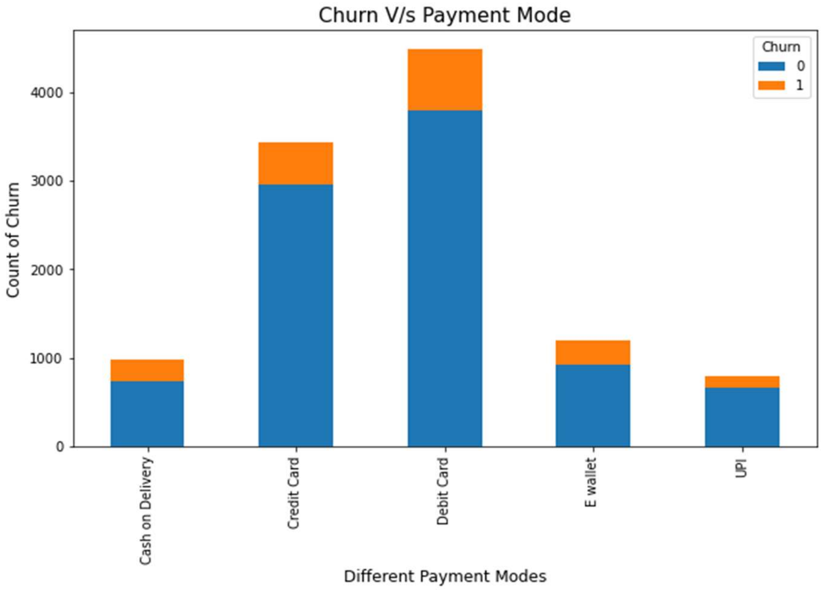


Figure 15: Payment Mode V/s Churn

Following inferences can be drawn by seeing the above plot,

- Majority of the transactions made by the customers are through Debit Card & Credit Card. Least number of transactions were made through Cash on delivery, E-wallet & UPI.
- Customers who preferred Debit Card & Credit Card, churned more when compared to other mode of transactions. Similarly, customers whose preferred mode of payment was UPI churned less.

2. Account Segment V/s Churn

Following inferences can be drawn by seeing the plot,

- Account segment consists of five classes, Regular, Regular Plus, Super, Super Plus & HNI which are segmented on the basis of spend.
- We can see majority of the accounts are of Regular Plus & Super.
- Around 25% of customers who are having Regular Plus account and 10% of customers who are having Super account have churned, which need to be checked as this is mid-level kind of account.
- Customers who have Regular & Super Plus account churned less when compared to other three types of account.

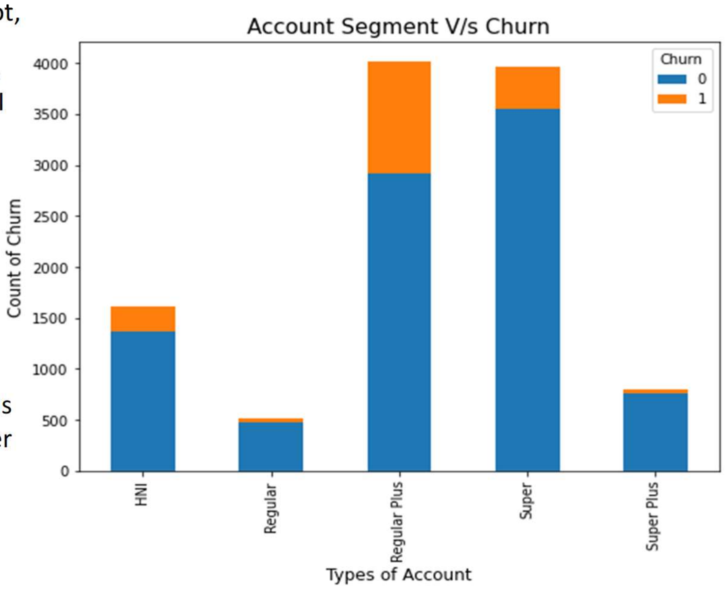


Figure 16: Account Segment V/s Churn

3. Gender V/s Churn

Following inferences can be drawn by seeing the plot,

- Majority of the primary user who owns the account is Male and rest of them are Female.
- We see that most of the male customers churned more when compared to Female.
- Customers who churned are less when compared to customers who retained. Majority of customers both Male & Female retained.

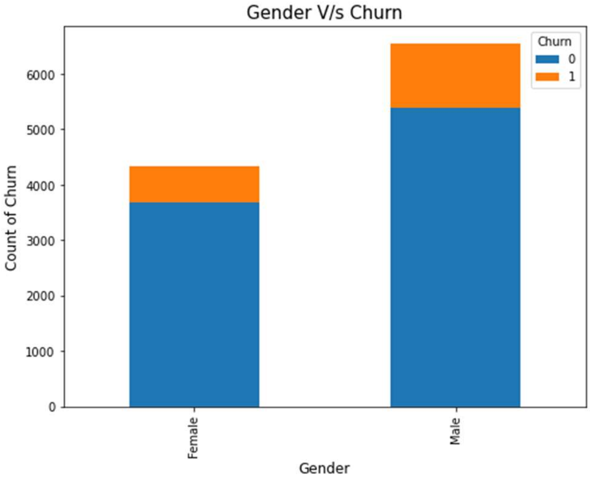


Figure 17: Gender V/s Churn

4. Marital Status V/s Churn

Following inferences can be drawn by seeing the plot,

- Majority of the primary customers are married.
- Primary customers who are single churned out more when compared to Married & Divorced. Most of the customers who are single might tend to share accounts.
- Primary customers who are divorced churned very less.

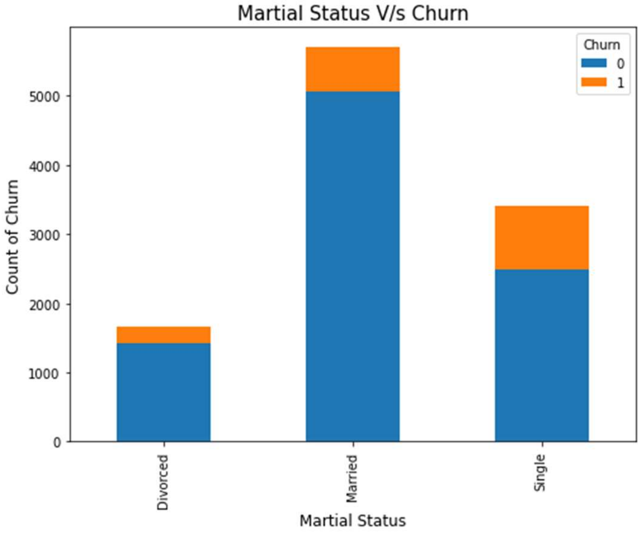


Figure 18: Stacked Bar Plot for Marital Status V/s Churn

5. Login Device V/s Churn

Following inferences can be drawn by seeing the plot,

- Most of the accounts preferred Mobile phones as their preferred login devices.
- Customers who preferred Mobile phones churned more when compared to customers who preferred computer.
- Around 14% of customers using mobile phones churned. Although customers who retained are more when compared to churned.

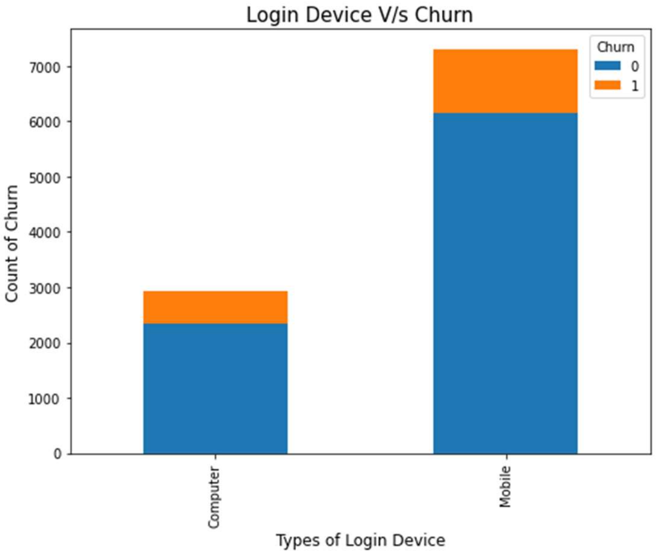


Figure 19: Stacked Bar Plot for Login Device V/s Churn

6. Complain_ly V/s Churn

Following inferences can be drawn by seeing the plot,

- Majority of the customers did not raise any complaint for the past twelve months/ 1 year.
- Although few customers churned who didn't raise any complaint in the last twelve months / 1 year.
- Customers who raised complained during the past twelve months / 1 year churned more. Around 25% of customer who complained have churned. This is higher when compared with customers who didn't complain but churned.

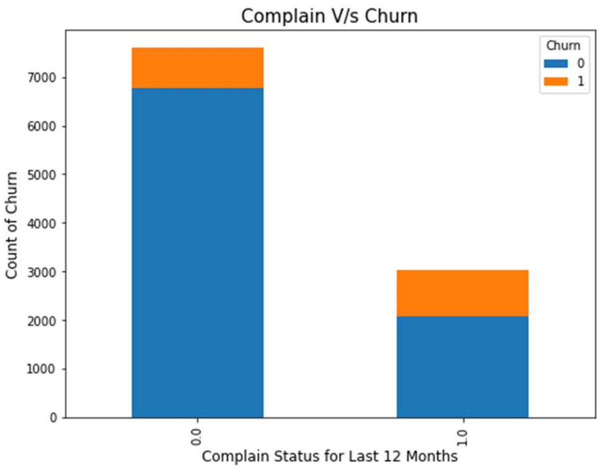


Figure 20: Stacked Bar Plot for Complain Status V/s Churn

7. City Tier V/s Churn

Following inferences can be drawn by seeing the plot,

- Majority of the customers are from Tier – 1 cities. Number of customers churned from Tier -1 cities are more when compared to other Tier cities.
- There are significantly few customers who are from Tier – 2 cities and the customer who churned are also less when compared to customer who retained.
- Around 15% of customers who are from Tier – 1 cities churned and around 20% of customers who are from Tier – 3 cities churned.

