CAPSTONE PROJECT NOTE – 1

DSBA

Content of Report

Introduction of the business problem

Problem Statement: -

An DTH 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 DTH 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.

we have been assigned to develop a churn prediction model for this company and provide business recommendations on the campaign. The model or campaign has to be unique and has to sharp when offers are suggested. The offers suggested should have a win-win situation for company as well as customers so that company doesn't hit on revenue and on the other hand able to retain the customers.

Need of the study/project

This study/project is very essential for the client to **plan for future** in terms of product designing, sales or in rolling out different offers for different segment of clients. The outcome of this project will give a clear understanding where the firm stands now and what's the capacity it holds in terms for taking **risk**. It will also denote what's the future prospective of the organization and how they can make it even better and can plan better for the same and can help them **retaining customers** in a longer run.

Understanding business/social opportunity

This a case study of a DTH company where in they have customers assigned with unique account ID and a single account ID can hold many customers (like family plan) across gender and marital status, customers get flexibility in terms of mode of payment they want to opt for. Customers are again segmented across various types of plans they opt for as per their usage which also based on the device they use (computer or mobile) moreover they ear cashbacks on bill payment.

The overall business runs in customers loyalty and stickiness which in-turn comes from providing quality and value-added services. Also, running various promotional and festivals offers may help organization in getting new customers and also retaining the old one.

We can conclude that a customer retained is a regular income for organization, a customer added is a new income for organization and a customers lost will be a negative impact as a single account ID holds multiple number of customers i.e.; closure of one account ID means loosing multiple customers.

It's a great opportunity for the company as it's a need of almost every individual of family to have a DTH connection which in-turn also leads to increase and competition. Question arises

how can a company creates difference when compared to other competitors, what are the parameter plays a vital role having customers loyalty and making them stay. All these social responsibilities will decide the best player in the market.

Data Report

Dataset of problem: - Customer Churn Data

Data Dictionary: -

- **AccountID --** account unique identifier
- **Churn --** account churn flag (Target Variable)
- **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

Data Ingestion: -

Loaded the required packages, set the work directory and load the datafile.

Data set has 11,260 number of observations and 19 variables (18 independent and 1 dependent or target variable).

	AccountID	Churn	Tenure	City_Tier	CC_Contacted_LY	Payment	Gender	Service_Score	Account_user_count	account_segment	...
0	20000	1	4	3	6	1	1	3	3.0	1	...
1	20001	1	0	1	8	2	2	3	4.0	2	...
2	20002	1	0	1	30	1	2	2	4.0	2	...
3	20003	1	0	3	15	1	2	2	4.0	1	...
4	20004	1	0	1	12	3	2	2	3.0	2	...

Table 1 – glimpse of data-frame head with top 5 rows

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

- data has been collected for random 11,260 unique account ID, across gender and marital status.
- Looking at variables "CC_Contacted_L12m", "rev_per_month", "Complain_l12m", "rev_growth_yoy", "coupon_used_l12m", "Day_Since_CC_connect" and "cashback_l12m" we can conclude that the data has been collected for last 12 month.
- Data has 19 variables, 18 independent and 1 dependent or the target variable, which shows if customer churned or not.
- The data is the combination of services customers are using along with their payment option and also then basic individuals details as well.
- Data is mixed of categorical as well as continuous variables.

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

- Data has 11,260 rows and 19 variables.

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

Table 2:- Dataset Information

```
The shape of dataset is :(11260, 19)
```

Fig 1:- Shape of dataset

- Describing data: - This shows description of variation in various statistical measurements across variables which denotes that each variable is unique and different.

