Capstone Project Customer Churn – Notes 1

TEJAS PADEKAR
PGP-DSBA Online
Date: 25/12/2022

CONTENTS:

List of Figures

1. Introduction of the business problem

a) Defining problem statement

Problem:

An E Commerce company provider is facing a lot of competition in the current market and it has become a challenge to retain the existing customers in the current situation. Hence, the company wants to develop a model through which they can do churn prediction of the accounts and provide segmented offers to the potential churners. In this company, account churn is a major thing because 1 account can have multiple customers. hence by losing one account the company might be losing more than one customer.

Problem Statement:

We have been assigned to develop a churn prediction model for this company and provide business recommendations on the campaign. Our campaign suggestion should be unique and be very clear on the campaign offer because our recommendation will go through the revenue assurance team. If they find that we are giving a lot of free (or subsidized) stuff thereby making a loss to the company; they are not going to approve our recommendation. Hence we need to be very careful while providing campaign recommendation.

b) Need of the study/project

Customer Churn is one of the biggest global problems faced by businesses across various industries which also includes the e-commerce service industry. With increased competition and a plethora of old, fresh and upcoming availability of business options to choose from, this problem is ever growing and continue existing throughout the business life cycle for a company. Moreover, affordable and ease of availability of data services and internet connectivity along with more awareness of information has also made customers smarter and more conscious. Thus, impacting their decision making when choosing their loyalty towards a particular brand or business.

Rapidly changing business environment and technological advancements create many opportunities for e-commerce businesses to draft and roll out plans to improve service quality, provide quick delivery solution, resolve queries quickly, track and collect customer information, track and collect competition information, create ideas and innovations which can help them in customer retention.

Customer base for e-commerce industry also varies in terms of attributes which calls for an important role for businesses to create varied products and services according to their customer segments in order to retain them. In our case, one account has multiple users. Hence, it becomes even more important to focus on retention as churning would impact revenue and profits tremendously. Also, generally customer retention is less expensive for any business than customer acquisition. Hence, it is both necessary as well as beneficial/profitable.

c) Understanding business/social opportunity

i. Business Opportunity

- Customer retention is indirectly profit retention.
- It provides stability for any business.
- Keeps investors and shareholders happy and helps business gain their trust. Gaining trust enables them to innovate and explore growth and expansion plans.
- It is necessary for maintaining continued demand and even market gain.
- Also helps in brand building and brand awareness.
- Facilitates in gaining more vendors and customers (word of mouth publicity). Increased vendor demand also helps gain pricing power for companies and they can buy at lower prices which in turn they can pass on to their customers as discounts to gain loyalty and stay competitive.

ii. Social Opportunity

- Contributes to job creation and economic development of a country.
- Improves demand for core vendor businesses as well as supportive businesses such as cargo and shipping, packaging, freight and transport, IT, services and communication etc.
- E-commerce facilitates purchases which contributes to the improvement in standard of living for its customers.
- Creates platform for diverse businesses to offer products and services under a single roof which benefits customers from ease of purchasing without hassle and save time.
- Creates awareness of various brands and suppliers to their customers.
- Creates internal competitiveness for vendors, which creates demand for improved quality of services which benefits customers with better quality products and services.
- E-commerce business also keeps prices in check and also competitive. It also contributes to refrain from being trapped by fraudsters and purchasing non-original products as it offers a one-stop solution for its customers.

2. Data Report

a) Understanding how data was collected in terms of time, frequency and methodology

- Data collected for majority variables is of last 12 months for a total of 11260 unique accounts.
- Data has been provided by the E-commerce company itself.
- Data is a mix of information related to **demographics of customers** (tenure, gender, city tier, marital status, Account_user_count), **customer spending behaviour/patterns** (payment, account segment, rev_per_month, rev_growth_yoy, coupon_used_for_payment) , **customer business relationship** (CC_Contacted_L12m, CC_Agent_Score, Complain_ly, Day_Since_CC_connect) & **customer preference** (cashback, login). Please see below image for snippet of the variables and observations in them.

b) Visual inspection of data (rows, columns, descriptive details)

	AccountID	Churn	Tenure	City_Tier	CC_Contacted_LY	Payment	Gender	Service_Score	Account_user_count	account_segment	CC_Agent_Score
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

Figure 1: Dataset Head (1 to 11 variables and top 5 rows)

Marital_Status	rev_per_month	Complain_ly	rev_growth_yoy	coupon_used_for_payment	Day_Since_CC_connect	cashback	Login_device
Single	9	1.0	11	1	5	159.93	Mobile
Single	7	1.0	15	0	0	120.9	Mobile
Single	6	1.0	14	0	3	NaN	Mobile
Single	8	0.0	23	0	3	134.07	Mobile
Single	3	0.0	11	1	3	129.6	Mobile

Figure 2: Dataset Head (12 to 19 variables and top 5 rows)

	count	unique	top	freq	mean	std	min	25%	50%	75%	max
AccountID	11260.00	NaN	NaN	NaN	25629.50	3250.63	20000.00	22814.75	25629.50	28444.25	31259.00
Churn	11260.00	NaN	NaN	NaN	0.17	0.37	0.00	0.00	0.00	0.00	1.00
Tenure	11158.00	38.00	1.00	1351.00	NaN	NaN	NaN	NaN	NaN	NaN	NaN
City_Tier	11148.00	NaN	NaN	NaN	1.65	0.92	1.00	1.00	1.00	3.00	3.00
CC_Contacted_LY	11158.00	NaN	NaN	NaN	17.87	8.85	4.00	11.00	16.00	23.00	132.00
Payment	11151	5	Debit Card	4587	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Gender	11152	4	Male	6328	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Service_Score	11162.00	NaN	NaN	NaN	2.90	0.73	0.00	2.00	3.00	3.00	5.00
Account_user_count	11148.00	7.00	4.00	4569.00	NaN	NaN	NaN	NaN	NaN	NaN	NaN
account_segment	11163	7	Super	4062	NaN	NaN	NaN	NaN	NaN	NaN	NaN
CC_Agent_Score	11144.00	NaN	NaN	NaN	3.07	1.38	1.00	2.00	3.00	4.00	5.00
Marital_Status	11048	3	Married	5860	NaN	NaN	NaN	NaN	NaN	NaN	NaN
rev_per_month	11158.00	59.00	3.00	1746.00	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Complain_ly	10903.00	NaN	NaN	NaN	0.29	0.45	0.00	0.00	0.00	1.00	1.00
rev_growth_yoy	11260.00	20.00	14.00	1524.00	NaN	NaN	NaN	NaN	NaN	NaN	NaN

coupon_used_for_payment	11260.00	20.00	1.00	4373.00	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Day_Since_CC_connect	10903.00	24.00	3.00	1816.00	NaN	NaN	NaN	NaN	NaN	NaN	NaN
cashback	10789.00	5693.00	155.62	10.00	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Login_device	11039	3	Mobile	7482	NaN	NaN	NaN	NaN	NaN	NaN	NaN

Figure 3: Data Description

- Dataset contains total of 18 predictor variables and 1 binary target variable. The predictor variables are a mix of categorical and continuous variables where some of continuous variables such as City_Tier, Service_Score, CC_Agent_Score and Complain_ly are already provided in encoded format
- Observations across variables are in different scales such as text, decimal values, numbers and percentages.
- There are inconsistencies in the data (M, Male, Premium Plus, Premium +) and presence of bad data with incorrect characters assigned such as $, +, 99, #, etc. Hence, will need data preprocessing and cleaning. It is recommended that the company maintain a more standardized, uniform and accurate data as much as possible.
- Outliers seem to be present in 'CC_Contacted_LY' variable and could also be present in the other variables as well which we will explore once we have imputed the null values and provided the appropriate 'int/float' data type to the incorrectly assigned 'object' data type variables.

```
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)
```

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

Figure 4: Data Information with Missing Value Count for each variable

- AccountID variable is not of much significance for prediction. Hence, we will be dropping it.
- City_Tier, Service_Score, CC_Agent_Score and Complain_ly are all actually Categorical variables which have been encoded already and hence, they are of 'float' data type and it will not make sense to analyse the measures of central tendencies such as mean, median or mode for the same.
- Actual categorical variables are 10: Churn, City_Tier, Payment, Gender, Service_Score, account_segment, CC_Agent_Score, Marital_Status, Complain_ly and Login_device.
- Actual Numeric variables are 9: AccountID, Tenure, CC_Contacted_LY, Account_user_count, rev_per_month, rev_growth_yoy, coupon_used_for_payment, Day_Since_CC_connect and cashback.
- There are no duplicate values. However, there are missing values and null values.

- Apart from the variables of 'AccountID', 'Churn' (Target Variable), 'rev_growth_yoy' and 'coupon_used_for_payment' all other variables have null values where ‘cashback’ has the highest missing values (471).
- Once, we have imputed the missing values and Nan values, transformed the data set with correct variable data type, we would analyse the variables again.

c) Understanding of attributes (variable info, renaming if required)

- **AccountID –** Unique Account ID of primary holder. This is not an important variable for analysis and hence, we will be dropping the same.
- **Churn –** Binary Target Variable where 0 within the row represents account id which has not churned and 1 represents account id which has churned.
- **Tenure** – Period since primary holder is a customer of the company in months.
- **City_Tier –** Primary customer's city tier from 1, 2 and 3.
- **CC_Contacted_LY –** No of times all the customers of the account has contacted customer care in last 12 months.
- **Payment –** Preferred Payment mode of the customers in the account.
- **Gender –** Gender of the primary customer.
- **Service_Score –** Satisfaction score given by customers to company.
- **Account_user_count –** Number of customers tagged with an account.
- **account_segment –** Account segmentation on the basis of spend.
- **CC_Agent_Score –** Satisfaction score given by customers on customer care service.
- **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_ly –** If complaint has been raised by account in last 12 months
- **rev_growth_yoy –** revenue growth percentage of the account (last 12 months vs last 13 to 24 to month).
- **coupon_used_for_payment –** 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 from an account has contacted the customer care.
- **Cashback –** Monthly average cashback generated by account in last 12 months.
- **Login_device –** Preferred login device of the customers in the account.

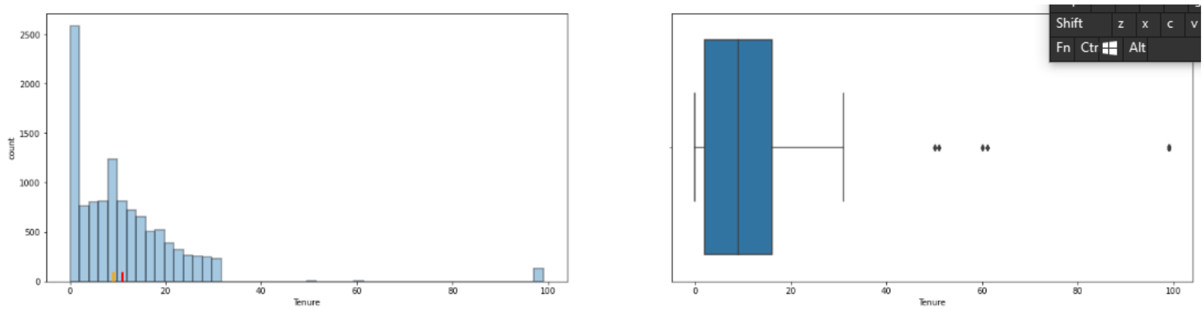
We did not find any need to rename the variables as they were clearly understandable.

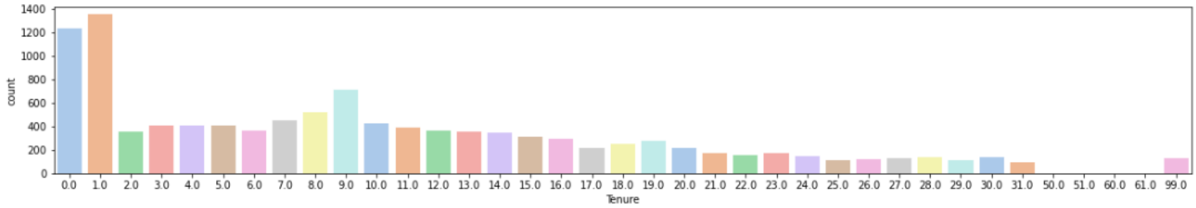
3. Exploratory Data Analysis

a) Univariate analysis (distribution and spread for every continuous attribute, distribution of data in categories for categorical ones)

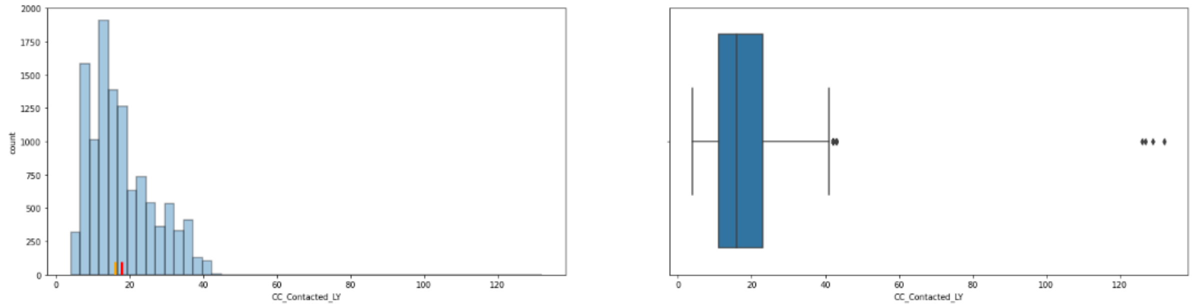
i. Univariate analysis of continuous variables (This analysis is post imputing the missing and null values to make the report more relevant)