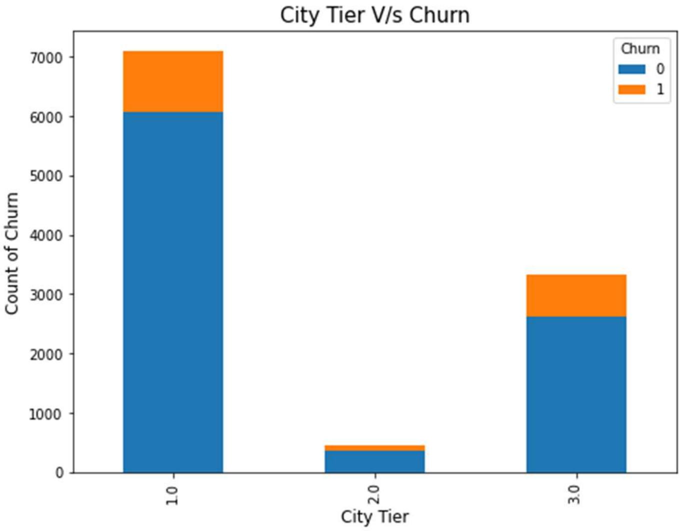


Figure 21: Stacked Bar Plot for City Tier V/s Churn

8. Service Score V/s Churn

Following inferences can be drawn by seeing the plot,

- Majority of the customers have given a satisfactory score of 3.0 and above for the service provided by the company.
- Similarly, customers who has given score of 3.0 churned more when compared to customers who has given scores other than 3.0.
- Customers who have given score of 2.0 also churned more when compared to customers who has given score of 4.0.

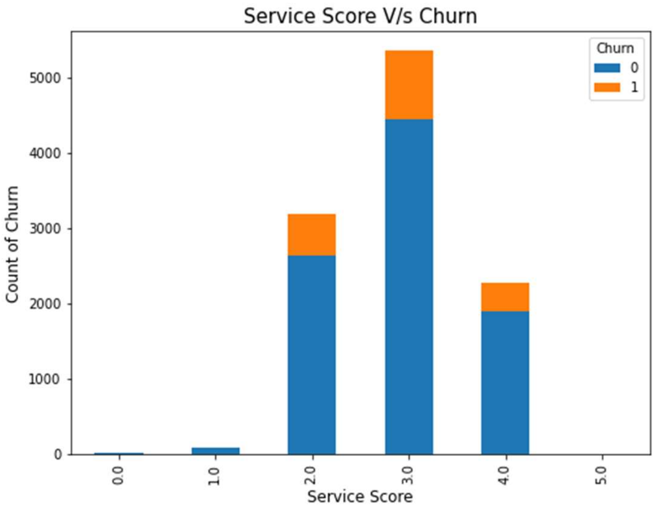


Figure 22: Stacked Bar Plot for Service Score V/s Churn

9. Account User count V/s Churn

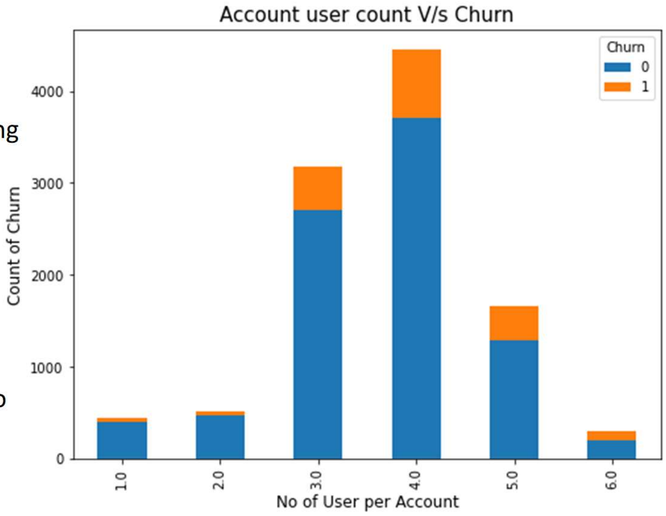


Figure 23: Stacked Bar Plot for No of Users Per Account V/s Churn

Following inferences can be drawn by seeing the plot,

- Most of the primary users of the account are having more than three users per account.
- Primary users who are having four users per account churned more when compared to others.
- Primary users who are having two or less than two users per account churned less.
- In total primary users who are having more than three users per account churned more when compared to primary users who are having less two users per account.

10. CC Agent Score V/s Churn

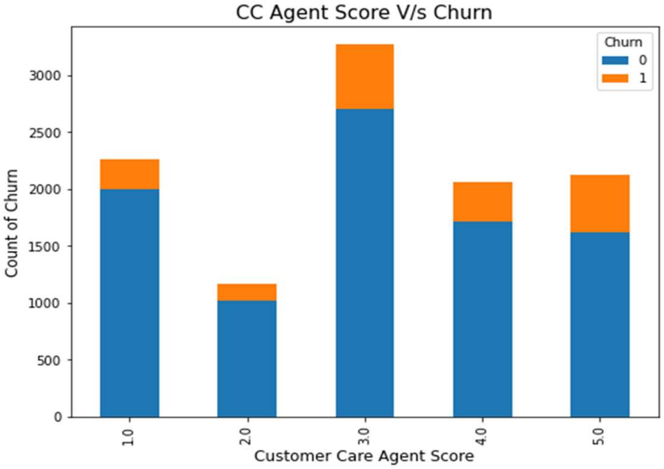


Figure 24: Stacked Bar Plot for Customer Care Agent Score V/s Churn

Following inferences can be drawn by seeing the plot,

- Most of the customers have given a satisfactory score 3.0 and above to the customer care provided by the company.
- Customers who have given a satisfactory score of 3.0 and 5.0 have churned more when compared to others. Overall customers who have given score 3.0 and above churned more when compared to others.
- Customers who have given a satisfactory score of 2.0 and less than 2.0 have churned less when compared to others

(ii) Continuous Variables

1. Tenure V/s Churn

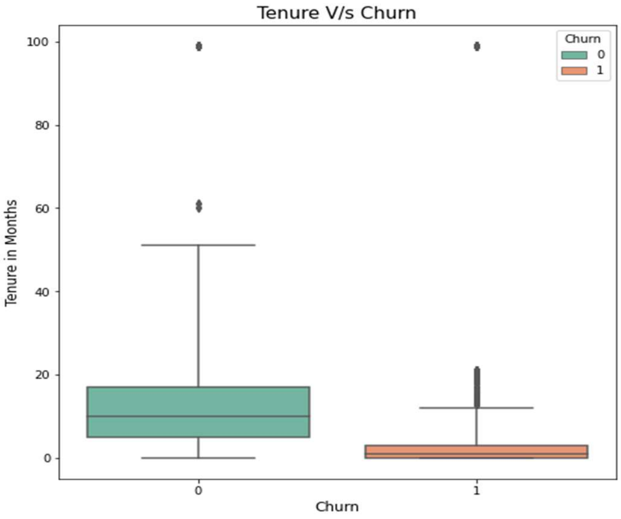


Figure 25 Box Plot for Tenure V/s Churn

Following inferences can be drawn by seeing the plot,

- Most of the customers whose account tenure is less than five months have churned and only few whose account tenure is more than five months have churned.
- Average account tenure for customers who have churned is around 1~2 months.
- Average account tenure for customers who did not churn is around 10 months.

2. Contacted Customer Care V/s Churn

Following inferences can be drawn by seeing the plot,

- Majority of times the customer has contacted the customer care is between 0 to 20 times.
- There is not much significant difference in the data spread between the customers who have churned and who have not churned.
- Average number of times the customer contacted customer care for both churned and non-churned customer is almost same i.e., between 15 to 18 times.

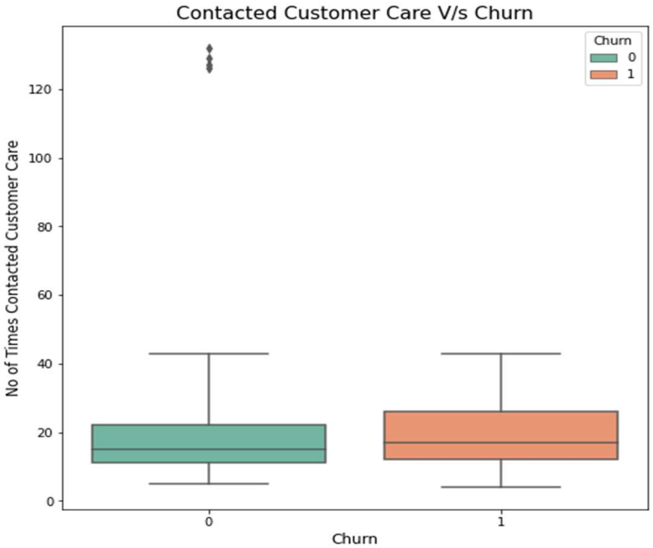


Figure 26: Box Plot for No of Times Contacted Customer Care V/s Churn

3. Revenue per Month V/s Churn

Following inferences can be drawn by seeing the plot,

- Majority of the revenue per month generated from customer is between 0 to 18 thousand, whereas the average revenue per month generated is around 5 thousand.
- There is not much significant difference in the data spread between the customers who have churned and who have not churned. Most of the customers whose revenue per month is between 0 to 5 thousand churned
- Few exceptions can be observed whose revenue per month is greater than 100 thousand, which might be rare occurrences or extreme values which don't tend to repeat frequently.

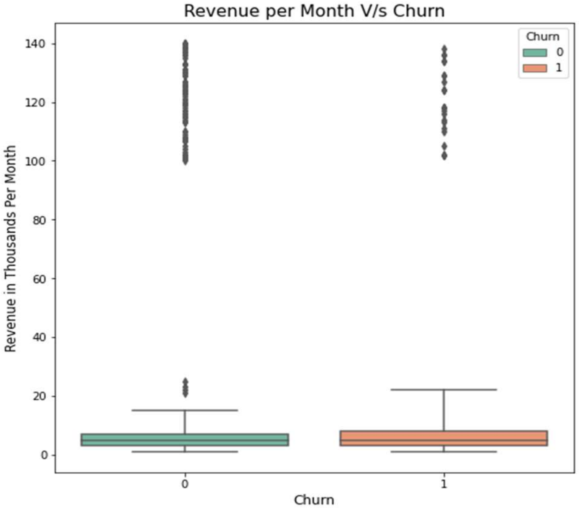


Figure 27: Box Plot for Revenue per Month V/s Churn

4. Revenue Growth V/s Churn

Following inferences can be drawn by seeing the plot,

- Majority of the revenue growth of the account is between 12 to 18 %
- Average revenue growth percentage of customer who has churned and who has not churned is almost same i.e., around 15%.
- Revenue growth for customer who has churned range from 12% to 16%.

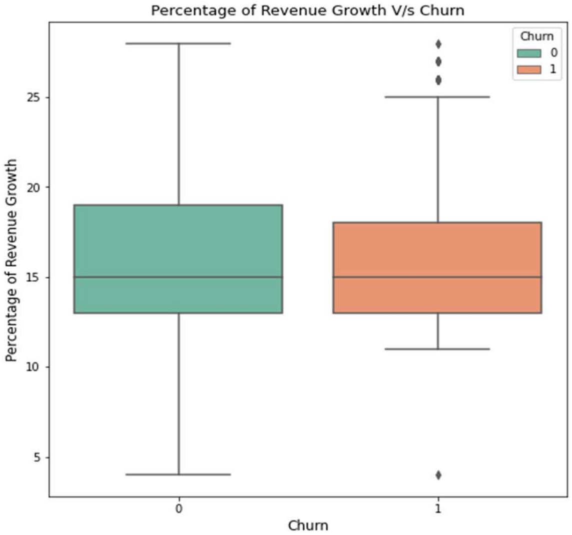


Figure 28: Box Plot for Revenue Growth V/s Churn

Days Since Contacted CC V/s Churn

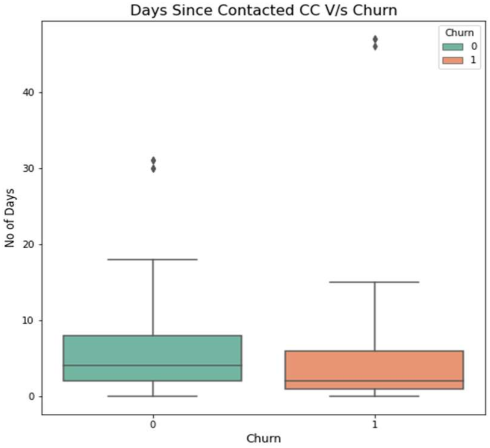


Figure 29: Box Plot for Days Since Contacted CC V/s Churn

Following inferences can be drawn by seeing the plot,

- Most of the customers who last contacted customer care is between 0 to 10 days.
- Few exceptions can be observed for both customers who have churned and who have not churned, which might be rare occurrences.
- Customers who last contacted customer care between 0 to 5 days have churned. Where the average days for customer who churned is around 2~3 days.
- Average days for customer who has not churned is around 5 days.

5. Cashback V/s Churn

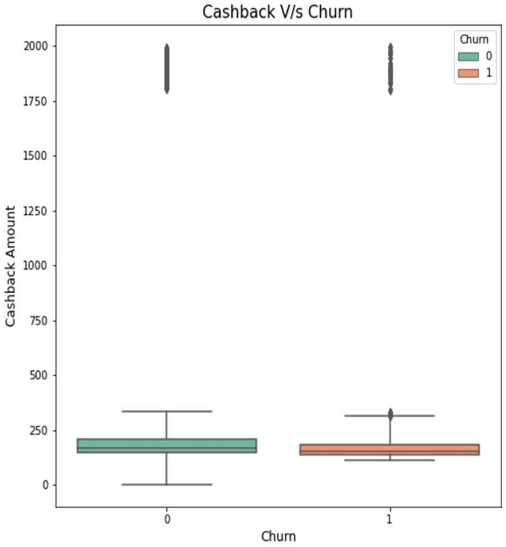


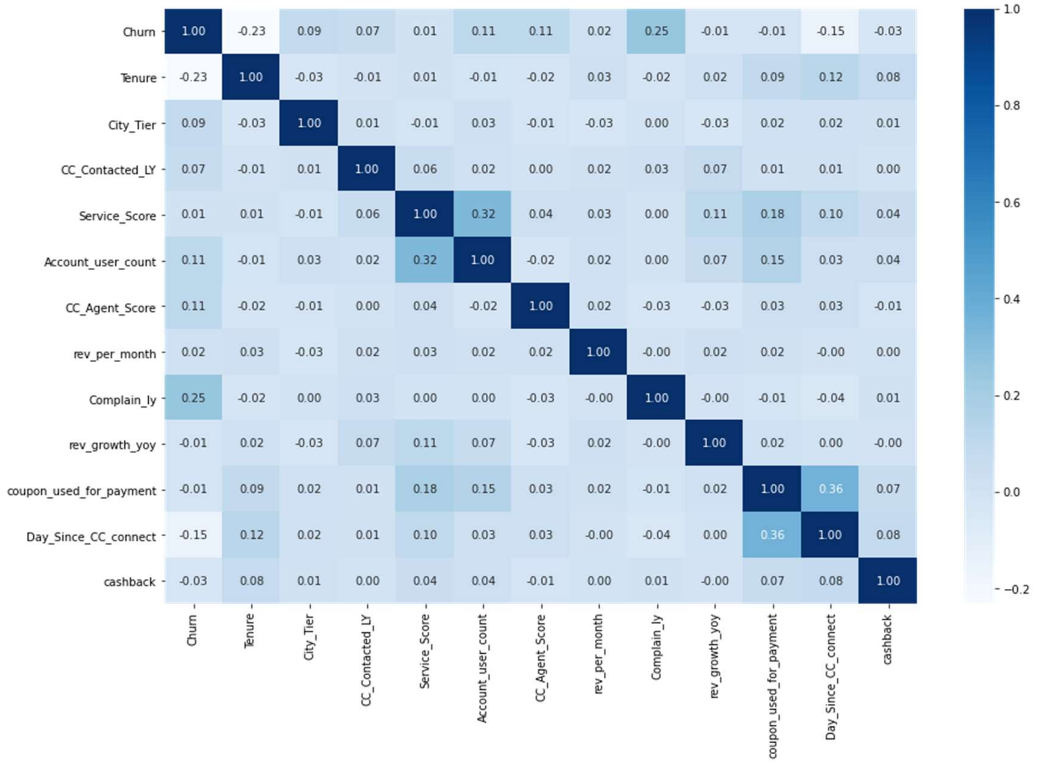
Figure 30: Boxplot for Cashback V/s Churn

Following inferences can be drawn by seeing the plot,

- Most of the customers got cashback between 100 to 300 Rupees, except for few customers who got cashback between 1800 to 2000 Rupees which is extreme values.
- There is not much significant difference in the data spread between the customers who have churned and who have not churned.
- Average cashback received for customer who has churned is around 100 to 110 Rupees which is almost equal to average cashback received by customer who has not churned.

d) Correlation Plot

From the below correlation plot (Pearson Method) we can see that there is not much of correlation between the independent variables, maximum correlation (with respect to the plot) observed is between Coupons used for payment and days since cc connect which is 0.36. Actually, in the correlation plot/matrix values range from -1 to +1 where negative symbol indicates inversely correlated and vice versa



	Churn	Tenure	City_Tier	CC_Contacted_LY	Service_Score	Account_user_count	CC_Agent_Score	rev_per_month	Complain_ly	rev_growth_yoy	coupon_used_for_payment	Day_Since_CC_connect	cashback
Churn	1.00	-0.23	0.09	0.07	0.01	0.11	0.11	0.02	0.25	-0.01	-0.01	-0.15	-0.03
Tenure	-0.23	1.00	-0.03	-0.01	0.01	-0.01	-0.02	0.03	-0.02	0.02	0.09	0.12	0.08
City_Tier	0.09	-0.03	1.00	0.01	-0.01	0.03	-0.01	-0.03	0.00	-0.03	0.02	0.02	0.01
CC_Contacted_LY	0.07	-0.01	0.01	1.00	0.06	0.02	0.00	0.02	0.03	0.07	0.01	0.01	0.00
Service_Score	0.01	0.01	-0.01	0.06	1.00	0.32	0.04	0.03	0.00	0.11	0.18	0.10	0.04
Account_user_count	0.11	-0.01	0.03	0.02	0.32	1.00	-0.02	0.02	0.00	0.07	0.15	0.03	0.04
CC_Agent_Score	0.11	-0.02	-0.01	0.00	0.04	-0.02	1.00	0.02	-0.03	-0.03	0.03	0.03	-0.01
rev_per_month	0.02	0.03	-0.03	0.02	0.03	0.02	0.02	1.00	-0.00	0.02	0.02	-0.00	0.00
Complain_ly	0.25	-0.02	0.00	0.03	0.00	0.00	-0.03	-0.00	1.00	-0.00	-0.01	-0.04	0.01
rev_growth_yoy	-0.01	0.02	-0.03	0.07	0.11	0.07	-0.03	0.02	-0.00	1.00	0.02	0.00	-0.00
coupon_used_for_payment	-0.01	0.09	0.02	0.01	0.18	0.15	0.03	0.02	-0.01	0.02	1.00	0.36	0.07
Day_Since_CC_connect	-0.15	0.12	0.02	0.01	0.10	0.03	0.03	-0.00	-0.04	0.00	0.36	1.00	0.08
cashback	-0.03	0.08	0.01	0.00	0.04	0.04	-0.01	0.00	0.01	-0.00	0.07	0.08	1.00

Figure 31: Correlation Plot Between Variables (Pearson Method)

3. Data Cleaning & Preprocessing:

a) Missing Value Treatment:

From the below table we can see that all of the variables except for our Target Variable (Churn) are having more than 1 missing values. These missing values need to be treated before building our model. Hence, we will be splitting our data-frame into two as mentioned below because the treatment of missing values will also differ.

- Categorical Variables
- Continuous & Discrete Variables

S.No	Variable Description	Total	Percent
1	rev_per_month	791	7.19%
2	Login_device	760	6.91%
3	cashback	473	4.30%
4	Account_user_count	444	4.04%
5	Day_Since_CC_connect	358	3.26%
6	Complain_ly	357	3.25%
7	Tenure	218	1.98%
8	Marital_Status	211	1.92%
9	CC_Agent_Score	116	1.05%
10	City_Tier	112	1.02%
11	Payment	109	0.99%
12	Gender	108	0.98%
13	CC_Contacted_LY	102	0.93%
14	Service_Score	98	0.89%
15	account_segment	97	0.88%
16	rev_growth_yoy	3	0.03%
17	coupon_used_for_payment	3	0.03%
18	Churn	0	0.00%

Table 7:List of Variables with Percentage of Missing Values

(i) Treating Missing Values for Categorical Variables

S.No	Categorical Variable	Total	Percent
1	Login_device	760	6.91%
2	Marital_Status	211	1.92%
3	Payment	109	0.99%
4	Gender	108	0.98%
5	account_segment	97	0.88%

Table 8: Missing Values in Categorical Variable

Here Since the most the missing value is less than 2% except for Login Device, we will be imputing the missing values with a random category 'Unknown'. Because if we impute these missing values with mode then data will be more skewed towards the majority % present in the variable which will affect the model. Hence, we will be assigning these missing values as unknown.