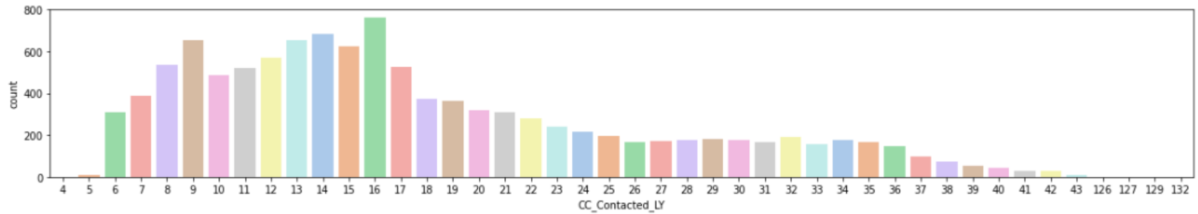
```
Tenure
Skew: 3.94
```



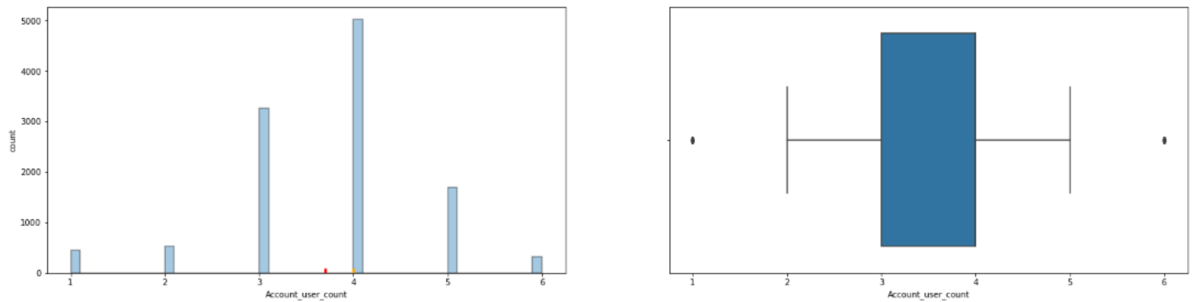


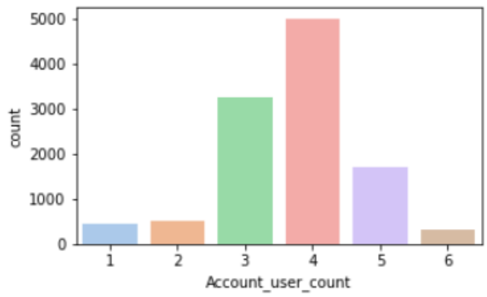
CC_Contacted_LY
Skew: 1.43



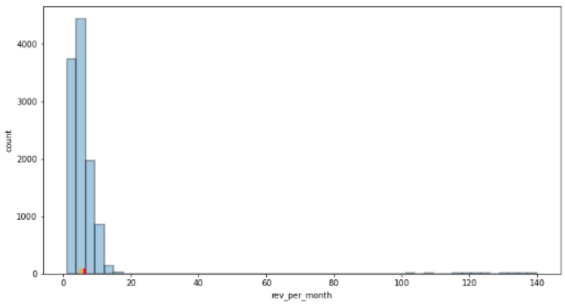


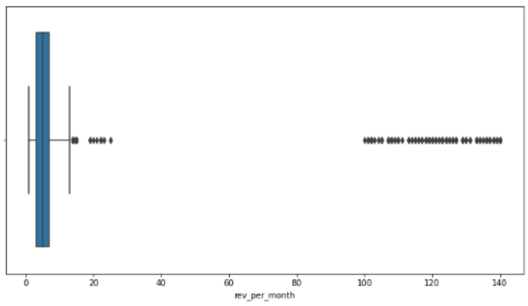
Account_user_count
Skew: -0.43

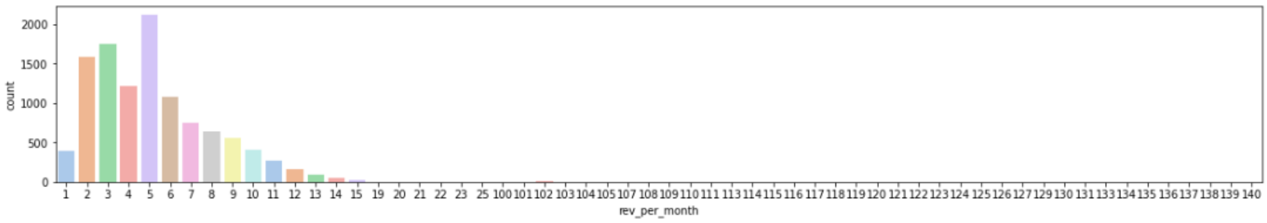




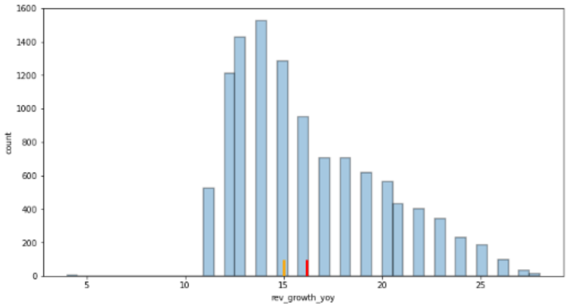

```
rev_per_month
Skew: 9.44
```

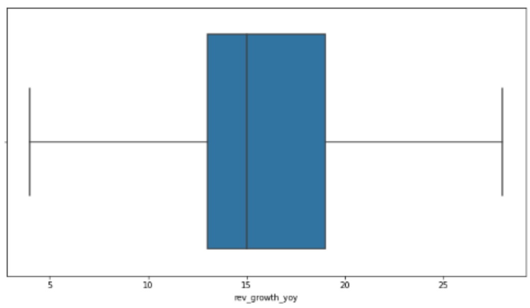


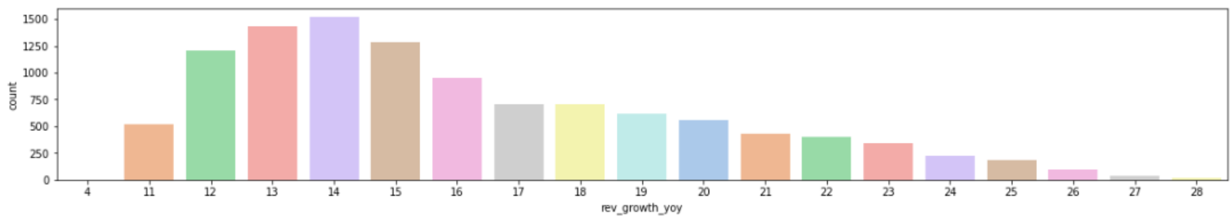




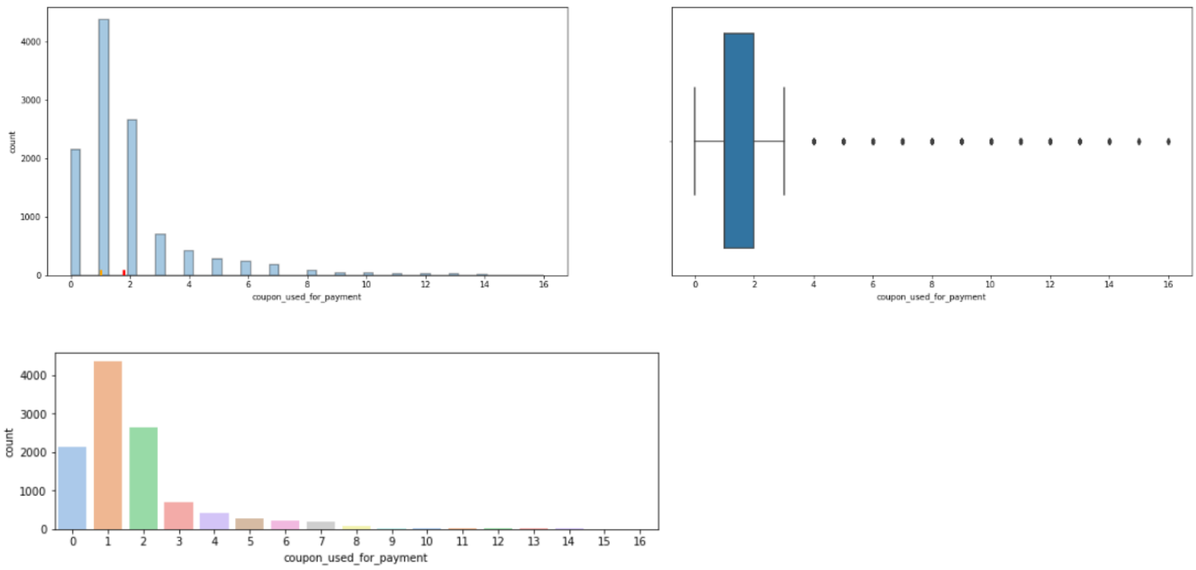

```
rev_growth_yoy
Skew: 0.75
```



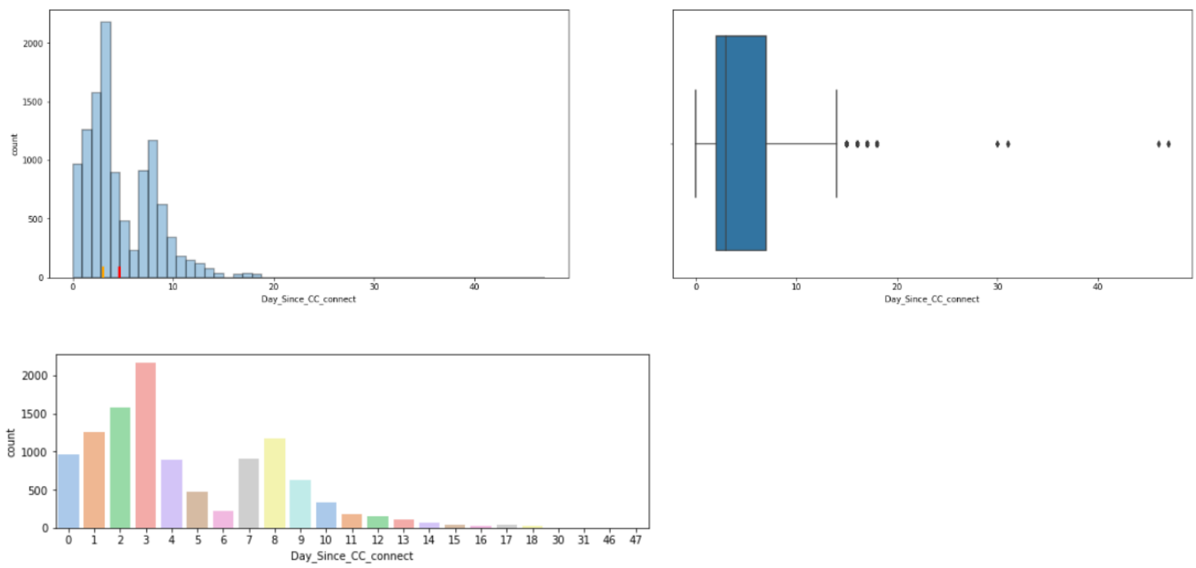





```
coupon_used_for_payment
Skew: 2.58
```



Day_Since_CC_connect
Skew: 1.32



cashback
Skew: 8.97

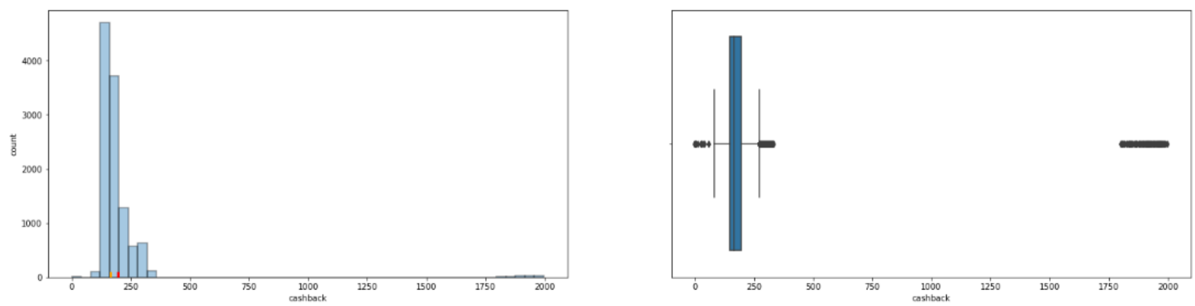
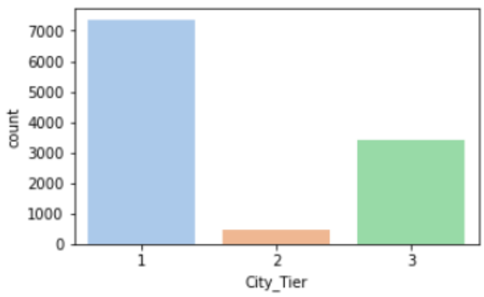


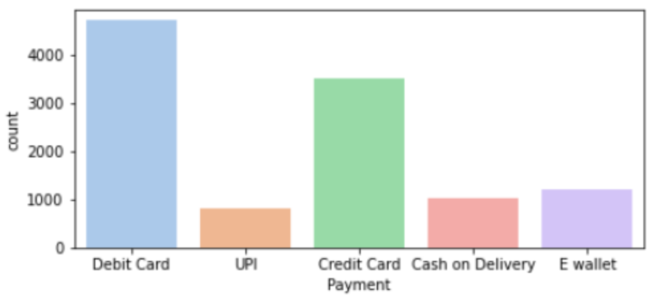
Figure 5: Univariate Analysis of Truly Continuous Variables

- None of the truly continuous variables in above figure are normally distributed and have presence of right skewness in most of them. Left skewness is seen only in Account_user_count variable.