(ii) Treating Missing Values for Continuous & Discrete Variables

Missing values for continuous & discrete variables can be treated in numerous ways/methods, in which most common are mentioned below

1. Dropping the missing values if overall % of missing values is less than 1%.
2. Replacing the missing values with Mean or Median whichever is better.
3. Using regression-based models to predict the missing values.
4. Using KNN imputer to replace missing values

In the business point of view, we cannot drop the missing values and similarly we cannot impute missing values with Mean or Median because if we impute the missing values with mean & median, the data will tend to be more skewed towards these values which will impact while building our predictive model.

Hence in our case we will be considering k-Nearest Neighbors Imputation which is more accurate way of treating the missing values.

- First, we will initially import KNN Imputer from sklearn library
- KNNImputer is a data transform that is first configured based on the method used to estimate the missing values.
- The default distance measure is a Euclidean distance measure that is NaN aware, e.g., will not include NaN values when calculating the distance between members of the dataset.
- The number of neighbors is set to five by default and can be configured by the "n_neighbors" argument.
- Then, the imputer is fit & transformed on the dataset (Cont & Disc).
- Each missing value will be replaced with a value estimated by the model.

S.No	Variable Description	Total	Percent
1	Churn	0	0%
2	Tenure	0	0%
3	Marital_Status	0	0%
4	account_segment	0	0%
5	Gender	0	0%
6	Payment	0	0%
7	cashback	0	0%
8	Day_Since_CC_connect	0	0%
9	coupon_used_for_payment	0	0%
10	rev_growth_yoy	0	0%
11	Complain_ly	0	0%
12	rev_per_month	0	0%
13	CC_Agent_Score	0	0%
14	Account_user_count	0	0%
15	Service_Score	0	0%
16	CC_Contacted_LY	0	0%
17	City_Tier	0	0%
18	Login_device	0	0%

Table 9: List of Missing Values after Treating Missing Values

From the above table we can see that all our variables having missing values have been treated and there is no presence of missing values.

b) Outlier Treatment (For Continuous Variables):

From the below boxplot we can see that there is presence of outliers in the continuous variables, but in boxplot function by default whiskers are capped at 1.5 (generally called as inner hinge), hence we will be checking boxplot again by changing the whisker to 3.0 (generally called as outer hinge) because values beyond this rarely occur and will be considered as extreme values which do not occur frequently.

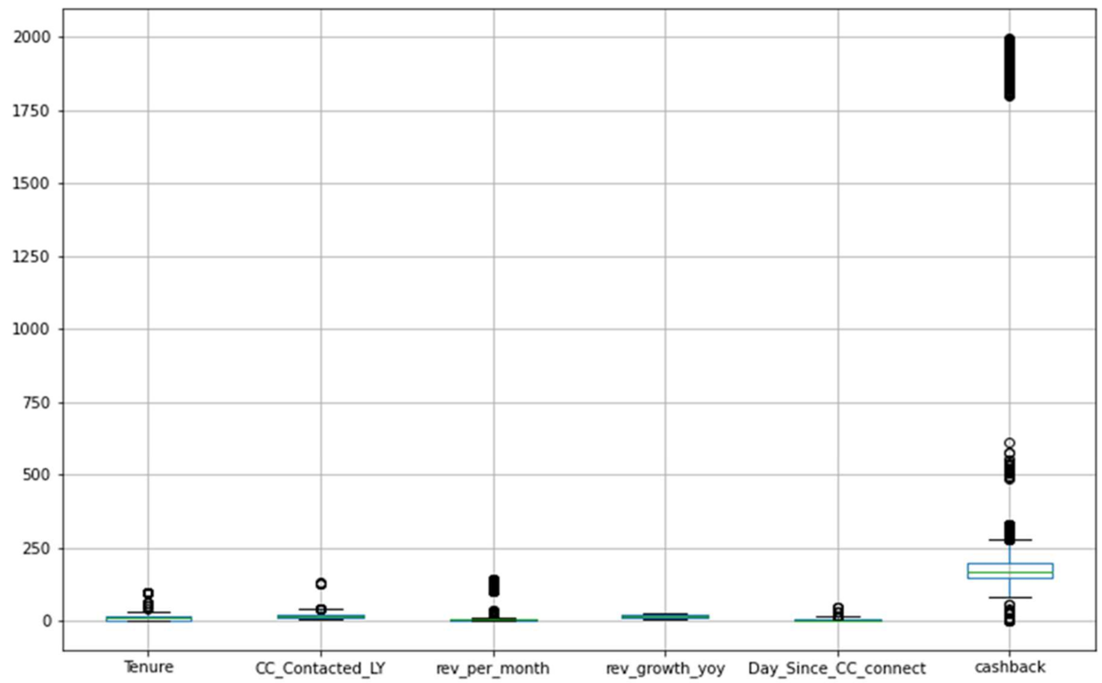


Figure 32: Box Plot for Checking Outliers in Continuous Variables (With Whis = 1.5)

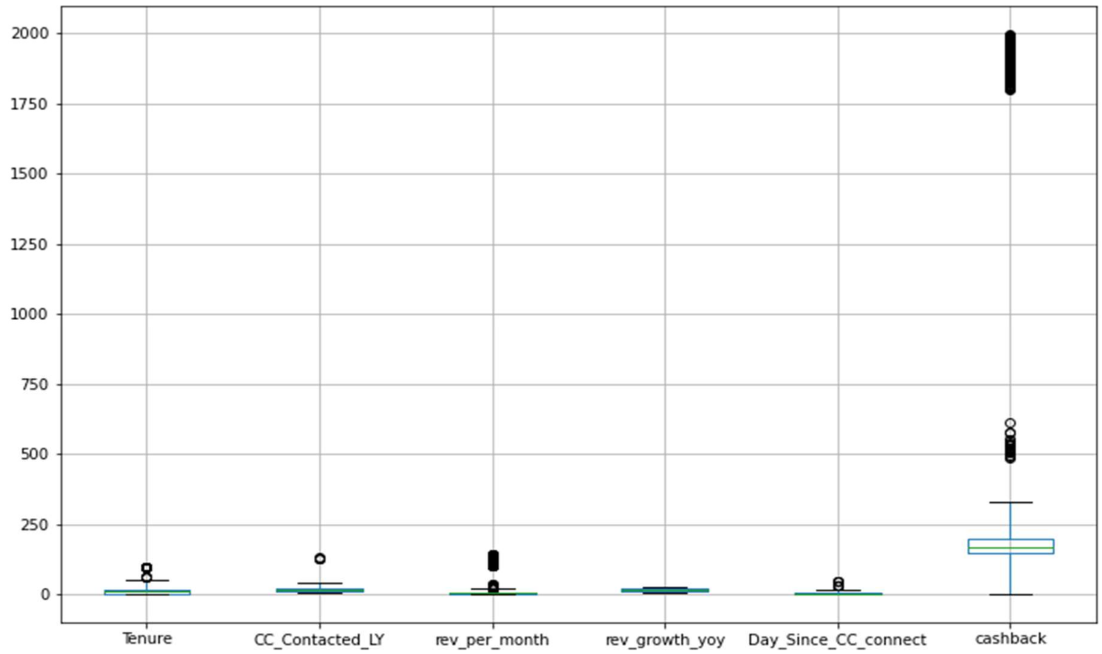


Figure 33: Box Plot for Checking Outliers in Continuous Variables (With Whis = 3.0)

After changing the whis = 3.0 still we observe outliers in following variables, which tell us that the outliers are beyond the whis = 3.0 are extreme values and need to be treated.

- Tenure
- CC_Contacted_LY
- rev_per_month
- Day_Since_CC_connect
- Cashback

The extreme values identified for the above variables will be treated with IQR method i.e., we will be capping those extreme values to the maximum range i.e. (Q3 + 3.0 x IQR) & minimum range i.e. (Q1 - 3.0 x IQR).

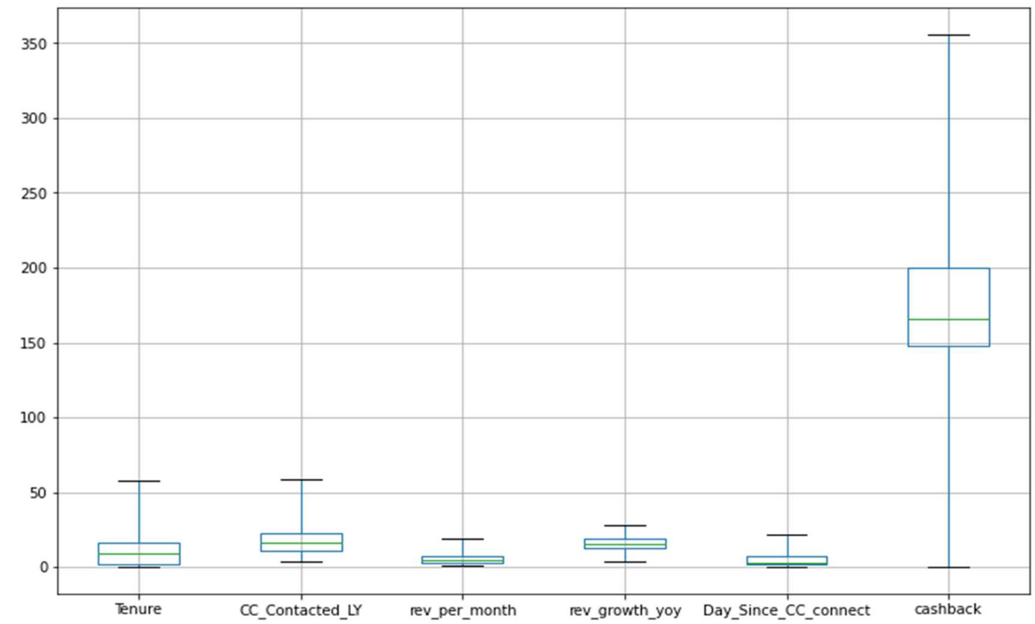


Figure 34: Box Plot after Treating Outliers in Continuous Variables (With Whis = 3.0)

After treating the outliers, we can see that there is no presence of outlier further and our data is now good for building the model

c) Variable Encoding / Transformation:

S.No	Column Name	Non-Null Count	Data Type
1	Churn	10996	Continuous
2	Tenure	10996	Continuous
3	City_Tier	10996	Continuous
4	CC_Contacted_LY	10996	Continuous
5	Service_Score	10996	Continuous
6	Account_user_count	10996	Continuous
7	CC_Agent_Score	10996	Continuous
8	Rev_per_month	10996	Continuous
9	Complain_ly	10996	Continuous
10	Rev_growth_yoy	10996	Continuous
11	Coupon_used_for_payment	10996	Continuous
12	Day_Since_CC_connect	10996	Continuous
13	Cashback	10996	Continuous
14	Payment	10996	Continuous
15	Gender	10996	Categorical
16	Account_segment	10996	Categorical
17	Marital_Status	10996	Categorical
18	Login_device	10996	Categorical

Table 10: List of Variables & Their Data Type

By looking at the data we can see that there are few variables which are of categorical type. Before proceeding to model building, we need to convert these categorical variables into numerical since models cannot treat categorical. Hence, we will be doing label encoding with these categorical variables.

Label Encoding: Label Encoding is a popular encoding technique for handling categorical variables. In this technique, each label is assigned a unique integer based on alphabetical ordering or ordinal.

- First, we will be importing the LabelEncoder library from sklearn.
- Initially we will identify the features which need to be converted.
- We will be checking the unique counts in each feature.
- Create LabelEncoder and fit & transform to the features which need to be converted.
- Let’s check the top 5 observations after doing label encoding.

Churn	Tenure	City_Tier	CC_Contacted_LY	Service_Score	Account_user_count	CC_Agent_Score	rev_per_month	Complain_ly	rev_growth_yoy	coupon_used_for_payment	Day_Since_CC_connect	cashback	Payment	Gender	account_segment	Marital_Status	Login_device
1	4	3	6	3	3	2	9	1	11	1	5	160	2	0	3	2	1
1	0	1	8	3	4	3	7	1	15	0	0	121	4	1	2	2	1
1	0	1	30	2	4	3	6	1	14	0	3	164	2	1	2	2	1
1	0	3	15	2	4	5	8	0	23	0	3	134	2	1	3	2	1
1	0	1	12	2	3	5	3	0	11	1	3	130	1	1	2	2	1

Table 11: Top 5 Observations after doing Label Encoding

4. Splitting the data set:

- First, we need to separate the dependent & independent variable, i.e., we will be dropping the Target variable from the dataset and assign the dataset to variable ‘X’ and pop the target variable in another variable ‘y’.
- Now we will split the data set into Train & Test, where the train set consists of 70% of the data and test set consists of 30% of data. These are randomly distributed with random_state equal to 3 and stratify = True, which equally distributes the proportion of target variable into Train & Test dataset.
- We will be building the model based on the train set and check the performance on test set.
- After splitting the data, we can see that
    - ✓ X_train consists of 7,697 Observations & 17 Variables
    - ✓ X_test consists of 3,299 Observations & 17 Variables
- Checking the percentage unique values of the Target variables in Train & Test
    - ✓ Class 0 is of 83.16%
    - ✓ Class 1 is of 16.83%

5. Model Building:

We have opted for 6 models with around 22 combinations, which includes model with hyperparameter tuning, SMOTE etc.

- K-Nearest Neighbors
- Random Forest Classifier
- XG Boost
- Artificial Neural Networks
- Logistic Regression
- Support Vector Classifier

a) Building Model with Default Parameters

(i) K-Nearest Neighbors (KNN):

- KNN or K-Nearest Neighbors is basically a distance-based algorithm, which means that KNN classifier divides the target variable into classes and checks the distance between the existing class data point and new incoming data point. New data point will be labeled to Class which has minimum distance.
- Since it's a distance-based algorithm, all the data points need to be scaled except for the target variable. Hence here we will be using MinMax scaler.
- From sklearn library we will be import MinMax Scaler function.
- Define the MinMaxScaler and assign to a variable.
- Further we will be doing a fit & transform function to Train data set and do a transform function to the test data set in order to avoid data leakage.
- After scaling we will be importing the KNN model from sklearn library
- Initially we will be creating KNN model with default parameters.
- After creating the model, we will fit the model to the training set.
- Once we fit the model to the training set, next we do prediction on Train & Test dataset.
- Now we check the Performance Metrics for Class 1 Label

Train Dataset					**Test Dataset**				
Accuracy Score	**Precision**	**Recall**	**F1 Score**	**AUC Score**	**Accuracy Score**	**Precision**	**Recall**	**F1 Score**	**AUC Score**
97.21	95.00	88.00	91.00	99.40	93.96	87.00	75.00	81.00	96.07

Table 12: Train & Test Data Performance Metrics of Class 1 for KNN with Default Parameters

- Interpretation from the KNN Model (Default Parameters)
  - ✓ From the Train set we can see that precision is of 0.95 & recall is of 0.88 for Class 1
  - ✓ From the Test set we can see that precision is of 0.87 & recall is of 0.75 for Class 1
  - ✓ Dataset might be imbalanced, further we will check with another model.

(ii) Random Forest Classifier (RFCL):

- For Random Forest Classifier scaling is not required, hence we will be using normal dataset.
- Importing Random Forest Classifier from sklearn library
- First, we will build random forest classifier model with default parameters
- After creating the model, we will fit the model to the training set.
- Once we fit the model to the training set, next we do prediction on Train & Test dataset.
- Now we check the Performance Metrics for Class 1 Label

Train Dataset					Test Dataset				
Accuracy Score	**Precision**	**Recall**	**F1 Score**	**AUC Score**	**Accuracy Score**	**Precision**	**Recall**	**F1 Score**	**AUC Score**
100.00	100.00	100.00	100.00	100.00	96.33	95.00	82.00	88.00	99.15

Table 13: Train & Test Data Performance Metrics of Class 1 for RFCL with Default Parameters

- Interpretation from the RFCL Model (Default Parameters)
  - ✓ From the Training set we can see that precision is of 1.00 & recall is of 1.00 for Class 1
  - ✓ From the Test set we can see that precision is of 0.95 & recall is of 0.82 for Class 1
  - ✓ Dataset might be imbalanced, further we will check with another model.

(iii) XG Boost Classifier:

- For XG Boost Classifier scaling is not required, hence we will be using normal dataset.
- Importing XG Boost Classifier
- First, we will build XG Boost Classifier model with default parameters
- After creating the model, we will fit the model to the train set.
- Once we fit the model to the training set, next we do prediction on Train & Test dataset.
- Now we check the Performance Metrics for Class 1 Label

Train Dataset					Test Dataset				
Accuracy Score	**Precision**	**Recall**	**F1 Score**	**AUC Score**	**Accuracy Score**	**Precision**	**Recall**	**F1 Score**	**AUC Score**
91.78	85.00	63.00	72.00	95.18	90.69	83.00	56.00	67.00	93.28

Table 14: Train & Test Data Performance Metrics of Class 1 for XG Boost with Default Parameters

- Interpretation from the XG Boost Model (Default Parameters)
  - ✓ From the Train set we can see that precision is of 0.85 & recall is of 0.63 for Class 1
  - ✓ From the Test set we can see that precision is of 0.83 & recall is of 0.56 for Class 1
  - ✓ Poor recall values for both Train & Test dataset. Further tuning of parameters to be done.

(iv) Artificial Neural Network (ANN):

- For Neural Network model, we will be using Scaled dataset.
- Importing MLP Classifier from sklearn.neural network library
- First, we will build the neural network model with default parameters
- After creating the model, we will fit the model to the train set.
- Once we fit the model to the train set, next we do prediction on Train & Test dataset.
- Now we check the Performance Metrics for Class 1 Label.

Train Dataset					Test Dataset				
Accuracy Score	**Precision**	**Recall**	**F1 Score**	**AUC Score**	**Accuracy Score**	**Precision**	**Recall**	**F1 Score**	**AUC Score**
93.14	85.00	72.00	78.00	96.25	90.78	79.00	61.00	69.00	93.47

Table 15: Train & Test Data Performance Metrics of Class 1 for ANN Model with Default Parameters

- Interpretation from the Neural Network Model (Default Parameters)
  - ✓ From the Training set we can see that precision is of 0.85 & recall is of 0.72 for Class 1
  - ✓ From the Test set we can see that precision is of 0.79 & recall is of 0.61 for Class 1
  - ✓ Poor recall values for both Train & Test dataset. Further tuning of parameters to be done.

(v) Logistic Regression (LR):

- For Logistic Regression, we will be using Scaled dataset.
- Importing Logistic Regression from sklearn library
- First, we will build Logistic Regression model with default parameters
- After creating the model, we will fit the model to the train set.
- Once we fit the model to the train set, next we do prediction on Train & Test dataset.
- Now we check the Performance Metrics for Class 1 Label.

Train Dataset					**Test Dataset**				
Accuracy Score	**Precision**	**Recall**	**F1 Score**	**AUC Score**	**Accuracy Score**	**Precision**	**Recall**	**F1 Score**	**AUC Score**
88.34	79.00	42.00	55.00	87.19	87.29	74.00	38.00	50.00	84.26

Table 16: Train & Test Data Performance Metrics of Class 1 for Logistic Regression with Default Parameters

- Interpretation from the Logistic Regression Model (Default Parameters)
  - ✓ From the Training set we can see that precision is of 0.79 & recall is of 0.42 for Class 1
  - ✓ From the Test set we can see that precision is of 0.74 & recall is of 0.38 for Class 1
  - ✓ Poor recall values for both Train & Test dataset. Further tuning of parameters to be done.

(vi) Support Vector Classifier (SVC):

- For Support Vector Classifier scaling is not required, hence we will be using normal dataset.
- Importing Support Vector Classifier from sklearn library.
- First, we will build Support Vector Classifier model with default parameters.
- After creating the model, we will fit the model to the train set.
- Once we fit the model to the train set, next we do prediction on Train & Test dataset.
- Now we check the Performance Metrics for Class 1 Label.

Train Dataset					**Test Dataset**				
Accuracy Score	**Precision**	**Recall**	**F1 Score**	**AUC Score**	**Accuracy Score**	**Precision**	**Recall**	**F1 Score**	**AUC Score**
98.89	99.00	94.00	97.00	99.95	87.20	84.00	30.00	44.00	91.39

Table 17: Train & Test Data Performance Metrics of Class 1 for SVC with Default Parameters

- Interpretation from the Support Vector Classifier Model (Default Parameters)
  - ✓ From the Training set we can see that precision is of 0.99 & recall is of 0.94 for Class 1
  - ✓ From the Test set we can see that precision is of 0.84 & recall is of 0.30 for Class 1
  - ✓ Poor recall values for both Train & Test dataset and can see there is imbalance in the dataset.

b) Building Model with Random Search:

(i) K-Nearest Neighbors (KNN):

- First, we will be importing RandomizedSearchCV library from sklearn
- Entering the parameters into the grid so as to find optimal parameters using random search.
- Dataset used here will be of scaled since it's a distance-based model.
- Next, we will be creating KNN model with default parameters.
- Building Random Search with a cross validation of 5 along with above defined parameters grid & model.
- After creating the model, we will fit the model to the train set.
- Now we will be checking the best parameters using best params function.
  - ✓ **Best Parameters :** {'weights': 'distance', 'p': 1, 'n_neighbors': 10, 'metric': 'minkowski', 'leaf_size': 30, 'algorithm': 'auto'}
- Creating the model with best parameters & next we do prediction on Train & Test dataset.
- Now we check for performance metrics for Class 1 Label.