- rev_per_month has an extremely high right skewness of 9.44 and cashback has an extremely high right skewness of 8.97.
- Tenure has a very high right skewness of 3.94 and coupon_used_for_payment has a very high right skewness of 2.58.
- CC_Contacted_LY has a high right skewness of 1.43, , Day_Since_CC_connect has a high right skewness of 1.32.
- rev_growth_yoy has a high right skewness of 0.75, Account_user_count has a medium left skewness of 0.43.
- We can see presence of outliers for the truly numerical variables Tenure, CC_Contacted_LY, Account_user_count, rev_per_month, coupon_used_for_payment, Day_Since_CC_connect and cashback.
- Only rev_growth_yoy variable does not contain outliers. However, outliers presence in Account_user_count also does not seem justifiable as there could be many users registered under one account and this condition should not be applicable as an outlier.
- As per Tenure, maximum accounts have been added in the last couple of months with highest added in the previous month (approx 1351). There is a sign of bad data '99' which could have been assigned to customers for whom the Tenure is not clearly known by the company and hence, it is also shown as an outlier. It is recommended that they try and find this information if possible and reanalyse the data.
- As per Account_user_count, maximum accounts have 4 users and there are approx. 5000 such accounts.
- As per rev_per_month, approx 2000 accounts have generated the highest average monthly revenue of 5000 units in last 12 months (The currency is not clearly provided by the company here)
- As per rev_growth_yoy, approx the highest revenue growth is 14% obtained by approx. 1500 accounts in last 12 months compared to the previous year followed by 13% achieved by approx. 1400 accounts and 12% by approx. 1250 accounts.
- As per coupon_used_for_payment, maximum of approx. 4500 accounts have used 1 coupon to make payment in last 12 months.
- As per cashback, approx. 1997 units is the highest average monthly cashback received by an account. Median 150 accounts approx. have generated maximum no of cashbacks.

ii. Univariate Analysis of Categorical Variables including encoded variables (This analysis is post imputing the missing and null values to make the report more relevant)





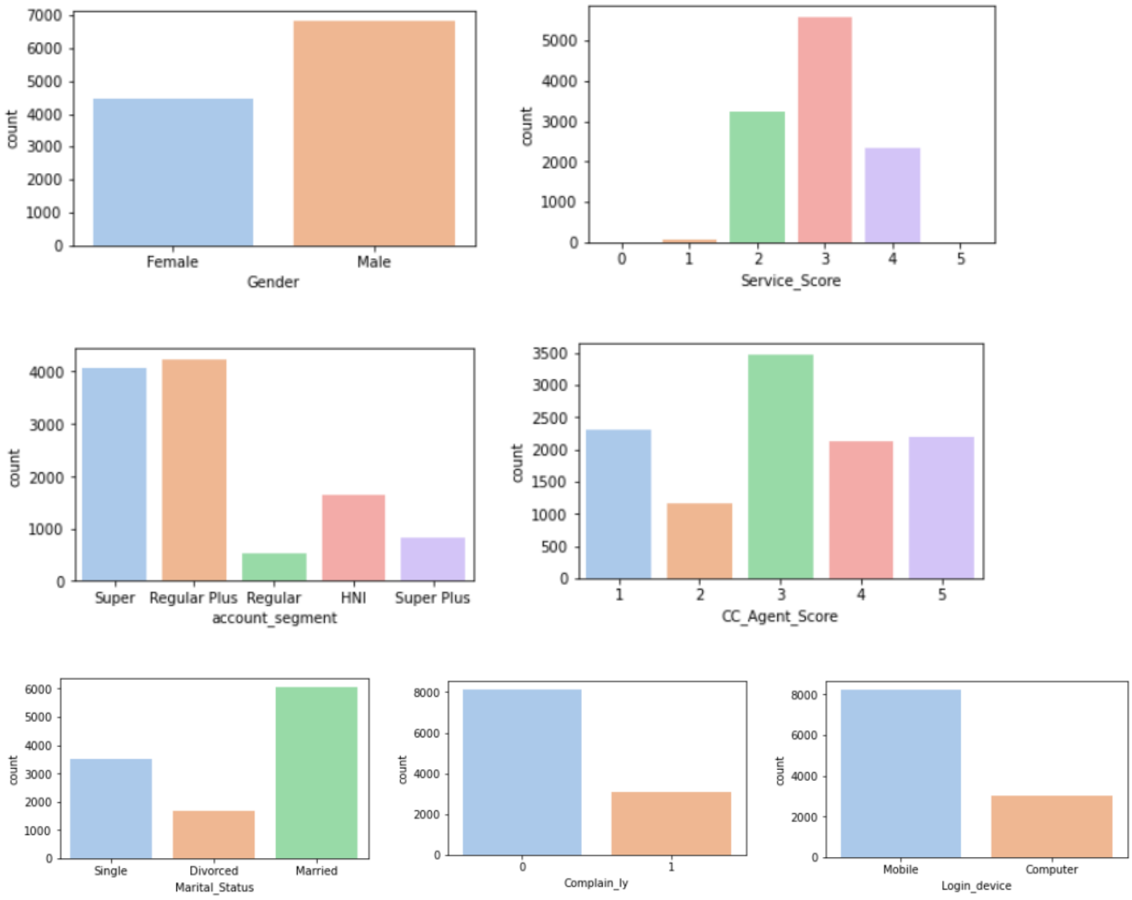


Figure 6: Univariate Analysis of Truly Categorical Variables

- Maximum accounts belong to tier 1 city around approx. 7000 accounts.
- Debit cards are the most preferred payment option for the maximum account's users from the 5 different modes of payments accepted by the business. Approx. 4500 account's users pay using a debit card followed by credit card with approx. 3500 account's users .
- Most of the primary customers are Male (approx 7000).
- Satisfaction Score of 3 has been given by maximum accounts to the company.
- The Regular Plus segment tops the list among 4 other options available (earlier summary in the analysis had identified Super as the top spending segment. However, post data cleaning (correctly replacing M as Male and F as Female) we are now able to determine the actual spending segment). It comprises of approx 4500 of the total accounts. Super is 2nd with approx. 4000 accounts.
- Satisfaction score of 3 has been given by maximum accounts for customer care service.
- Most of the primary customers are Married, around 6000.
- Approx 3000 accounts have raised complaints in last 12 months
- Most logins are via Mobile phones (approx 73%). Approx 8000 accounts

b) Bivariate analysis (relationship between different variables , correlations)

i. Bivariate analysis of continuous variables (This analysis is post imputing the missing and null values to make the report more relevant)

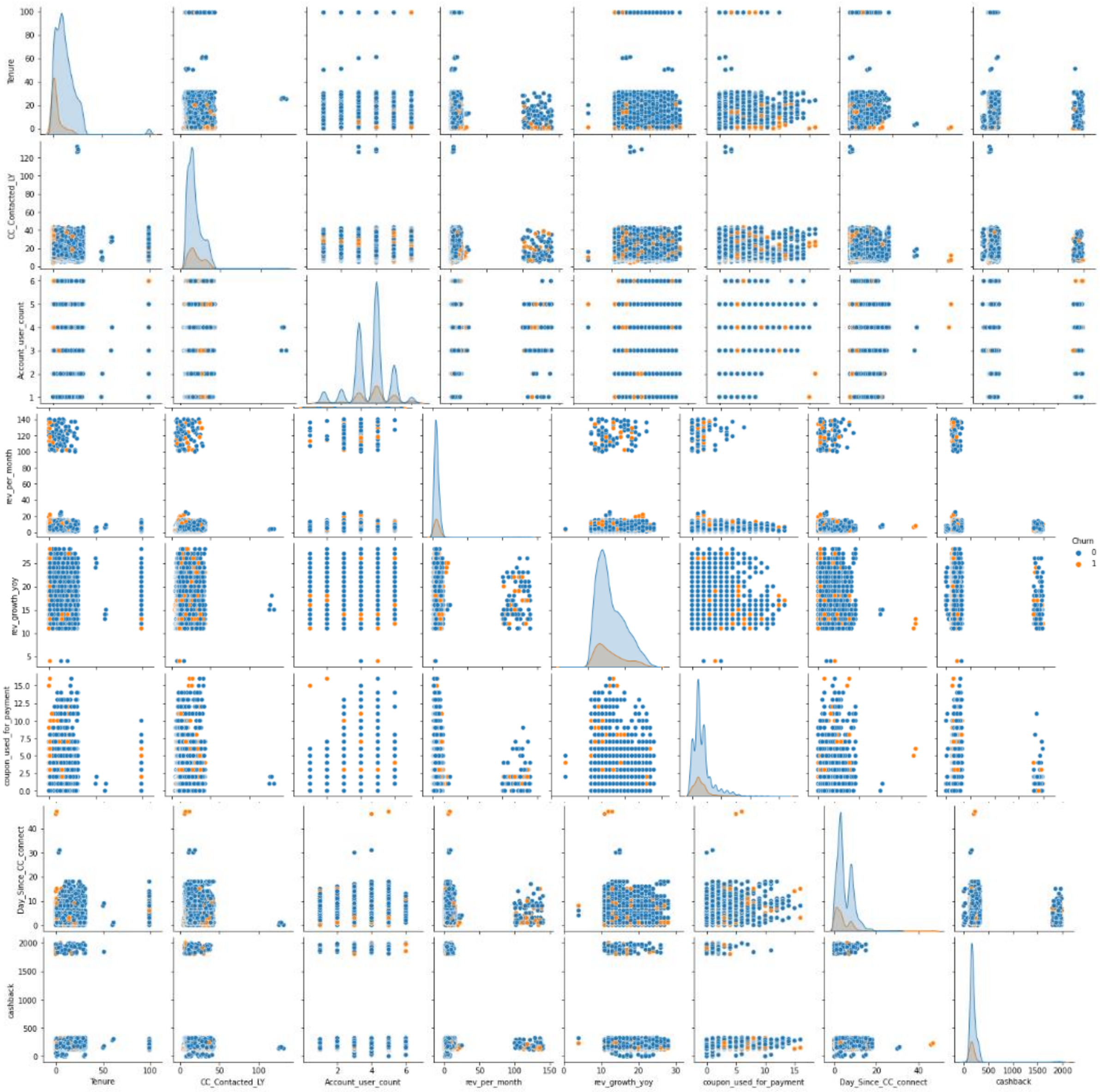
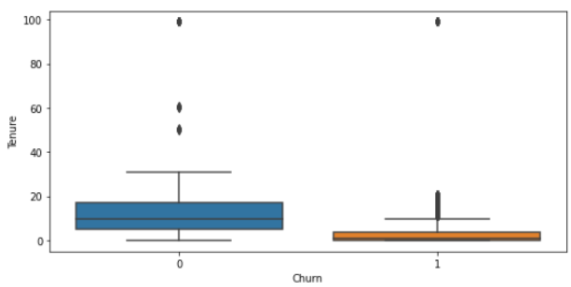
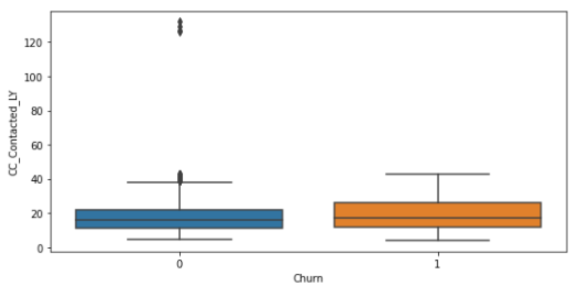


Figure 7: Bivariate Analysis of Truly Continuous Variables

- We do not see any signs of strong correlation among the truly continuous variables with each other.
- We do not see any pattern suggesting any variable being individually strong in predicting the accounts that are likely to churn. This can be seen as the distribution of accounts that have churned are stacked within the range of the accounts not likely to churn.





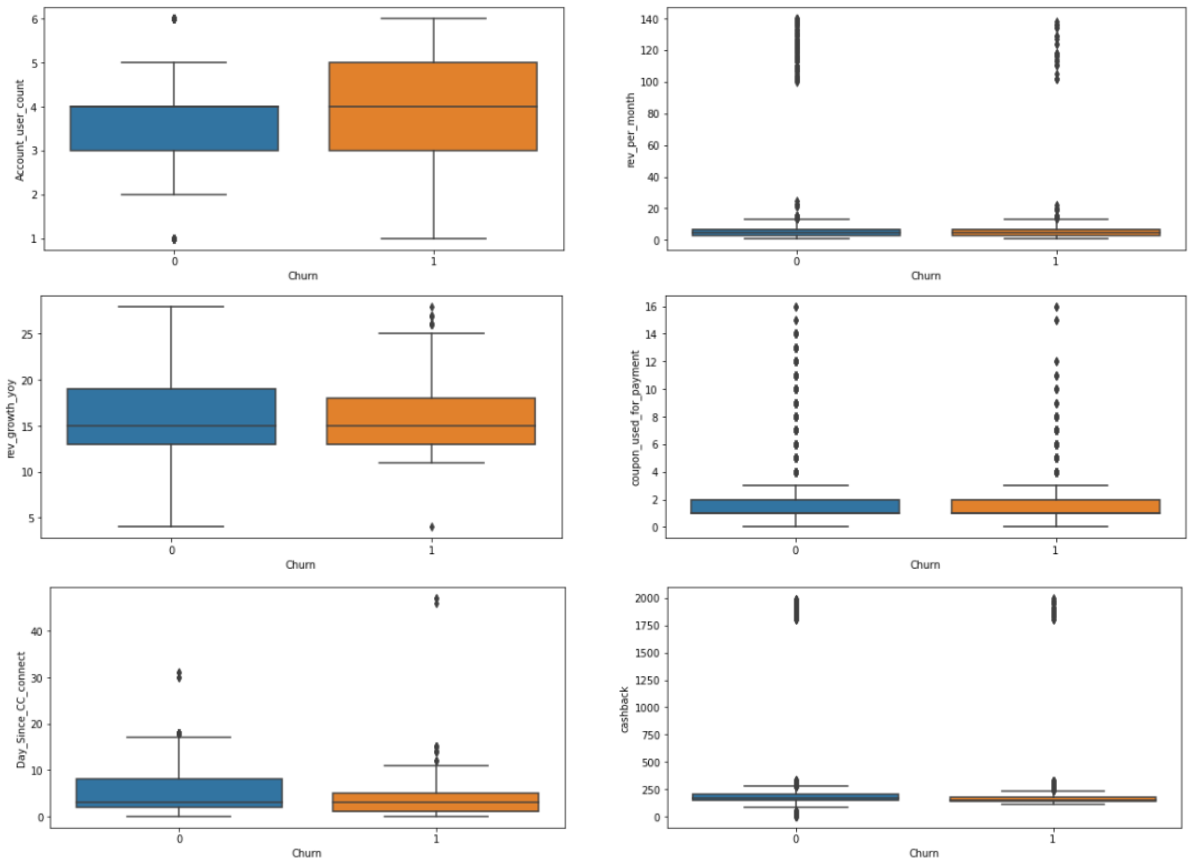
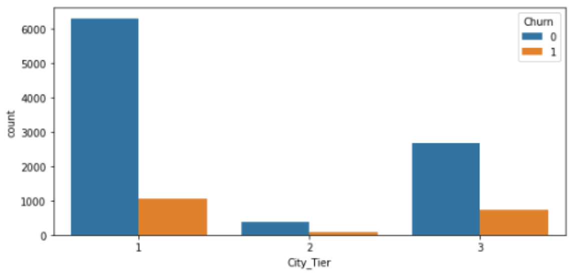
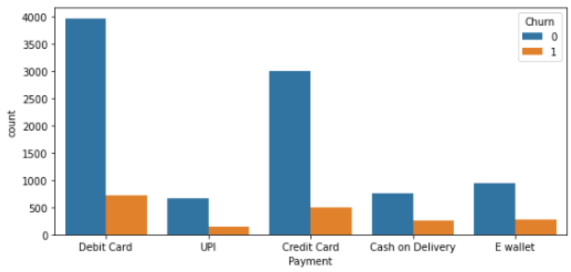