Train Dataset					Test Dataset				
Accuracy Score	Precision	Recall	F1 Score	AUC Score	Accuracy Score	Precision	Recall	F1 Score	AUC Score
100.00	100.00	100.00	100.00	100.00	95.99	95.00	81.00	87.00	98.88

Table 18: Train & Test Data Performance Metrics of Class 1 for KNN with Hyperparameter Tunning Parameters

- Interpretation from the KNN Tuned Model
  - ✓ From the Train set we can see that precision is of 1.00 & recall is of 1.00 for Class 1
  - ✓ From the Test set we can see that precision is of 0.95 & recall is of 0.81 for Class 1
  - ✓ Poor recall values for both Train & Test dataset and can see there is imbalance in the dataset.

(ii) Random Forest Classifier:

- Entering the parameters into the grid so as to find optimal parameters using random search
- Next, we will be creating Random Forest model with default parameters.
- Building Random Search with a cross validation of 5 along with above defined parameters grid & model.
- After creating the model, we will fit the model to the train set.
- Now we will be checking the best parameters using best params function.
  - ✓ **Best Parameters :** {'n_estimators': 100, 'min_samples_split': 60, 'max_features': 4, 'max_depth': 10}
- Creating the model with best parameters & next we do prediction on Train & Test dataset.
- Now we check for performance metrics for Class 1 Label.

Train Dataset					Test Dataset				
Accuracy Score	Precision	Recall	F1 Score	AUC Score	Accuracy Score	Precision	Recall	F1 Score	AUC Score
92.17	90.00	60.00	72.00	96.65	90.60	85.00	54.00	66.00	94.68

Table 19: Train & Test Data Performance Metrics of Class 1 for RFCL with Hyperparameter Tunning Parameters

- Interpretation from the RFCL Tuned Model
  - ✓ From the Train set we can see that precision is of 0.90 & recall is of 0.60 for Class 1
  - ✓ From the Test set we can see that precision is of 0.85 & recall is of 0.54 for Class 1
  - ✓ Poor recall values for both Train & Test dataset and can see there is imbalance in the dataset.

(iii) XG Boost Classifier:

- Entering the parameters into the grid so as to find optimal parameters using random search
- Next, we will be creating XG Boost model with default parameters.
- Building Random Search with a cross validation of 5 along with above defined parameters grid & model.
- After creating the model, we will fit the model to the train set.
- Now we will be checking the best parameters using best params function.
  - ✓ **Best Parameters :** {'subsample': 1, 'n_estimators': 200, 'max_depth': 10, 'learning_rate': 0.01, 'gamma': 1, 'colsample_bytree': 0.8}
- Creating the model with best parameters & next we do prediction on Train & Test dataset.
- Now we check for performance metrics for Class 1 Label .

Train Dataset					Test Dataset				
Accuracy Score	Precision	Recall	F1 Score	AUC Score	Accuracy Score	Precision	Recall	F1 Score	AUC Score
98.09	100.00	89.00	94.00	99.73	94.63	94.00	73.00	82.00	98.19

Table 20: Train & Test Data Performance Metrics of Class 1 for XG Boost with Hyperparameter Tunning Parameters

- Interpretation from the XG Boost Tuned Model
  - ✓ From the Train set we can see that precision is of 1.00 & recall is of 0.89 for Class 1
  - ✓ From the Test set we can see that precision is of 0.94 & recall is of 0.73 for Class 1
  - ✓ We can slight improvement in recall values for both Train & Test dataset and can see there is imbalance in the dataset.

(iv) Artificial Neural Networks:

- Entering the parameters into the grid so as to find optimal parameters using random search.
- Dataset used here will be of scaled.
- Next, we will be creating Neural Network model with default parameters.
- Building Random Search with a cross validation of 5 along with above defined parameters grid & model.
- After creating the model, we will fit the model to the train set.
- Now we will be checking the best parameters using best params function.
    - ✓ **Best Parameters :** {'tol': 0.001, 'solver': 'adam', 'max_iter': 300, 'hidden_layer_sizes': 300, 'activation': 'relu'}
- Creating the model with best parameters & next we do prediction on Train & Test dataset.
- Now we check for performance metrics for Class 1 Label.

Train Dataset					Test Dataset				
Accuracy Score	Precision	Recall	F1 Score	AUC Score	Accuracy Score	Precision	Recall	F1 Score	AUC Score
95.06	92.00	78.00	84.00	98.22	92.02	85.00	64.00	73.00	95.23

Table 21: Train & Test Data Performance Metrics of Class 1 for ANN with Hyperparameter Tunning Parameters

- Interpretation from the Neural Network Tuned Model
    - ✓ From the Train set we can see that precision is of 0.92 & recall is of 0.78 for Class 1
    - ✓ From the Test set we can see that precision is of 0.85 & recall is of 0.64 for Class 1
    - ✓ We can slight improvement in recall values for both Train & Test dataset and can see there is imbalance in the dataset.

(v) Logistic Regression :

- Entering the parameters into the grid so as to find optimal parameters using random search.
- Dataset used here will be of scaled.
- Next, we will be creating Logistic Regression model with default parameters.
- Building Random Search with a cross validation of 5 along with above defined parameters grid & model.
- After creating the model, we will fit the model to the train set.
- Now we will be checking the best parameters using best params function
    - ✓ **Best Parameters** : {'tol': 0.00001, 'solver': 'lbfgs', 'penalty': 'none', 'max_iter': 300, 'class_weight': 'none'}
- Creating the model with best parameters & next we do prediction on Train & Test dataset.
- Now we check for performance metrics for Class 1 Label.

Train Dataset					Test Dataset				
Accuracy Score	Precision	Recall	F1 Score	AUC Score	Accuracy Score	Precision	Recall	F1 Score	AUC Score
88.61	78.00	46.00	57.00	87.37	87.63	74.00	41.00	53.00	84.53

Table 22: Train & Test Data Performance Metrics of Class 1 for Logistic Regression with Hyperparameter Tunning Parameters

- Interpretation from the Logistic Regression Tuned Model
    - ✓ From the Train set we can see that precision is of 0.78 & recall is of 0.46 for Class 1
    - ✓ From the Test set we can see that precision is of 0.74 & recall is of 0.41 for Class 1
    - ✓ We can see slight improvement in recall values for both Train & Test dataset and can see there is imbalance in the dataset.

c) Applying SMOTE (Synthetic Minority Oversampling Technique)

- From the pie chart we can see that around 16.8% of customers have churned and around 83.2% of customers have not churned. This tells us the given dataset is biased towards customers who have not churned.
- Hence if we build a predictive model, there might be a chance that the model also tend / show bias towards the customers who has not churned.
- Considering the business perspective there are three methods to deal with an imbalanced dataset they are

  - ✓ Under sampling
  - ✓ Over Sampling
  - ✓ Synthetic Minority Oversampling technique (SMOTE)

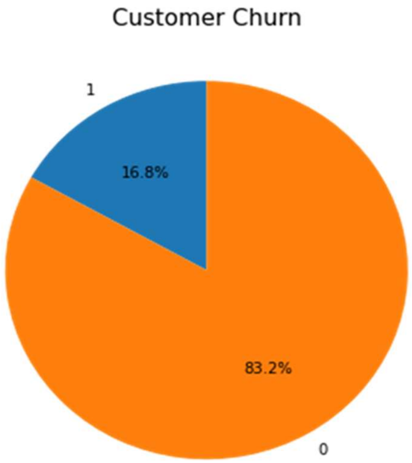


Figure 35: Pie Chart for Churn & Non-Churn

SMOTE is an oversampling technique where the synthetic samples are generated for the minority class. This algorithm helps to overcome the overfitting problem posed by random oversampling. It focuses on the feature space to generate new instances with the help of interpolation between the positive instances that lie together.

In our dataset we can see that there is an imbalance in the data set where the data is skewed towards the negative class i.e., customers who didn’t churn. Hence, we will be using SMOTE technique to our dataset and then build the models performance.

- We will be importing SMOTE from imblearn library.
- Creating SMOTE model with random state – 3.
- After creating the model, we will fit the model to Train set. (SMOTE will only be done to Train Set)
- We can see below the changes after doing SMOTE

  - ✓ Before OverSampling, counts of label '1': 1296
  - ✓ Before OverSampling, counts of label '0': 6401

  - ✓ After OverSampling, the shape of X_train: (12802, 17)
  - ✓ After OverSampling, the shape of y_train: (12802,)

  - ✓ After OverSampling, counts of label '1': 6401
  - ✓ After OverSampling, counts of label '0': 6401

d) Building Models with SMOTE & Default Parameters

(i) K-Nearest Neighbors (KNN):

- Initially we need to scale the resampled dataset before building the KNN model.
- After scaling, will be creating a KNN model with default parameters
- Fitting the model into Resampled Train data set
- After fitting the model, we will do the prediction on the train and test dataset.
- Now we check for performance metrics for Class 1 Label.

Train Dataset					**Test Dataset**				
Accuracy Score	**Precision**	**Recall**	**F1 Score**	**AUC Score**	**Accuracy Score**	**Precision**	**Recall**	**F1 Score**	**AUC Score**
96.51	95.00	98.00	97.00	99.56	89.57	64.00	88.00	74.00	95.17

Table 23: Train & Test Data Performance Metrics of Class 1 for KNN with SMOTE & Default Parameters

- Interpretation from the KNN Model with balanced dataset
  - ✓ From the Train set we can see that precision is of 0.95 & recall is of 0.98 for Class 1
  - ✓ From the Test set we can see that precision is of 0.64 & recall is of 0.88 for Class 1
  - ✓ We can see slight improvement in recall values for both Train & Test dataset. But still further tuning might be required.

(ii) Random Forest Classifier (RFCL):

- Will be creating a Random Forest model with default parameters
- Fitting the model into Resampled Train data set
- After fitting the model, we will do the prediction on the train and test dataset.
- Now we check for performance metrics for Class 1 Label

Train Dataset					Test Dataset				
Accuracy Score	Precision	Recall	F1 Score	AUC Score	Accuracy Score	Precision	Recall	F1 Score	AUC Score
100.00	100.00	100.00	100.00	100.00	95.57	86.00	89.00	87.00	98.70

Table 24: Train & Test Data Performance Metrics of Class 1 for RFCL with SMOTE & Default Parameters

- Interpretation from the Random Forest Model with balanced dataset
  - ✓ From the Train set we can see that precision is of 1.00 & recall is of 1.00 for Class 1
  - ✓ From the Test set we can see that precision is of 0.86 & recall is of 0.89 for Class 1
  - ✓ We can see slight improvement in recall values for both Train & Test dataset. But still further tuning might be required.

(iii) XG Boost Classifier:

- Will be creating a XG Boost model with default parameters
- Fitting the model into Resampled Train data set
- After fitting the model, we will do the prediction on the train and test dataset.
- Now we check for performance metrics for Class 1 Label

Train Dataset					Test Dataset				
Accuracy Score	Precision	Recall	F1 Score	AUC Score	Accuracy Score	Precision	Recall	F1 Score	AUC Score
90.36	91.00	89.00	90.00	96.53	88.36	63.00	74.00	68.00	91.43

Table 25: Train & Test Data Performance Metrics of Class 1 for XG Boost with SMOTE & Default Parameters

- Interpretation from the XG Boost Model with balanced dataset
  - ✓ From the Train set we can see that precision is of 0.91 & recall is of 0.89 for Class 1
  - ✓ From the Test set we can see that precision is of 0.63 & recall is of 0.74 for Class 1
  - ✓ We can see slight improvement in recall values for both Train & Test dataset. But still further tuning might be required.

(iv) Artificial Neural Networks:

- For Neural Network we will be using scaled dataset.
- Will be creating a Neural Network model with default parameters
- Fitting the model into Resampled Train data set
- After fitting the model, we will do the prediction on the train and test dataset.
- Now we check for performance metrics for Class 1 Label.

Train Dataset					Test Dataset				
Accuracy Score	Precision	Recall	F1 Score	AUC Score	Accuracy Score	Precision	Recall	F1 Score	AUC Score
88.14	90.00	92.00	91.00	96.58	87.90	62.00	75.00	68.00	91.16

Table 26: Train & Test Data Performance Metrics of Class 1 for ANN with SMOTE & Default Parameters

- Interpretation from the Neural Network Model with balanced dataset
  - ✓ From the Train set we can see that precision is of 0.90 & recall is of 0.92 for Class 1
  - ✓ From the Test set we can see that precision is of 0.62 & recall is of 0.75 for Class 1
  - ✓ We can see slight improvement in recall values for both Train & Test dataset. But still further tuning might be required.

(v) Logistic Regression:

- For Logistic Regression we will be using scaled dataset.
- Will be creating a Logistic Regression model with default parameters
- Fitting the model into Resampled Train data set
- After fitting the model, we will do the prediction on the train and test dataset.
- Now we check for performance metrics for Class 1 Label.

Train Dataset					Test Dataset				
Accuracy Score	Precision	Recall	F1 Score	AUC Score	Accuracy Score	Precision	Recall	F1 Score	AUC Score
78.55	76.00	83.00	79.00	86.16	76.08	39.00	75.00	51.00	82.36

Table 27: Train & Test Data Performance Metrics of Class 1 for Logistic Regression with SMOTE & Default Parameters

- Interpretation from the Logistic Regression Model with balanced dataset
  - ✓ From the Train set we can see that precision is of 0.76 & recall is of 0.83 for Class 1
  - ✓ From the Test set we can see that precision is of 0.39 & recall is of 0.75 for Class 1
  - ✓ We can see slight improvement in recall values for both Train & Test dataset. But still further tuning might be required.

(vi) Support Vector Classifier:

- Will be creating a Support Vector Classifier model with default parameters.
- Fitting the model into Resampled Train data set.
- After fitting the model, we will do the prediction on the train and test dataset.
- Now we check for performance metrics for Class 1 Label.

Train Dataset					Test Dataset				
Accuracy Score	Precision	Recall	F1 Score	AUC Score	Accuracy Score	Precision	Recall	F1 Score	AUC Score
99.67	99.00	100.00	100.00	99.95	88.51	74.00	49.00	59.00	91.46

Table 28: Train & Test Data Performance Metrics of Class 1 for SVC with SMOTE & Default Parameters

- Interpretation from the Support Vector Classifier Model with balanced dataset
  - ✓ From the Train set we can see that precision is of 0.99 & recall is of 1.00 for Class 1
  - ✓ From the Test set we can see that precision is of 0.74 & recall is of 0.49 for Class 1
  - ✓ We can see slight improvement in recall values for both Train & Test dataset. But still further tuning might be required.

e) Building Models with SMOTE & Tuned Parameters

(i) K-Nearest Neighbors (KNN):

- Entering the parameters into the grid so as to find optimal parameters using random search.
- Dataset used here will be of scaled since it's a distance-based model.
- Next, we will be creating KNN model with default parameters.
- Building Random Search with a cross validation of 5 along with above defined parameters grid & model.
- After creating the model, we will fit the model to resampled train set.
- Now we will be checking the best parameters using best params function
  - ✓ **Best Parameters** : {'weights': 'distance', 'p': 1, 'n_neighbors': 10, 'metric': 'minkowski', 'leaf_size': 30, 'algorithm': 'auto'}
- Creating the model with best parameters & next we do prediction on Train & Test dataset.

- Now we check for performance metrics for Class 1 Label.

Train Dataset					Test Dataset				
Accuracy Score	Precision	Recall	F1 Score	AUC Score	Accuracy Score	Precision	Recall	F1 Score	AUC Score
100.00	100.00	100.00	100.00	100.00	93.21	73.00	96.00	83.00	98.28

Table 29: Train & Test Data Performance Metrics of Class 1 for KNN with SMOTE & Tuned Parameters

- Interpretation from the KNN Model with balanced & Tuned dataset
  - ✓ From the Train set we can see that precision is of 1.00 & recall is of 1.00 for Class 1
  - ✓ From the Test set we can see that precision is of 0.73 & recall is of 0.96 for Class 1
  - ✓ We can see slight improvement in recall values for both Train & Test dataset.

(ii) Random Forest Classifier:

- Entering the parameters into the grid so as to find optimal parameters using random search.
- Next, we will be creating Random Forest model with default parameters.
- Building Random Search with a cross validation of 5 along with above defined parameters grid & model.
- After creating the model, we will fit the model to resampled train set.
- Now we will be checking the best parameters using best params function
  - ✓ **Best Parameters** : {'n_estimators': 100, 'min_samples_split': 60, 'max_features': 3, 'max_depth': 15}
- Creating the model with best parameters & next we do prediction on Train & Test dataset.
- Now we check for performance metrics for Class 1 Label.

Train Dataset					Test Dataset				
Accuracy Score	Precision	Recall	F1 Score	AUC Score	Accuracy Score	Precision	Recall	F1 Score	AUC Score
92.59	92.00	93.00	93.00	98.15	89.87	66.00	81.00	73.00	94.35

Table 30: Train & Test Data Performance Metrics of Class 1 for RFCL with SMOTE & Tuned Parameters

- Interpretation from the RFCL Model with balanced & Tuned dataset
  - ✓ From the Train set we can see that precision is of 0.92 & recall is of 0.93 for Class 1
  - ✓ From the Test set we can see that precision is of 0.66 & recall is of 0.81 for Class 1
  - ✓ We can see slight improvement in recall values for both Train & Test dataset.

(iii) XG Boost Classifier:

- Entering the parameters into the grid so as to find optimal parameters using random search.
- Next, we will be creating XG Boost model with default parameters.
- Building Random Search with a cross validation of 5 along with above defined parameters grid & model.
- After creating the model, we will fit the model to resampled train set.
- Now we will be checking the best parameters using best params function
  - ✓ **Best Parameters** : {'subsample': 0.8, 'n_estimators': 300, 'max_depth': 10, 'learning_rate': 0.01, 'gamma': 0, 'colsample_bytree': 0.5}
- Creating the model with best parameters & next we do prediction on Train & Test dataset.
- Now we check for performance metrics for Class 1 Label.

Train Dataset					Test Dataset				
Accuracy Score	Precision	Recall	F1 Score	AUC Score	Accuracy Score	Precision	Recall	F1 Score	AUC Score
97.75	97.00	99.00	98.00	97.47	93.45	77.00	87.00	82.00	99.80

Table 31: Train & Test Data Performance Metrics of Class 1 for XG Boost with SMOTE & Tuned Parameters

- Interpretation from the XG Boost Model with balanced & Tuned dataset
  - ✓ From the Train set we can see that precision is of 0.97 & recall is of 0.99 for Class 1
  - ✓ From the Test set we can see that precision is of 0.77 & recall is of 0.87 for Class 1
  - ✓ We can see slight improvement in recall values for both Train & Test dataset.

(iv) Artificial Neural Networks:

- Entering the parameters into the grid so as to find optimal parameters using random search.
- Dataset used here will be of scaled.
- Next, we will be creating Neural Network model with default parameters.
- Building Random Search with a cross validation of 5 along with above defined parameters grid & model.
- After creating the model, we will fit the model to resampled train set.
- Now we will be checking the best parameters using best params function
    - **Best Parameters** : {'tol': 0.001, 'solver': 'adam', 'max_iter': 300, 'hidden_layer_sizes': 300, 'activation': 'relu'}
- Creating the model with best parameters & next we do prediction on Train & Test dataset.
- Now we check for performance metrics for Class 1 Label.

Train Dataset					**Test Dataset**				
Accuracy Score	**Precision**	**Recall**	**F1 Score**	**AUC Score**	**Accuracy Score**	**Precision**	**Recall**	**F1 Score**	**AUC Score**
95.49	94.00	97.00	96.00	99.08	90.84	91.00	69.00	79.00	95.19

Table 32: Train & Test Data Performance Metrics of Class 1 for ANN with SMOTE & Tuned Parameters

- Interpretation from the Neural Networks Model with balanced & Tuned dataset
    - From the Train set we can see that precision is of 0.94 & recall is of 0.97 for Class 1
    - From the Test set we can see that precision is of 0.91 & recall is of 0.69 for Class 1
    - We can see slight improvement in recall values for both Train & Test dataset.