Figure 8: Bivariate Analysis of Truly Continuous Variables II

- Again looking at the above boxplots where we are comparing the continuous variables distribution w.r.t to target variable of Churn, we do not see any major differentiation between 0 and 1 for all variables as none of the classes are completely above or below from each other.
- Only Tenure variables is to able distinguish between class 0 and class 1 of the target variable as the median line of class 0 is outside from the other class. However, class 0 is not the class we are trying to predict.
- Median line of class 0 is slightly outside of class 1 for Account_user_count and perhaps can be a good predictor. Once we build the models, we will be able to evaluate the same.

ii. Bivariate analysis of categorical variables (This analysis is post imputing the missing and null values to make the report more relevant)





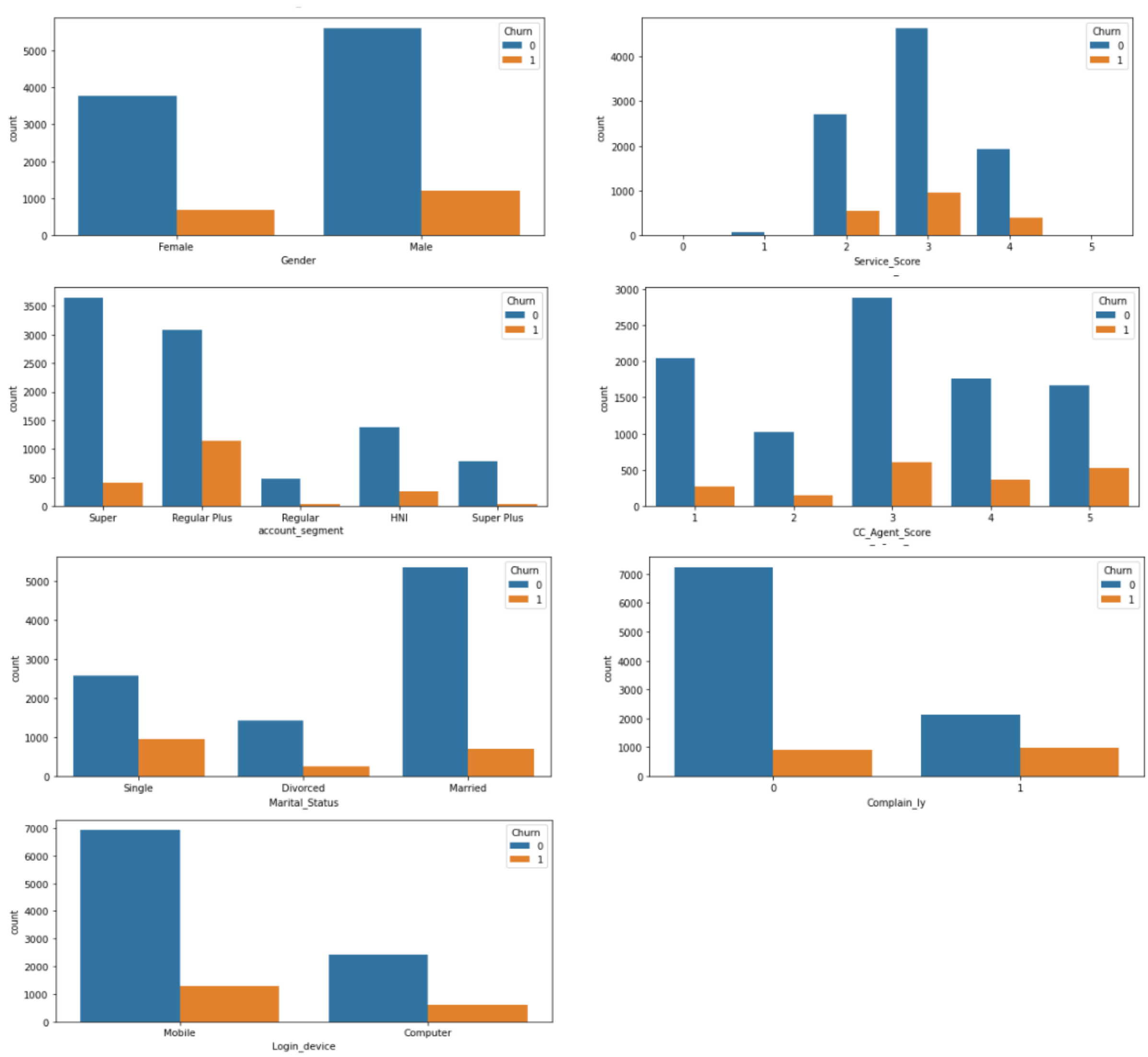


Figure 9: Bivariate Analysis of Truly Categorical Variables

- Even though Tier 1 city has highest no of customers who have churned across all 3 tier cities, the similarity of proportion of class 0 and class 1 is more significant for Tier 3 city. This indicates that business is losing more customers from tier 3 compared to customers in this tier they are able to retain.
- Customers paying cash on delivery and e-wallet seem to churn at a faster rate. UPI transactions are lowest.
- Female account holders need to be added as generally they prefer to shop more than the male. Female seem to be churning at a higher rate than male as per the proportions.
- Need to reduce Satisfaction score for company of 2 and increase that of 4 and 5. Also, need to understand why customers with Satisfaction score of 4 are also churning along with what are the reasons for receiving a poor Satisfaction score of 2 and 1.
- Regular Plus and HNI accounts are churning more.
- Surprisingly, accounts with higher satisfaction score for customer care service have higher churn proportion compared to poor satisfaction score. Accounts churning at similar rate with score of 5 and 3. A good number of accounts have also given a score of 1 which needs to be looked into.
- Account holders who are Single tend to churn more.

- Proportion of customers who have registered a complaint are less compared to who have not registered but they are churning at a faster rate comparatively. 27.6% of accounts have raised complaints in the last one year (approx. 3000) and approx. 1/3rd of these have churned which is quite high.
- Proportion of customers churning using both mobile and laptop seem same.

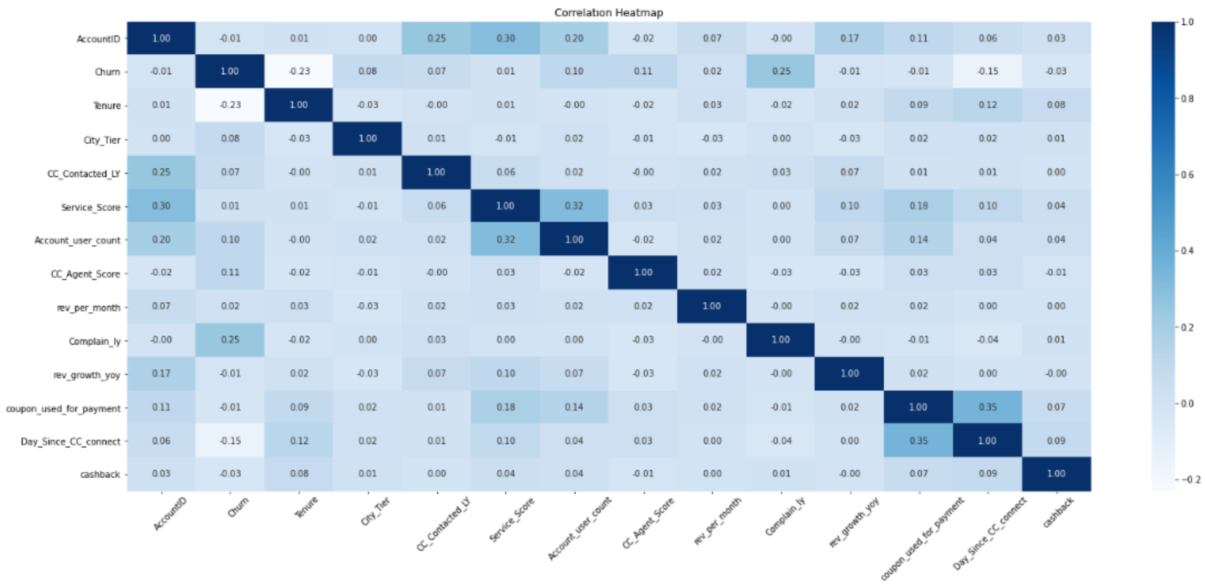


Figure 10: Correlation Heatmap

- Tenure is slightly negatively correlated to Churn (-0.23). Hence, higher the tenure better the chances for retention.
- Complain_ly is slightly positively correlated to Churn (0.25). Hence, complaints encourage probability to churn.
- Day_Since_CC_connect is slightly negatively correlated to Churn (-0.15). Hence, lesser contact with customer care can prevent customer from churning.
- Account_user_count is slightly positively correlated to Satisfaction Score to company (0.32). Better Satisfaction Score to company results in more users being added to primary account.
- coupon_used_for_payment is slightly positively correlated to Day_Since_CC_connect (0.35). Customers using coupons to make payments tend to connect with customer care more.

c) Removal of unwanted variables (if applicable)

- AccountID has to be dropped/removed as it adds no significance for predicting the target variable.
- For the balance remaining variables, Individually the predictor variables look weak except for Tenure. However, none of variables is recommended to be dropped as neither there are more than 15% to 20% of missing data in any of them nor are they extremely correlated with each other.

d) Missing Value treatment (if applicable)

AccountID	0	AccountID	0
Churn	0	Churn	0
Tenure	218	Tenure	0
City_Tier	112	City_Tier	0
CC_Contacted_LY	102	CC_Contacted_LY	0
Payment	109	Payment	0
Gender	108	Gender	0
Service_Score	98	Service_Score	0
Account_user_count	444	Account_user_count	0
account_segment	97	account_segment	0
CC_Agent_Score	116	CC_Agent_Score	0
Marital_Status	212	Marital_Status	0
rev_per_month	791	rev_per_month	0
Complain_ly	357	Complain_ly	0
rev_growth_yoy	3	rev_growth_yoy	0
coupon_used_for_payment	3	coupon_used_for_payment	0
Day_Since_CC_connect	358	Day_Since_CC_connect	0
cashback	473	cashback	0
Login_device	760	Login_device	0

Figure 11: Missing Values including Null Values and Treated Missing Values

- Most Tree based models can handle missing values. However, there a few models such as neural network which require to impute the missing values.
- Also, the total amount of missing values was only 1.25% excluding null values and hence, imputing them would not have changed the data significantly. Even when added with the count of null values to the missing values, the total values requiring imputation came to only 2.04%. Hence, we decided to impute these values appropriately.
- Already encoded variables in original dataset such as City_Tier, Service_Score, CC_Agent_Score and Complain_ly were imputed them using their respective mode values (as it is the preferred way of imputing categorical variables)
- Balance categorical variables such as Payment, Gender, account_segment, Marital_Status and Login_device were also impute using their respective mode values.
- Continuous variables of Tenure, CC_Contacted_LY, Account_user_count, rev_per_month, rev_growth_yoy, coupon_used_for_payment, Day_Since_CC_connect and cashback were imputed by either their respective median values as they had outliers present in them as imputation with median values is preferred for continuous variables having outliers.

e) Outlier treatment (if required)

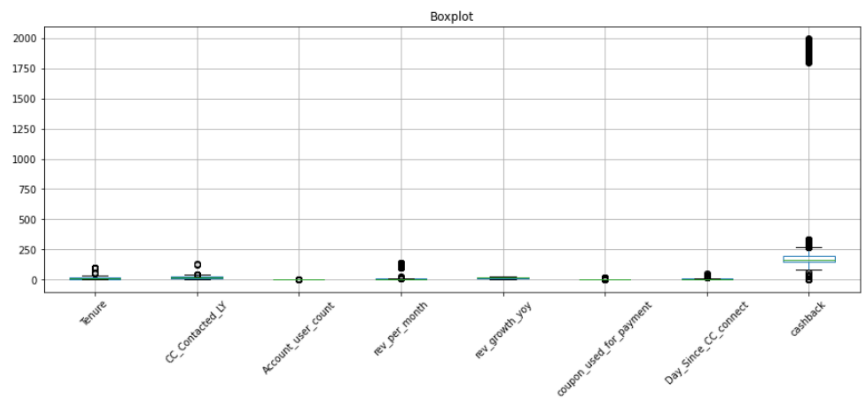


Figure 12: Boxplot with Outliers

- We can see presence of outliers for the numerical variables Tenure, CC_Contacted_LY,Account_user_count, rev_per_month, coupon_used_for_payment, Day_Since_CC_connect and cashback.
- Only rev_growth_yoy variable does not contain outliers. However, outlier presence in Account_user_count also does not seem justified as there could be many users registered under one account and this condition should not be applicable as an outlier.

- Most of the tree based models such as Cart, Random Forest, Neural Network, Bagging are not affected by outliers. However, regression models such as Logistic Regression and boosting techniques such as AdaBoost, XG Boost etc are affected. Hence, we will split our original data in two subsets one which includes the outliers and another one without outliers and build models using both the subsets and check the prediction and performance.

AccountID	0.00
Churn	0.00
Tenure	1.16
City_Tier	0.00
CC_Contacted_LY	0.04
Payment	0.00
Gender	0.00
Service_Score	0.00
Account_user_count	0.00
account_segment	0.00
CC_Agent_Score	0.00
Marital_Status	0.00
rev_per_month	0.94
Complain_ly	0.00
rev_growth_yoy	0.00
coupon_used_for_payment	0.04
Day_Since_CC_connect	0.06
cashback	0.96
Login_device	0.00

Figure 13: Percentage of Outliers basis lower limit of 5% and upper limit of 95% quantiles

- From above we see that the variables Tenure (1.16%), CC_Contacted_LY (0.04%), rev_per_month (0.94%), coupon_used_for_payment (0.04%), Day_Since_CC_connect (0.06%) and cashback (0.96%) have presence of outliers w.r.t 5% and 95% quantile limits. We will treat the outliers basis these quantile limits. The quantile limits are kept as such so that not a large chuck of data is manipulated when treating the outliers keeping in mind the best interest for the business as well.

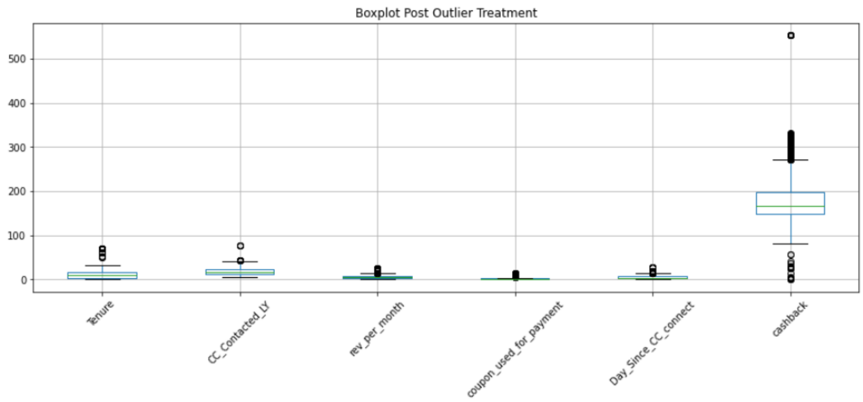


Figure 14: Boxplot post Treating Outliers basis lower limit of 5% and upper limit of 95% quantiles

f) Variable transformation (if applicable)

```
<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)
```

```
<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                 11260 non-null  int64
 3   City_Tier              11260 non-null  int64
 4   CC_Contacted_LY        11260 non-null  int64
 5   Payment                11260 non-null  int8
 6   Gender                 11260 non-null  int8
 7   Service_Score          11260 non-null  int64
 8   Account_user_count     11260 non-null  int64
 9   account_segment        11260 non-null  int8
 10  CC_Agent_Score         11260 non-null  int64
 11  Marital_Status         11260 non-null  int8
 12  rev_per_month          11260 non-null  int64
 13  Complain_ly            11260 non-null  int64
 14  rev_growth_yoy         11260 non-null  int64
 15  coupon_used_for_payment  11260 non-null  int64
 16  Day_Since_CC_connect   11260 non-null  int64
 17  cashback               11260 non-null  float64
 18  Login_device           11260 non-null  int8
dtypes: float64(1), int64(13), int8(5)
```

Figure 15: Transformed Data with Correct Data Types

- The dataset has variables which are object data type. These needs to be converted to integer or float data type as predictive modelling for supervised learning techniques such as Cart, Random Forest require so to build models and predict the results. Hence, we have used label encoding to transform the variables Payment, Gender, account_segment, Marital_status and Login_device having object data type to numerical values as seen in the above figure.
- We have also assigned correct data type of integer to already encoded variables City_Tier, Service_Score, CC_Agent_Score and Complain_ly present in original dataset and numerical variable CC_Contacted_LY and Account_user_count has been changed from float to integer as well in order to keep correct data type across all variables.

g) Addition of new variables (if required)

- As of now we do not see any requirement to create any new variables which would add significance to our model building and/or predicting the customers probability to churn.

4. Business insights from EDA

a) Is the data unbalanced? If so, what can be done? Please explain in the context of the business

```
0     9364        0    0.83
1     1896        1    0.17
Name: Churn,      Name: Churn,
```

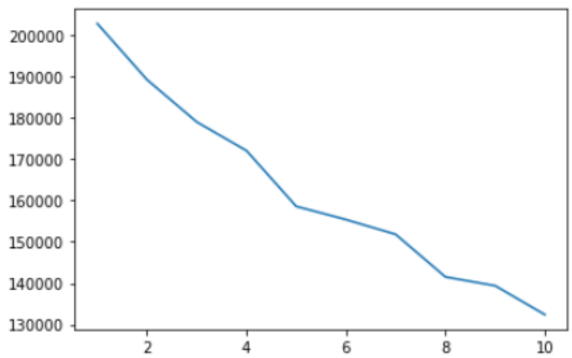
Figure 16: Data Imbalance Proportion

- We observe only 17% of the data belongs to class 1 (customers who have churned) and the rest 83% belongs to class 0 (customers who have not churned). Hence, there seems to be data imbalance but it is not yet clear on what impact will it have on our model building techniques. Once we build our models we will be able determine whether there exists any over-fitting.
- In case any of our model shows signs of over-fitting even after pruning or using various hyper parameters for our model building, then we can confirm there is a class imbalance problem and we can then use the technique of 'SMOTE' (Synthetic Minority

Oversampling Technique) in which the minority class ie; class 1 for our dataset would be over sampled by generating synthesized data. Thus, creating a balanced dataset.

- However, from business perspective, they are highly unlikely to appreciate using the SMOTE technique as it means that the original data would be modified/tampered and the models predictions and performance would also be based on this modified balanced data and not the original data. Moreover, we have also imputed 2.04% of bad data previously. Hence, they could question the reliability of the model built using balanced data on any of their future data and whether that model will actually be the best model for them.
- However, we must try and implement all the best possible ways to build a suitable predictive model for our client.

b) Any business insights using clustering (if applicable)



```
[202680.0000000002,
 189123.20187858335,
 178921.24167248802,
 171990.45733854486,
 158543.1243746025,
 155349.7426388408,
 151777.27212788898,
 141509.88882323386,
 139380.2544146019,
 132391.98663643817]
```

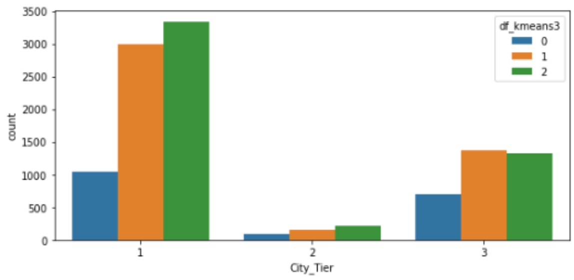
Figure 17: WSS Plot and Scores

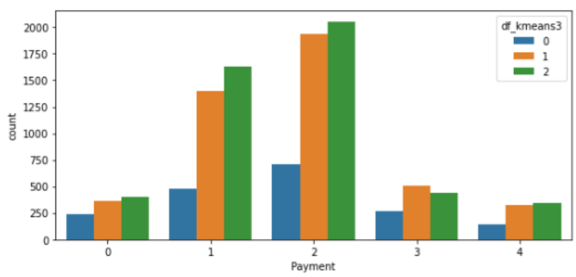
- From the above figure, the sharp decline in WSS figures can be observed till only first two or three clusters only but it does not make proper business sense to keep just 2 segments of customers as it does not offer clear differentiation. Hence, we will go with 3 clusters.

```
0     1841
1     4535
2     4884
```

- The clusters have divided the customers in 3 segments (0,1,2) with segment 0 having 1841 customers, segment 1 having 4535 customers and segment 2 having highest customers of total 4884.

i. Business insights using clustering for truly categorical variables





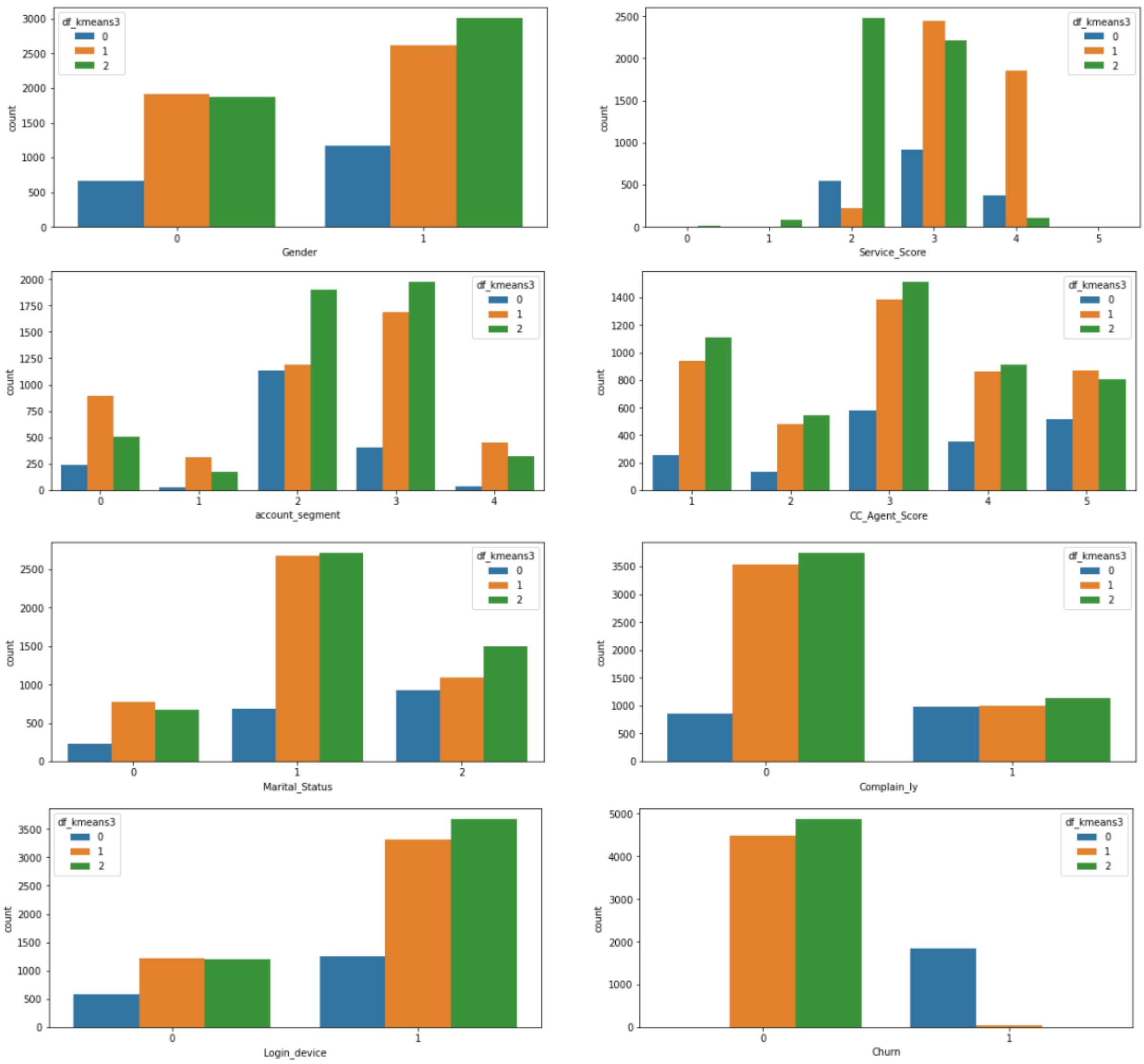


Figure 18: Business insights using clustering for truly categorical variables

- **<u>Segment 0 has highest no of churners and all are churners</u>** despite having only 50% of account holders compared to the other 2 segments. Hence, we will try and focus on patterns for this segment in our analysis.
- Segment 0 customers are lowest for all tiers but significantly lower for tier 1.
- Segment 0 customers prefer 'Debit Card', 'Credit Card', 'Cash on Delivery' in the same order.
- Segment 0 customers have higher Males.
- Segment 0 customers have given Service Score 3, 2 and 4 to company in the same order.
- Segment 0 customers belong mainly to 'Regular Plus', 'Super', 'HNI' in the same order.
- Segment 0 customers have given Service Score 3, 5 and 4 to customer care in the same order.
- Segment 0 customers are higher for Married followed by Divorced given Service Score 3, 5 and 4 to customer care in the same order.
- 50% of customers of segment 0 have raised complaints in last 12 months
- 50% more customers of segment 0 prefer Mobile to Login