(v) Logistic Regression:

- Entering the parameters into the grid so as to find optimal parameters using random search.
- Dataset used here will be of scaled.
- Next, we will be creating Neural Network model with default parameters.
- Building Random Search with a cross validation of 5 along with above defined parameters grid & model.
- After creating the model, we will fit the model to resampled train set.
- Now we will be checking the best parameters using best params function
    - **Best Parameters** : {'tol': 0.001, 'solver': 'saga', 'penalty': 'none', 'max_iter': 100, 'class_weight': 'none'}
- Creating the model with best parameters & next we do prediction on Train & Test dataset.
- Now we check for performance metrics for Class 1 Label.

Train Dataset					**Test Dataset**				
Accuracy Score	**Precision**	**Recall**	**F1 Score**	**AUC Score**	**Accuracy Score**	**Precision**	**Recall**	**F1 Score**	**AUC Score**
78.75	77.00	83.00	80.00	86.22	76.17	39.00	75.00	51.00	82.42

Table 33: Train & Test Data Performance Metrics of Class 1 for Logistic Regression with SMOTE & Tuned Parameters

- Interpretation from the Logistic Regression Model with balanced & Tuned dataset
    - From the Train set we can see that precision is of 0.77 & recall is of 0.83 for Class 1
    - From the Test set we can see that precision is of 0.39 & recall is of 0.75 for Class 1
    - We can see slight improvement in recall values for both Train & Test dataset.

6. Model Validation:

In total we selected 6 models with 22 different combinations i.e., default parameters, hyper tuned parameters, SMOTE with default parameters & SMOTE with hyper tuned parameters. Below are the comparison table of performance metrics of all 22 combinations.

S.No	Model Combinations	Model Name	Train Dataset					Test Dataset					Remarks
			Accuracy Score	Precision	Recall	F1 Score	AUC Score	Accuracy Score	Precision	Recall	F1 Score	AUC Score	
1	With Default Parameters	K- Nearest Neighbours	97.21	95	88	91	99.4	93.96	87	75	81	96.07	
2		Random Forest Classifier	100	100	100	100	100	96.33	95	82	88	99.15	
3		XG Boost	91.78	85	63	72	95.18	90.69	83	56	67	93.28	
4		Neural Network	92.77	84	71	77	96.12	90.69	79	62	69	93.17	
5		Logistic Regression	88.34	79	42	55	87.19	87.29	74	38	50	84.26	
6		Support Vector Classifier	98.89	99	94	97	99.95	87.2	84	30	44	91.39	
7	With Hypertuning	K- Nearest Neighbours	100	100	100	100	100	95.99	95	81	87	98.88	
8		Random Forest Classifier	92.165	90	60	72	96.65	90.6	85	54	66	94.68	
9		XG Boost	98.09	100	89	94	99.73	94.63	94	73	82	98.19	
10		Neural Network	95.06	92	78	84	98.22	92.02	85	64	73	95.23	
11		Logistic Regression	88.61	78	46	57	87.37	87.63	74	41	53	84.53	
12	With SMOTE & Default Parameters	K- Nearest Neighbours	96.51	95	98	97	99.56	89.57	64	88	74	95.17	
13		Random Forest Classifier	100	100	100	100	100	95.57	86	89	87	98.7	
14		XG Boost	87.3	88	87	87	94.73	86.78	58	77	66	90.94	
15		Neural Network	90.36	91	89	90	96.53	88.36	63	74	68	91.43	
16		Logistic Regression	78.55	76	83	79	86.16	76.08	39	75	51	82.36	
17		Support Vector Classifier	99.67	99	100	100	99.95	88.51	74	49	59	91.46	
18	With SMOTE & Hypertuning	K- Nearest Neighbours	100	100	100	100	100	93.21	73	96	83	98.28	
19		Random Forest Classifier	92.54	92	93	93	98.16	90.26	67	82	74	94.58	
20		XG Boost	97.42	97	98	97	99.7	93.08	75	87	81	97.28	
21		Neural Network	96.25	96	97	96	99.23	92.05	64	77	70	94.97	
22		Logistic Regression	78.75	77	83	80	86.22	76.17	39	75	51	82.42	

Table 34: Model Performance Metrics Comparison Table

a) Top Five (5) Better Performing Models:

From the above table identified top five (5) models which are performing better when compared to other models. Below are the identified models which are performing better based on the Test Recall values, Accuracy & AUC Score.

- ✓ KNN with SMOTE & Hyper tuning
- ✓ Random Forest with SMOTE & Default Parameters
- ✓ KNN with SMOTE & Default Parameters
- ✓ XG Boost with SMOTE & Hyper tuning
- ✓ Random Forest with SMOTE & Hyper tuning

S.No	Model Name & Combination	Train Set		Test Set	
		Accuracy Score	Recall	Accuracy Score	Recall
1	KNN with SMOTE & Hypertuning	100	100	93.21	96
2	Random Forest with SMOTE & Default Parameters	100	100	95.57	89
3	KNN with SMOTE & Default Parameters	96.51	98	89.57	88
4	XG Boost with SMOTE & Hypertuning	97.42	93	93.08	87
5	Random Forest with SMOTE & Hypertuning	92.54	93	90.26	82

Table 35: Top Five (5) Better Performing Models

b) ROC Curve for Top Five (5) Better Performing Models

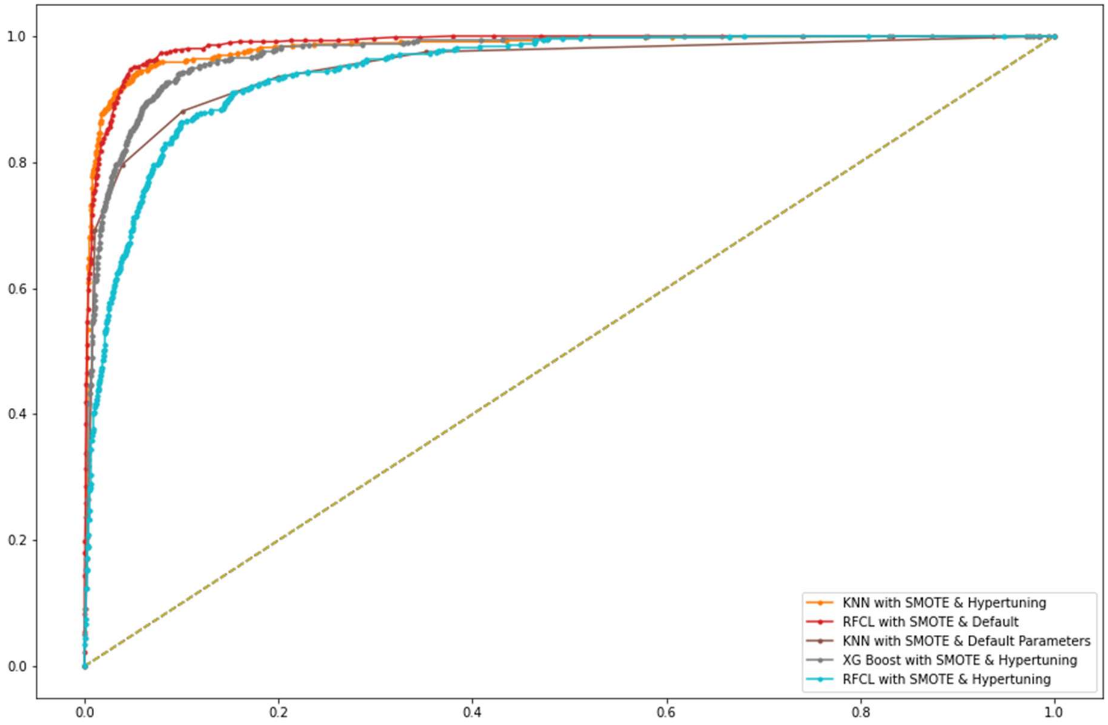


Figure 36: ROC Curve for Top Five (5) Better Performing Models

c) AUC Score of Test Set for Top Five (5) Performing Models

S.No	Model Description	Test AUC Score
1	KNN with SMOTE & Hypertuning	98.28
2	Random Forest with SMOTE & Default Parameters	98.7
3	KNN with SMOTE & Default Parameters	95.17
4	XG Boost with SMOTE & Hypertuning	97.28
5	Random Forest with SMOTE & Hypertuning	94.58

Table 36: AUC Score of Test Set for Top Five (5) Selected Models

d) Top Features or Feature Importance based on the model.

From the modelling exercise these are the top 5 features/variables which are important than other variables,

- Tenure
- Account Segment
- Days Since Last Connect with Customer care
- Cashback
- Revenue Growth (YOY)

7. Business Insights & Recommendations:

a) Insights on Basis of Exploratory Data Analysis:

- Majority of customers whose account tenure is less than 10 months are more likely to churn. Most probably they might be new customers who tend to move or try another service.
- Most of the customers who are from Tier – 1 cities are more likely to churn when compared to other tier cities. This might be due to various options available in Tier – 1 cities.
- Around 73% of account preferred payment mode is Debit & Credit Card. Out of the 73% of accounts around 15% of customers are more likely to churn.
- Primary users of the account who are male are more likely to churn than Female, i.e., around 17% of them are likely to churn.
- Around 72% of the accounts in the given data set are of Regular plus & super type of accounts. Out of which around 19% of the primary users of the account tend to churn. Regular plus and super might be mid-level accounts and there must be no significant differences between them hence they might churn.
- Most the accounts whose primary user is single churned more when compared to married and divorced.
- Most of the primary users of the account who are having more than three users per account are more likely to churn when compared to users who are having less two users per account.
- Around 20% of customers who have given a satisfactory score of 3 and above for the customer care provided by the company tend to churn which indicates that customers are not satisfied with the customer care service.
- Around 31% of customers who have complained during the last one year are more likely to churn, similarly around 10% of customers who didn't raise any complain during the last one year are also more likely to churn. This is a bit serious issue, as customer not raising any complain but churning.

b) Business Recommendations:

- **More than 70% of transactions** are through Debit & Credit Card. Hence company can tie up with credit/debit card agency for coupons, rewards etc. to attract customers & similarly retain existing customer.
- **Around 73% of accounts** are of ‘Regular Plus’ & ‘Super’ segment. We can see around 20% churn in these segments combined. Company can further look into these segments for any changes in plans or any customization of plans, etc. so that they can reduce the churn rate.
- **There is around 21% churn** in accounts whose tenure is less than 15 months. Hence company can offer them any discounts/offers or suggest them different plans or providing them loyalty programs to retain their account.
- **Around 12% of customers** who didn’t complaint, yet they have churned out which need to be addressed in detail by the company.
- **More than 20% of customers** who have given satisfactory score of 3 & above for customer care provided tend to churn, hence company should focus more on customer complaints, feedback and addressing them with minimum time.

---END OF THE PROJECT---