ii. Business insights using clustering for truly continuous variables

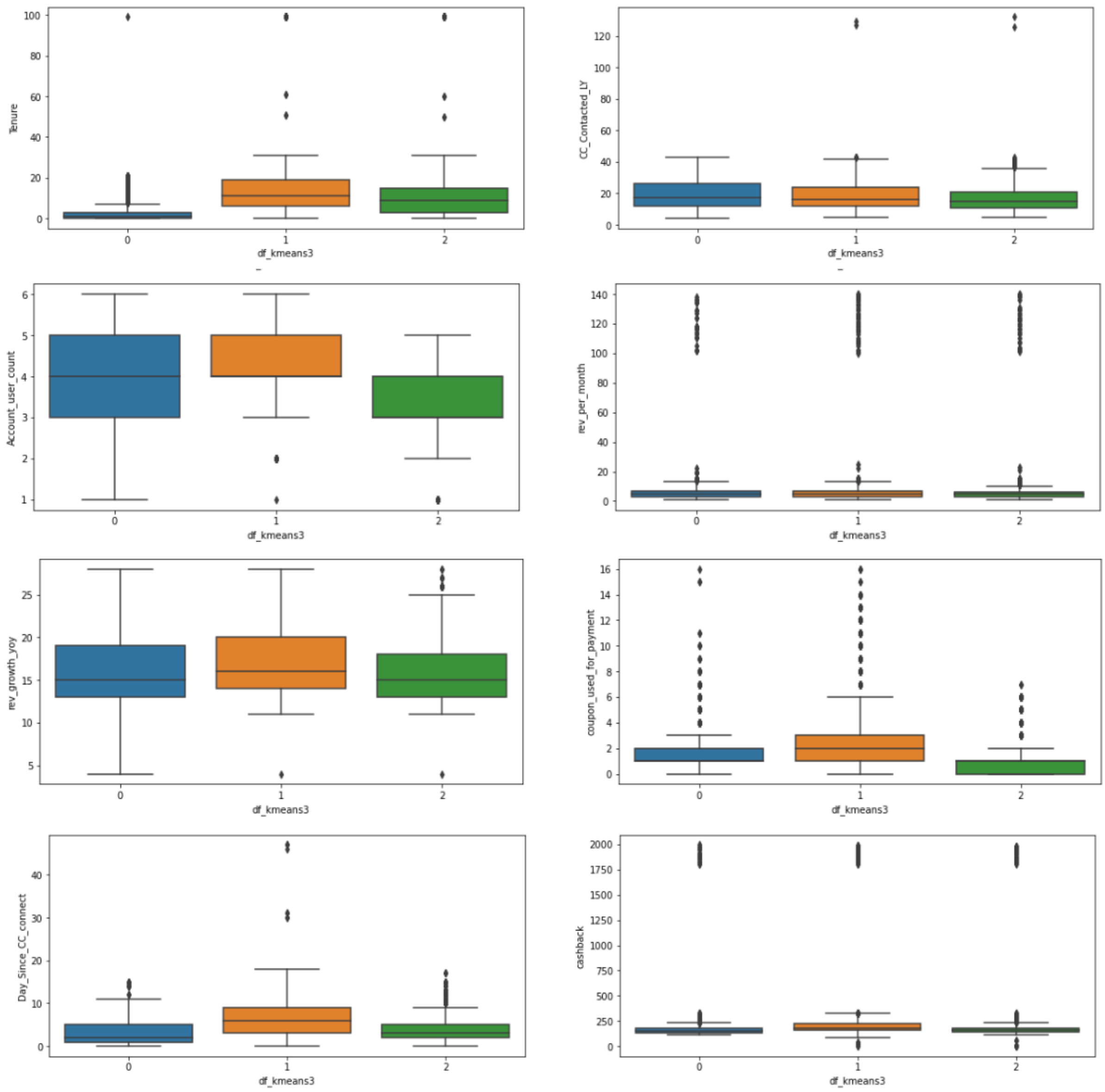


Figure 19: Business insights using clustering for truly continuous variables

- As per Tenure, segment 0 has customers are acquired newly compared to the other segments and ranges between 0 to 20 months.
- Segment 0 customers have contacted a median 19 times in last 12 months.
- Segment 0 customers have a median 4 users per account.
- Segment 0 customers contributes 0 to 1700 units monthly average revenue generated in last 12 months
- Segment 0 customers contributes a median of 15% growth yoy.
- Segment 0 customers connect with customer care more often.

c) Any other business insights

	count	mean	std	min	25%	50%	75%	max
AccountID	11260.00	25629.50	3250.63	20000.00	22814.75	25629.50	28444.25	31259.00
Churn	11260.00	0.17	0.37	0.00	0.00	0.00	0.00	1.00
Tenure	11260.00	10.99	12.76	0.00	2.00	9.00	16.00	99.00
City_Tier	11260.00	1.65	0.91	1.00	1.00	1.00	3.00	3.00
CC_Contacted_LY	11260.00	17.85	8.81	4.00	11.00	16.00	23.00	132.00
Service_Score	11260.00	2.90	0.72	0.00	2.00	3.00	3.00	5.00
Account_user_count	11260.00	3.70	1.00	1.00	3.00	4.00	4.00	6.00
CC_Agent_Score	11260.00	3.07	1.37	1.00	2.00	3.00	4.00	5.00
rev_per_month	11260.00	6.27	11.49	1.00	3.00	5.00	7.00	140.00
Complain_ly	11260.00	0.28	0.45	0.00	0.00	0.00	1.00	1.00
rev_growth_yoy	11260.00	16.19	3.76	4.00	13.00	15.00	19.00	28.00
coupon_used_for_payment	11260.00	1.79	1.97	0.00	1.00	1.00	2.00	16.00
Day_Since_CC_connect	11260.00	4.58	3.65	0.00	2.00	3.00	7.00	47.00
cashback	11260.00	194.93	174.98	0.00	147.89	165.25	197.31	1997.00

	Payment	Gender	account_segment	Marital_Status	Login_device
count	11260	11260	11260	11260	11260
unique	5	2	5	3	2
top	Debit Card	Male	Regular Plus	Married	Mobile
freq	4696	6812	4221	6072	8242

Figure 20: Imputed Data Description

- Average Tenure for an account is 11 months approx. which is a good sign and suggests that the customers do have the potential to have a high life cycle with them.
- A higher number of customers are acquired within these couple of months, which is a good sign of growth and suggests that the business in heading in the right direction and gaining market share.
- City Tier 2 had the least customers which need to be targeted to retain.
- Business is losing more customers from tier 3 compared to customers in this tier they are able to retain. Specific offers and strategies for these customers can be implemented.
- Complaints and Churn are slightly positively correlated. Hence, business need to try and lower this frequency. Moreover, Day_Since_CC_connect and Churn are slightly negatively correlated. Hence, lesser connect with customer care, better probability of retention.
- Account holders who churned had highest average no of users contacting customer care last year of 19 users which is also similar to the overall average for this variable.
- Better Satisfaction Score to company results in more users being added to primary account.
- Coupon_used_for_payment is slightly positively correlated to Day_Since_CC_connect. Need to smoothen the process to use coupons where the need for assistance is reduced.
- Payment mode of UPI is preferred the least. Accounts using cash on delivery and e-wallet are churning proportionately more than others. Ensure all payment modes can be used safely and securely and keep customers informed about the steps the business is taking to build trust and confidence.

- Frequency of cashbacks offered is low which is good as this variable does not seem to affect probability of churn as well.
- Customer Care Service ratings need improvement.
- There is low frequency of primary account holders who are Single. Also, same is the case with Female customers.
- Customers providing High Satisfaction Score tag more users to the account.

Appendix

Raw Codes & Outputs

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
pd.set_option('display.max_rows', None)
pd.set_option('display.max_colwidth', None)
pd.set_option("display.max_columns", None)
```

```
data = pd.read_excel('E:\GL\Course Content\Capstone\Capstone Business Project\CC_EDTH_02_Customer Churn\Customer Churn Data.xlsx'
                     sheet_name='Data for DSBA')
data.head()
```

```
data.tail()
```

```
print('The number of rows (observations) is',data.shape[0],'\n''The number of columns (variables) is',data.shape[1])
```

```
data.info()
```

Check Missing Values in dataset

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

```
data.isnull().sum().sum()
```

```
2676
```

```
data.size
```

```
213940
```

```
round((2676/213940)*100,2)
```

```
1.25
```

Checking Duplicates

```
data.duplicated().sum()
```

```
0
```

Summary

```
pd.options.display.float_format = '{:.2f}'.format
data.describe(include='all').T
```

Check Proportion of Target Variable (Churn)

```
data.Churn.value_counts()
```

```
0    9364
1    1896
Name: Churn, dtype: int64
```

```
data.Churn.value_counts(normalize=True)
```

```
0   0.83
1   0.17
Name: Churn, dtype: float64
```

EDA

```
data.Tenure.value_counts()
```

```
data.Tenure.unique()
```

```
array([4, 0, 2, 13, 11, '#', 9, 99, 19, 20, 14, 8, 26, 18, 5, 30, 7, 1,
       23, 3, 29, 6, 28, 24, 25, 16, 10, 15, 22, nan, 27, 12, 21, 17, 50,
       60, 31, 51, 61], dtype=object)
```

```
data.Tenure = data.Tenure.replace('#', np.NaN)
```

```
data.Tenure.unique()
```

```
array([ 4.,  0.,  2., 13., 11., nan,  9., 99., 19., 20., 14.,  8., 26.,
       18.,  5., 30.,  7.,  1., 23.,  3., 29.,  6., 28., 24., 25., 16.,
       10., 15., 22., 27., 12., 21., 17., 50., 60., 31., 51., 61.])
```

```
data.Tenure.isnull().sum()
```

```
218
```

```
data.City_Tier.value_counts()
```

```
1.00    7263
3.00    3405
2.00     480
```

```
data.City_Tier.unique()
```

```
array([ 3.,  1., nan,  2.])
```

```
data.City_Tier.isnull().sum()
```

```
112
```

```
data.CC_Contacted_LY.value_counts()
```

```
data.CC_Contacted_LY.unique()
```

```
array([  6.,   8.,  30.,  15.,  12.,  22.,  11.,   9.,  31.,  18.,  13.,
        20.,  29.,  28.,  26.,  14.,  10.,  25.,  27.,  17.,  23.,  33.,
        19.,  35.,  24.,  16.,  32.,  21.,  nan,  34.,   5.,   4., 126.,
         7.,  36., 127.,  42.,  38.,  37.,  39.,  40.,  41., 132.,  43.,
       129.])
```

```
data.CC_Contacted_LY.isnull().sum()
```

```
102
```

```
data.Payment.value_counts()
```

```
Debit Card          4587
Credit Card         3511
E wallet            1217
Cash on Delivery    1014
UPI                  822
Name: Payment, dtype: int64
```

```
data.Payment.unique()
```

```
array(['Debit Card', 'UPI', 'Credit Card', 'Cash on Delivery', 'E wallet',
       nan], dtype=object)
```

```
data.Payment.isnull().sum()
```

```
109
```

```
data.Gender.value_counts()
```

```
Male      6328
Female    4178
M          376
F          270
Name: Gender, dtype: int64
```

```
data.Gender = data.Gender.replace("M", 'Male').replace("F", 'Female')
```

```
data.Gender.unique()
```

```
array(['Female', 'Male', nan], dtype=object)
```

```
data.Gender.isnull().sum()
```

```
108
```

```
data.Service_Score.value_counts()
```

```
3.00    5490
2.00    3251
4.00    2331
1.00      77
0.00       8
5.00       5
Name: Service_Score, dtype: int64
```

```
data.Service_Score.unique()
```

```
array([ 3.,  2.,  1., nan,  0.,  4.,  5.])
```

```
data.Service_Score.isnull().sum()
```

```
98
```

```
data.Account_user_count.value_counts()
```

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

```
data.Account_user_count.unique()
```

```
array([3, 4, nan, 5, 2, '@', 1, 6], dtype=object)
```

```
data.Account_user_count.isnull().sum()
```

```
112
```

```
data.Account_user_count = data.Account_user_count.replace('@', np.NaN)
```

```
data.Account_user_count.unique()
```

```
array([ 3.,  4., nan,  5.,  2.,  1.,  6.])
```

```
data.Account_user_count.isnull().sum()
```

```
444
```

```
data.account_segment.value_counts()
```

```
Super           4062
Regular Plus    3862
HNI             1639
Super Plus       771
Regular          520
Regular +        262
Super +           47
Name: account_segment, dtype: int64
```

```
data.account_segment = data.account_segment.replace("Regular +", 'Regular Plus').replace("Super +", 'Super Plus')
```

```
data.account_segment.unique()
```

```
array(['Super', 'Regular Plus', 'Regular', 'HNI', nan, 'Super Plus'],
      dtype=object)
```

```
data.account_segment.isnull().sum()
```

```
97
```

```
data.CC_Agent_Score.value_counts()
```

```
3.00    3360
1.00    2302
5.00    2191
4.00    2127
2.00    1164
Name: CC_Agent_Score, dtype: int64
```

```
data.CC_Agent_Score.unique()
```

```
array([ 2.,  3.,  5.,  4., nan,  1.])
```

```
data.CC_Agent_Score.isnull().sum()
```

```
116
```

```
data.Marital_Status.value_counts()
```

```
Married     5860
Single      3520
Divorced    1668
Name: Marital_Status, dtype: int64
```

```
data.Marital_Status.unique()
```

```
array(['Single', 'Divorced', 'Married', nan], dtype=object)
```

```
data.Marital_Status.isnull().sum()
```

```
212
```

```
data.rev_per_month.value_counts()
```

```
data.rev_per_month.unique()
```

```
array([9, 7, 6, 8, 3, 2, 4, 10, 1, 5, '+', 130, nan, 19, 139, 102, 120,
       138, 127, 123, 124, 116, 21, 126, 134, 113, 114, 108, 140, 133,
       129, 107, 118, 11, 105, 20, 119, 121, 137, 110, 22, 101, 136, 125,
       14, 13, 12, 115, 23, 122, 117, 131, 104, 15, 25, 135, 111, 109,
       100, 103], dtype=object)
```

```
data.rev_per_month.isnull().sum()
```

```
102
```

```
data.rev_per_month = data.rev_per_month.replace('+', np.NaN)
```

```
data.rev_per_month.unique()
```

```
array([  9.,   7.,   6.,   8.,   3.,   2.,   4.,  10.,   1.,   5.,  nan,
       130.,  19., 139., 102., 120., 138., 127., 123., 124., 116.,  21.,
       126., 134., 113., 114., 108., 140., 133., 129., 107., 118.,  11.,
       105.,  20., 119., 121., 137., 110.,  22., 101., 136., 125.,  14.,
        13.,  12., 115.,  23., 122., 117., 131., 104.,  15.,  25., 135.,
       111., 109., 100., 103.])
```

```
data.rev_per_month.isnull().sum()
```

```
791
```

```
data.Complain_ly.value_counts()
```

```
0.00    7792
1.00    3111
Name: Complain_ly, dtype: int64
```

```
data.Complain_ly.unique()
```

```
array([ 1.,  0., nan])
```

```
data.Complain_ly.isnull().sum()
```

```
357
```

```
data.rev_growth_yoy.value_counts()
```

```
data.rev_growth_yoy.unique()
```

```
array([11, 15, 14, 23, 22, 16, 12, 13, 17, 18, 24, 19, 20, 21, 25, 26,
       '$', 4, 27, 28], dtype=object)
```

```
data.rev_growth_yoy = data.rev_growth_yoy.replace('$', np.NaN)
```

```
data.rev_growth_yoy.unique()
```

```
array([11., 15., 14., 23., 22., 16., 12., 13., 17., 18., 24., 19., 20.,
       21., 25., 26., nan,  4., 27., 28.])
```

```
data.rev_growth_yoy.isnull().sum()
```

```
3
```

```
data.coupon_used_for_payment.value_counts()
```

```
data.coupon_used_for_payment.unique()
```

```
array([1, 0, 4, 2, 9, 6, 11, 7, 12, 10, 5, 3, 13, 15, 8, '#', '$', 14,
       '*', 16], dtype=object)
```

```
data.coupon_used_for_payment.isnull().sum()
```

```
0
```

```
data.coupon_used_for_payment = data.coupon_used_for_payment.replace('#', np.NaN).replace('$', np.NaN).replace('*', np.NaN)
```

```
data.coupon_used_for_payment.unique()
```

```
array([ 1.,  0.,  4.,  2.,  9.,  6., 11.,  7., 12., 10.,  5.,  3., 13.,
       15.,  8., nan, 14., 16.])
```

```
data.rev_growth_yoy.isnull().sum()
```

```
3
```

```
data.Day_Since_CC_connect.value_counts()
```

```
data.Day_Since_CC_connect.unique()
```

```
array([5, 0, 3, 7, 2, 1, 8, 6, 4, 15, nan, 11, 10, 9, 13, 12, 17, 16, 14,
       30, '$', 46, 18, 31, 47], dtype=object)
```

```
data.Day_Since_CC_connect.isnull().sum()
```

```
357
```

```
data.Day_Since_CC_connect = data.Day_Since_CC_connect.replace('$', np.NaN)
```

```
data.Day_Since_CC_connect.unique()
```

```
array([ 5.,  0.,  3.,  7.,  2.,  1.,  8.,  6.,  4., 15., nan, 11., 10.,
        9., 13., 12., 17., 16., 14., 30., 46., 18., 31., 47.])
```

```
data.Day_Since_CC_connect.isnull().sum()
```

```
358
```

```
data.cashback.value_counts()
```

```
data.cashback.unique()
```

```
array([159.93, 120.9, nan, ..., 227.36, 226.91, 191.42], dtype=object)
```

```
data.cashback.isnull().sum()
```

```
471
```

```
data.cashback = data.cashback.replace('$', np.NaN)
```

```
data.cashback.unique()
```

```
array([159.93, 120.9 ,    nan, ..., 227.36, 226.91, 191.42])
```

```
data.cashback.isnull().sum()
```

```
473
```

```
data.Login_device.value_counts()
```

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

```
data.Login_device.unique()
```

```
array(['Mobile', 'Computer', '&&&&', nan], dtype=object)
```

```
data.Login_device.isnull().sum()
```

```
221
```

```
data.Login_device = data.Login_device.replace('&&&&', np.NaN)
```

```
data.Login_device.unique()
```

```
array(['Mobile', 'Computer', nan], dtype=object)
```

```
data.Login_device.isnull().sum()
```

```
760
```

```
data.isnull().sum().sum()
```

```
4361
```

```
data.size
```

```
213940
```

```
round((4361/213940)*100,2)
```

```
2.04
```

```
data.isnull().sum().sort_values(ascending = False)/data.index.size
```

```
rev_per_month              0.07
Login_device               0.07
cashback                   0.04
Account_user_count         0.04
Day_Since_CC_connect       0.03
Complain_ly                0.03
Tenure                     0.02
Marital_Status             0.02
CC_Agent_Score             0.01
City_Tier                  0.01
Payment                    0.01
Gender                     0.01
CC_Contacted_LY            0.01
Service_Score              0.01
account_segment            0.01
rev_growth_yoy             0.00
coupon_used_for_payment    0.00
Churn                      0.00
AccountID                  0.00
dtype: float64
```

Impute Missing Values

```
sns.boxplot(x='Tenure', data=data)
```

```
data.Tenure.median()
```

```
9.0
```

```
data[data.Tenure == 9].shape[0]
```

```
496
```

```
data.Tenure.isnull().sum()
```

```
218
```

```
data.Tenure = data.Tenure.fillna(data.Tenure.median())
data[data.Tenure.isnull()]
```

AccountID	Churn	Tenure	City_Tier	CC_Contacted_LY	Payment	Gender	Service_Score	Account_user_count	account_segment	CC_Agent_Score	Marital_St

```
data[data.Tenure == 9].shape[0]
```

```
714
```

```
data.Tenure = data.Tenure.astype('int64')
```

```
data.Tenure.dtype
```

```
dtype('int64')
```

```
sns.boxplot(x='CC_Contacted_LY', data=data)
```

```
data.Account_user_count.median()
```

```
4.0
```

```
data[data.Account_user_count == 4].shape[0]
```

```
4569
```

```
data.Account_user_count.isnull().sum()
```

```
444
```

```
data.Account_user_count = data.Account_user_count.fillna(data.Account_user_count.median())
data[data.Account_user_count.isnull()]
```

```
data[data.Account_user_count == 4].shape[0]
```

```
5013
```

```
data.Account_user_count = data.Account_user_count.astype('int64')
```

```
data.Account_user_count.dtype
```

```
dtype('int64')
```

```
sns.boxplot(x='rev_per_month', data=data)
```

```
data.rev_per_month.median()
```

```
5.0
```

```
data[data.rev_per_month == 5].shape[0]
```

```
1337
```

```
data.rev_per_month.isnull().sum()
```

```
791
```

```
data.rev_per_month = data.rev_per_month.fillna(data.rev_per_month.median())
data[data.rev_per_month.isnull()]
```

```
data[data.rev_per_month == 5].shape[0]
```

```
2128
```

```
data.rev_per_month = data.rev_per_month.astype('int64')
```

```
data.rev_per_month.dtype
```

```
dtype('int64')
```

```
sns.boxplot(x='rev_growth_yoy', data=data)
```

```python
data.rev_growth_yoy.median()
```

15.0

```python
data[data.rev_growth_yoy == 15].shape[0]
```

1283

```python
data.rev_growth_yoy.isnull().sum()
```

3

```python
data.rev_growth_yoy = data.rev_growth_yoy.fillna(data.rev_growth_yoy.median())
data[data.rev_growth_yoy.isnull()]
```

AccountID	Churn	Tenure	City_Tier	CC_Contacted_LY	Payment	Gender	Service_Score	Account_user_count	account_segment	CC_Agent_Score	Marital_St

```python
data[data.rev_growth_yoy == 15].shape[0]
```

1286

```python
data.rev_growth_yoy = data.rev_growth_yoy.astype('int64')
```

```python
data.rev_growth_yoy.dtype
```

dtype('int64')

```python
sns.boxplot(x='coupon_used_for_payment', data=data)
```

```python
data.coupon_used_for_payment.median()
```

1.0

```python
data[data.coupon_used_for_payment == 1].shape[0]
```

4373

```python
data.coupon_used_for_payment.isnull().sum()
```

3

```python
data.coupon_used_for_payment = data.coupon_used_for_payment.fillna(data.coupon_used_for_payment.median())
data[data.coupon_used_for_payment.isnull()]
```

AccountID	Churn	Tenure	City_Tier	CC_Contacted_LY	Payment	Gender	Service_Score	Account_user_count	account_segment	CC_Agent_Score	Marital_St

```python
data[data.coupon_used_for_payment == 1].shape[0]
```

4376

```python
data.coupon_used_for_payment = data.coupon_used_for_payment.astype('int64')
```

```python
data.coupon_used_for_payment.dtype
```

dtype('int64')

```python
sns.boxplot(x='Day_Since_CC_connect', data=data)
```

```python
data.Day_Since_CC_connect.median()
```

3.0

```python
data[data.Day_Since_CC_connect == 3].shape[0]
```

1816

```python
data.Day_Since_CC_connect.isnull().sum()
```

358

```python
data.Day_Since_CC_connect = data.Day_Since_CC_connect.fillna(data.Day_Since_CC_connect.median())
data[data.Day_Since_CC_connect.isnull()]
```

AccountID	Churn	Tenure	City_Tier	CC_Contacted_LY	Payment	Gender	Service_Score	Account_user_count	account_segment	CC_Agent_Score	Marital_St

```python
data[data.Day_Since_CC_connect == 3].shape[0]
```

2174

```python
data.Day_Since_CC_connect = data.Day_Since_CC_connect.astype('int64')
```

```python
data.Day_Since_CC_connect.dtype
```

```
dtype('int64')
```

```python
sns.boxplot(x='cashback', data=data)
```

```python
data.cashback.median()
```

```
165.25
```

```python
data.cashback.isnull().sum()
```

```
473
```

```python
data.cashback = data.cashback.fillna(data.cashback.median())
data[data.cashback.isnull()]
```

AccountID	Churn	Tenure	City_Tier	CC_Contacted_LY	Payment	Gender	Service_Score	Account_user_count	account_segment	CC_Agent_Score	Marital_St

```python
data.cashback.dtype
```

```
dtype('float64')
```

```python
data.info()
```

```python
data.City_Tier.isnull().sum()
```

```
112
```

```python
data.City_Tier.mode()
```

```
0    1.00
dtype: float64
```

```python
data[data.City_Tier == 1].shape[0]
```

```
7263
```

```python
data.City_Tier = data.City_Tier.fillna(data.City_Tier.mode()[0])
data[data.City_Tier.isnull()]
```

AccountID	Churn	Tenure	City_Tier	CC_Contacted_LY	Payment	Gender	Service_Score	Account_user_count	account_segment	CC_Agent_Score	Marital_St

```python
data[data.City_Tier == 1].shape[0]
```

```
7375
```

```python
data.City_Tier = data.City_Tier.astype('int64')
```

```python
data.City_Tier.dtype
```

```
dtype('int64')
```

```python
data.Service_Score.isnull().sum()
```

```
98
```

```python
data.Service_Score.mode()
```

```
0    3.00
dtype: float64
```

```python
data[data.Service_Score == 3].shape[0]
```

```
5490
```

```python
data.Service_Score = data.Service_Score.fillna(data.Service_Score.mode()[0])
data[data.Service_Score.isnull()]
```

```
data[data.Service_Score == 3].shape[0]
```

```
5588
```

```
data.Service_Score = data.Service_Score.astype('int64')
```

```
data.Service_Score.dtype
```

```
dtype('int64')
```

```
data.CC_Agent_Score.isnull().sum()
```

```
116
```

```
data.CC_Agent_Score.mode()
```

```
0    3.00
dtype: float64
```

```
data[data.CC_Agent_Score == 3].shape[0]
```

```
3360
```

```
data.CC_Agent_Score = data.CC_Agent_Score.fillna(data.CC_Agent_Score.mode()[0])
data[data.CC_Agent_Score.isnull()]
```

AccountID	Churn	Tenure	City_Tier	CC_Contacted_LY	Payment	Gender	Service_Score	Account_user_count	account_segment	CC_Agent_Score	Marital_St

```
data[data.CC_Agent_Score == 3].shape[0]
```

```
3476
```

```
data.CC_Agent_Score = data.CC_Agent_Score.astype('int64')
```

```
data.CC_Agent_Score.dtype
```

```
dtype('int64')
```

```
data.Complain_ly.isnull().sum()
```

```
357
```

```
data.Complain_ly.mode()
```

```
0    0.00
dtype: float64
```

```
data[data.Complain_ly == 0].shape[0]
```

```
7792
```

```
data.Complain_ly = data.Complain_ly.fillna(data.Complain_ly.mode()[0])
data[data.Complain_ly.isnull()]
```

AccountID	Churn	Tenure	City_Tier	CC_Contacted_LY	Payment	Gender	Service_Score	Account_user_count	account_segment	CC_Agent_Score	Marital_St

```
data[data.Complain_ly == 0].shape[0]
```

```
8149
```

```
data.Complain_ly = data.Complain_ly.astype('int64')
```

```
data.Complain_ly.dtype
```

```
dtype('int64')
```

```
data.Payment.isnull().sum()
```

```
109
```

```
data.Payment.mode()
```

```
0    Debit Card
dtype: object
```

```
data.Payment.describe()
```

```
data.Payment = data.Payment.fillna(data.Payment.mode()[0])
data[data.Payment.isnull()]
```

```
data.Gender.isnull().sum()
```

```
108
```

```
data.Gender.mode()
```

```
0    Male
dtype: object
```

```
data.Gender.describe()
```

```
data.Gender = data.Gender.fillna(data.Gender.mode()[0])
data[data.Gender.isnull()]
```

```
data.account_segment.isnull().sum()
```

```
97
```

```
data.account_segment.mode()
```

```
0    Regular Plus
dtype: object
```

```
data.account_segment.describe()
```

```
data.account_segment = data.account_segment.fillna(data.account_segment.mode()[0])
data[data.account_segment.isnull()]
```

```
data.Marital_Status.isnull().sum()
```

```
212
```

```
data.Marital_Status.mode()
```

```
0    Married
dtype: object
```

```
data.Marital_Status = data.Marital_Status.fillna(data.Marital_Status.mode()[0])
data[data.Marital_Status.isnull()]
```

```
data.Login_device.isnull().sum()
```

```
760
```

```
data.Login_device.mode()
```

```
0    Mobile
dtype: object
```

```
data.Login_device.describe()
```

```
data.Login_device = data.Login_device.fillna(data.Login_device.mode()[0])
data[data.Login_device.isnull()]
```

```
data.Login_device.describe()
```

```
data.info()
```

```
data.isnull().sum()
```

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

Univariate Analysis (Numerical Variables)

```
cont = data.select_dtypes(include = ['float64', 'int64'])
lstnumericcolumns = list(cont.columns.values)
len(lstnumericcolumns)
```

14

```
cont = data.select_dtypes(include = ['float64', 'int64'])
cols = list(cont.columns)
for col in cols:
    print(col)
    print('Skew:', np.round(data[col].skew(),2))
    plt.figure(figsize=(25,6))
    plt.subplot(1,2,1)
    sns.distplot(data[col],norm_hist=False,kde=False,hist_kws=dict(edgecolor='black',linewidth=1.5))
    plt.vlines(data[col].mean(),ymin=0, ymax=100, color = 'red', linewidth=3)
    plt.vlines(data[col].median(),ymin=0, ymax=100, color = 'orange', linewidth=3)
    plt.ylabel('count')
    plt.subplot(1,2,2)
    sns.boxplot(data[col])
    plt.show()
```

```
plt.figure(figsize=(20,3))
sns.countplot(x='Tenure', data=data, palette='pastel')
```

```
plt.figure(figsize=(20,3))
sns.countplot(x='CC_Contacted_LY', data=data, palette='pastel')
```

```
plt.figure(figsize=(5,3))
sns.countplot(x='Account_user_count', data=data, palette='pastel')
```

```
plt.figure(figsize=(20,3))
sns.countplot(x='rev_per_month', data=data, palette='pastel')
```

```
plt.figure(figsize=(20,3))
sns.countplot(x='rev_growth_yoy', data=data, palette='pastel')
```

```
plt.figure(figsize=(10,3))
sns.countplot(x='coupon_used_for_payment', data=data, palette='pastel')
```

```
plt.figure(figsize=(10,3))
sns.countplot(x='Day_Since_CC_connect', data=data, palette='pastel')
```

```
plt.figure(figsize=(5,3))
sns.countplot(x='City_Tier', data=data, palette='pastel')
```

```
plt.figure(figsize=(7,3))
sns.countplot(x='Payment', data=data, palette='pastel')
```

```
plt.figure(figsize=(5,3))
sns.countplot(x='Gender', data=data, palette='pastel')
```

```
plt.figure(figsize=(5,3))
sns.countplot(x='Service_Score', data=data, palette='pastel')
```

```
plt.figure(figsize=(5,3))
sns.countplot(x='account_segment', data=data, palette='pastel')
```

```
plt.figure(figsize=(5,3))
sns.countplot(x='CC_Agent_Score', data=data, palette='pastel')
```

```
plt.figure(figsize=(5,3))
sns.countplot(x='Marital_Status', data=data, palette='pastel')
```

```
plt.figure(figsize=(5,3))
sns.countplot(x='Complain_ly', data=data, palette='pastel')
```

```
plt.figure(figsize=(5,3))
sns.countplot(x='Login_device', data=data, palette='pastel')
```

Bivariate Analysis (Numerical Variables)

```
sns.pairplot(data, hue ='Churn')
```

```
data_plot = data[['Tenure', 'Churn', 'CC_Contacted_LY', 'Account_user_count', 'rev_per_month',
                 'rev_growth_yoy', 'coupon_used_for_payment','Day_Since_CC_connect','cashback']]
data_plot.head()
```

```
sns.pairplot(data_plot, hue ='Churn')
```

```
fig, axes = plt.subplots(nrows=4,ncols=2)
fig.set_size_inches(20,20)
a = sns.boxplot(x='Churn', y='Tenure', data=data, ax = axes[0][0])
a = sns.boxplot(x='Churn', y='CC_Contacted_LY', data=data, ax = axes[0][1])
a = sns.boxplot(x='Churn', y='Account_user_count', data=data, ax=axes[1][0])
a = sns.boxplot(x='Churn', y='rev_per_month', data=data, ax=axes[1][1])
a = sns.boxplot(x='Churn', y='rev_growth_yoy', data=data, ax = axes[2][0])
a = sns.boxplot(x='Churn', y='coupon_used_for_payment', data=data, ax = axes[2][1])
a = sns.boxplot(x='Churn', y='Day_Since_CC_connect', data=data, ax = axes[3][0])
a = sns.boxplot(x='Churn', y='cashback', data=data, ax = axes[3][1])
```

Bivariate Analysis (Categorical Variables)

```
fig, axes = plt.subplots(nrows=5,ncols=2)
fig.set_size_inches(20,24)
a = sns.countplot(x='City_Tier', hue='Churn', data=data, ax = axes[0][0])
a = sns.countplot(x='Payment', hue='Churn', data=data, ax = axes[0][1])
a = sns.countplot(x='Gender', hue='Churn', data=data, ax=axes[1][0])
a = sns.countplot(x='Service_Score', hue='Churn', data=data, ax=axes[1][1])
a = sns.countplot(x='account_segment', hue='Churn', data=data, ax = axes[2][0])
a = sns.countplot(x='CC_Agent_Score', hue='Churn', data=data, ax = axes[2][1])
a = sns.countplot(x='Marital_Status', hue='Churn', data=data, ax = axes[3][0])
a = sns.countplot(x='Complain_ly', hue='Churn', data=data, ax = axes[3][1])
a = sns.countplot(x='Login_device', hue='Churn', data=data, ax = axes[4][0])
```

Multivariate Analysis (HeatMap)

```
plt.figure(figsize=(25,10))
sns.heatmap(data.corr(),annot=True,fmt=".2f", cmap='Blues')
plt.title("Correlation Heatmap")
plt.xticks(rotation=45)
plt.show()
```

Encoding the Object Variables (using Label Encoding)

```
for feature in data.columns:
    if data[feature].dtype == 'object':
        print('\n')
        print('feature:',feature)
        print(pd.Categorical(data[feature].unique()))
        print(pd.Categorical(data[feature].unique()).codes)
        data[feature] = pd.Categorical(data[feature]).codes
```

```
feature: Payment
['Debit Card', 'UPI', 'Credit Card', 'Cash on Delivery', 'E wallet']
Categories (5, object): ['Cash on Delivery', 'Credit Card', 'Debit Card', 'E wallet', 'UPI']
[2 4 1 0 3]


feature: Gender
['Female', 'Male']
Categories (2, object): ['Female', 'Male']
[0 1]


feature: account_segment
['Super', 'Regular Plus', 'Regular', 'HNI', 'Super Plus']
Categories (5, object): ['HNI', 'Regular', 'Regular Plus', 'Super', 'Super Plus']
[3 2 1 0 4]


feature: Marital_Status
['Single', 'Divorced', 'Married']
Categories (3, object): ['Divorced', 'Married', 'Single']
[2 0 1]
feature: Login_device
['Mobile', 'Computer']
Categories (2, object): ['Computer', 'Mobile']
[1 0]
```

```
data.info()
```

Outlier Treatment

```
data[['Tenure', 'CC_Contacted_LY', 'Account_user_count', 'rev_per_month', 'rev_growth_yoy', 'coupon_used_for_payment',
      'Day_Since_CC_connect','cashback']].boxplot(figsize=(15,5))
plt.title("Boxplot")
plt.xticks(rotation=45)
plt.show()
```

```
Q1 = data.quantile(0.05)
Q3 = data.quantile(0.95)
IQR = Q3 - Q1
pd.DataFrame((((data < (Q1 - 1.5 * IQR)) | (data > (Q3 + 1.5 * IQR))).sum()/data.shape[0]*100))
```

Split Data Keeping Outliers (xo & yo)

```
xo = data.drop(['Churn','AccountID'], axis = 1)
yo = data['Churn']
```

```
xo.shape
```

```
(11260, 17)
```

```
yo.shape
```

```
(11260,)
```

Split Data to Remove Outliers (x & y)

```
x = data.drop(['Churn','AccountID'], axis = 1)
y = data['Churn']
```

```
x.shape
```

```
(11260, 17)
```

```
y.shape
```

```
(11260,)
```

```
def remove_outlier(col):
    sorted(col)
    Q1,Q3=col.quantile([0.05,0.95])
    IQR=Q3-Q1
    lower_range= Q1-(1.5 * IQR)
    upper_range= Q3+(1.5 * IQR)
    return lower_range, upper_range
```

```
lw,up=remove_outlier(x['Tenure'])
x['Tenure']=np.where(x['Tenure']>up,up,x['Tenure'])
x['Tenure']=np.where(x['Tenure']<lw,lw,x['Tenure'])

lw,up=remove_outlier(x['CC_Contacted_LY'])
x['CC_Contacted_LY']=np.where(x['CC_Contacted_LY']>up,up,x['CC_Contacted_LY'])
x['CC_Contacted_LY']=np.where(x['CC_Contacted_LY']<lw,lw,x['CC_Contacted_LY'])

lw,up=remove_outlier(x['rev_per_month'])
x['rev_per_month']=np.where(x['rev_per_month']>up,up,x['rev_per_month'])
x['rev_per_month']=np.where(x['rev_per_month']<lw,lw,x['rev_per_month'])

lw,up=remove_outlier(x['coupon_used_for_payment'])
x['coupon_used_for_payment']=np.where(x['coupon_used_for_payment']>up,up,x['coupon_used_for_payment'])
x['coupon_used_for_payment']=np.where(x['coupon_used_for_payment']<lw,lw,x['coupon_used_for_payment'])

lw,up=remove_outlier(x['Day_Since_CC_connect'])
x['Day_Since_CC_connect']=np.where(x['Day_Since_CC_connect']>up,up,x['Day_Since_CC_connect'])
x['Day_Since_CC_connect']=np.where(x['Day_Since_CC_connect']<lw,lw,x['Day_Since_CC_connect'])

lw,up=remove_outlier(x['cashback'])
x['cashback']=np.where(x['cashback']>up,up,x['cashback'])
x['cashback']=np.where(x['cashback']<lw,lw,x['cashback'])

x.shape
```

```
(11260, 17)
```

```
x[['Tenure', 'CC_Contacted_LY', 'rev_per_month', 'coupon_used_for_payment','Day_Since_CC_connect',
   'cashback']].boxplot(figsize=(15,5))
plt.title("Boxplot Post Outlier Treatment")
plt.xticks(rotation=45)
plt.show()
```

K-Means Clustering

```
from sklearn.cluster import KMeans
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import silhouette_samples, silhouette_score
```

```
df = data.drop(['AccountID'], axis = 1)
```

```
df_scaled = StandardScaler().fit_transform(df)
```

```
df_scaled
```

Calculating WSS for values of K - Elbow Method

```
wss =[]
```

```
for i in range(1,11):
    KM = KMeans(n_clusters=i,random_state=1)
    KM.fit(df_scaled)
    wss.append(KM.inertia_)
```

```
wss
```

```
plt.plot(range(1,11), wss)
```

```
k_means = KMeans(n_clusters = 2,random_state=1)
k_means.fit(df_scaled)
labels = k_means.labels_
```

```
silhouette_score(df_scaled,labels)
```

```
0.12336437604603798
```

```
silhouette_samples(df_scaled,labels).min()
```

```
0.0013580240285513245
```

```
k_means = KMeans(n_clusters = 3,random_state=1)
k_means.fit(df_scaled)
labels = k_means.labels_
```

```
silhouette_score(df_scaled,labels)
```

```
0.0742940376682887
```

```
silhouette_samples(df_scaled,labels).min()
```

```
-0.07283738766512156
```

```
k_means = KMeans(n_clusters = 4,random_state=1)
k_means.fit(df_scaled)
labels = k_means.labels_
```

```
silhouette_score(df_scaled,labels)
```

```
0.07424420609512918
```

```
silhouette_samples(df_scaled,labels).min()
```

```
-0.17533947454568208
```

```
df["df_kmeans3"] = labels
df.head()
```

```
df.df_kmeans3.value_counts().sort_index()
```

```
clust_profile=df
clust_profile=clust_profile.groupby('df_kmeans3').mean()
clust_profile['df_cust_segment']=df.df_kmeans3.value_counts().sort_index()
np.round(clust_profile,2)
```

```
fig, axes = plt.subplots(nrows=5,ncols=2)
fig.set_size_inches(20,24)
a = sns.countplot(x='City_Tier', hue='df_kmeans3', data=df, ax = axes[0][0])
a = sns.countplot(x='Payment', hue='df_kmeans3', data=df, ax = axes[0][1])
a = sns.countplot(x='Gender', hue='df_kmeans3', data=df, ax=axes[1][0])
a = sns.countplot(x='Service_Score', hue='df_kmeans3', data=df, ax=axes[1][1])
a = sns.countplot(x='account_segment', hue='df_kmeans3', data=df, ax = axes[2][0])
a = sns.countplot(x='CC_Agent_Score', hue='df_kmeans3', data=df, ax = axes[2][1])
a = sns.countplot(x='Marital_Status', hue='df_kmeans3', data=df, ax = axes[3][0])
a = sns.countplot(x='Complain_ly', hue='df_kmeans3', data=df, ax = axes[3][1])
a = sns.countplot(x='Login_device', hue='df_kmeans3', data=df, ax = axes[4][0])
a = sns.countplot(x='Churn', hue='df_kmeans3', data=df, ax = axes[4][1])
```

```
fig, axes = plt.subplots(nrows=4,ncols=2)
fig.set_size_inches(20,20)
a = sns.boxplot(x='df_kmeans3', y='Tenure', data=df, ax = axes[0][0])
a = sns.boxplot(x='df_kmeans3', y='CC_Contacted_LY', data=df, ax = axes[0][1])
a = sns.boxplot(x='df_kmeans3', y='Account_user_count', data=df, ax=axes[1][0])
a = sns.boxplot(x='df_kmeans3', y='rev_per_month', data=df, ax=axes[1][1])
a = sns.boxplot(x='df_kmeans3', y='rev_growth_yoy', data=df, ax = axes[2][0])
a = sns.boxplot(x='df_kmeans3', y='coupon_used_for_payment', data=df, ax = axes[2][1])
a = sns.boxplot(x='df_kmeans3', y='Day_Since_CC_connect', data=df, ax = axes[3][0])
a = sns.boxplot(x='df_kmeans3', y='cashback', data=df, ax = axes[3][1])
```

```
df.to_csv('df_Capstone_Kmeans.csv',index=False)
```

THE END