	count	unique	top	freq	mean	std	min	25%	50%	75%	max
AccountID	11260.0	NaN	NaN	NaN	25629.5	3250.62635	20000.0	22814.75	25629.5	28444.25	31259.0
Churn	11260.0	NaN	NaN	NaN	0.168384	0.374223	0.0	0.0	0.0	0.0	1.0
Tenure	11158.0	38.0	1.0	1351.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN
City_Tier	11148.0	NaN	NaN	NaN	1.653929	0.915015	1.0	1.0	1.0	3.0	3.0
CC_Contacted_LY	11158.0	NaN	NaN	NaN	17.867091	8.853269	4.0	11.0	16.0	23.0	132.0
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	11162.0	NaN	NaN	NaN	2.902526	0.725584	0.0	2.0	3.0	3.0	5.0
Account_user_count	11148.0	7.0	4.0	4569.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN
account_segment	11163	7	Super	4062	NaN	NaN	NaN	NaN	NaN	NaN	NaN
CC_Agent_Score	11144.0	NaN	NaN	NaN	3.066493	1.379772	1.0	2.0	3.0	4.0	5.0
Marital_Status	11048	3	Married	5860	NaN	NaN	NaN	NaN	NaN	NaN	NaN
rev_per_month	11158.0	59.0	3.0	1746.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Complain_ly	10903.0	NaN	NaN	NaN	0.285334	0.451594	0.0	0.0	0.0	1.0	1.0
rev_growth_yoy	11260.0	20.0	14.0	1524.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN
coupon_used_for_payment	11260.0	20.0	1.0	4373.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Day_Since_CC_connect	10903.0	24.0	3.0	1816.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN
cashback	10789.0	5693.0	155.62	10.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Login_device	11039	3	Mobile	7482	NaN	NaN	NaN	NaN	NaN	NaN	NaN

Table 3: - Describing Dataset

- Except variables “AccountID”, “Churn”, “rev_growth_yoy” and “coupon_used_for_payment” all other variables have null values present.

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

Table 4: - Showing Null Values in Dataset

- Data has “NIL” duplicate observations.

Understanding of attributes (variable info, renaming if required)

This project has 18 attributes contributing towards the target variable. Let’s discuss about these variables one after another.

1. **AccountID** – This variable represents a unique ID which represents a unique customer. This is of Integer data type and there is no null values present in this.
2. **Churn** – This is our target variable, which represents if customer has churned or not. This is categorical in nature will no null values. “0” represents “NO” and ”1” represents “YES”.
3. **Tenure** – This represents the total tenure of the account since opened. This is a continuous variable with 102 null values.
4. **City_Tier** – These variable segregates customer into 3 parts based on city the primary customer resides. This variable is categorical in nature and have 112 null values.
5. **CC_Contacted_L12m** – This variable represents the number of times all the customers of the account has contacted customer care in last 12months. This variable is continuous in nature and have 102 null values.
6. **Payment** – This variable represents the preferable mode of bill payment opted by customer. This is categorical in nature and have 109 null values.
7. **Gender** – This variable represents the gender of the primary account holder. This is categorical in nature and 108 null values.
8. **Service_Score** – Scores provided by the customer basis the service provided by the company. This variable is categorical in nature and have 98 null values.
9. **Account_user_count** – This variable gives the number of customers attached with an accountID. This is continuous in nature and have 112 null values.
10. **account_segment** – These variable segregates customers into different segment basis their spend and revenue generation. This is categorical in nature and have 97 null values.
11. **CC_Agent_Score** -- Scores provided by the customer basis the service provided by the customer care representative of the company. This variable is categorical in nature and have 116 null values.
12. **Marital_Status** – This represents marital status of the primary account holder. This is categorical in nature and have 212 null values.
13. **rev_per_month** – This represents average revenue generated per account ID in last 12 months. This variable is continuous in nature and have 102 null values.
14. **Complain_l12m** – This denotes if customer have raised any complaints in last 12 months. This is categorical in nature and have 357 null values.
15. **rev_growth_yoy** – This variable shows revenue growth in percentage of account for 12 months Vs 24 to 13 months. This is continuous in nature and doesn’t have any null values.

16. **coupon_used_l12m** – This represents the number of times customer's have used discount coupons for bill payment. This is continuous in nature and doesn't have any null values.
17. **Day_Since_CC_connect** – This represents the number of days since customer have contacted the customer care. Higher the number of days denotes better the service. This is continuous in nature and have 357 null values.
18. **cashback_l12m** – This variable represents the amount of cash back earned by the customer during bill payment. This is continuous in nature and have 471 null values.
19. **Login_device** – This variable represents in which device customer is availing the services if it's on phone or on computer. This is categorical in nature and have 221 null values.

- With above understanding of data renaming of any of the variable is not required**.**
- With the above understanding of data, we can move towards the EDA part where in we will understand the data little better along with treating bad data, null values and outliers.

Exploratory data analysis

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

Univariate Analysis: -

- The variable shows outlier in data, which needs to be treated in further steps.

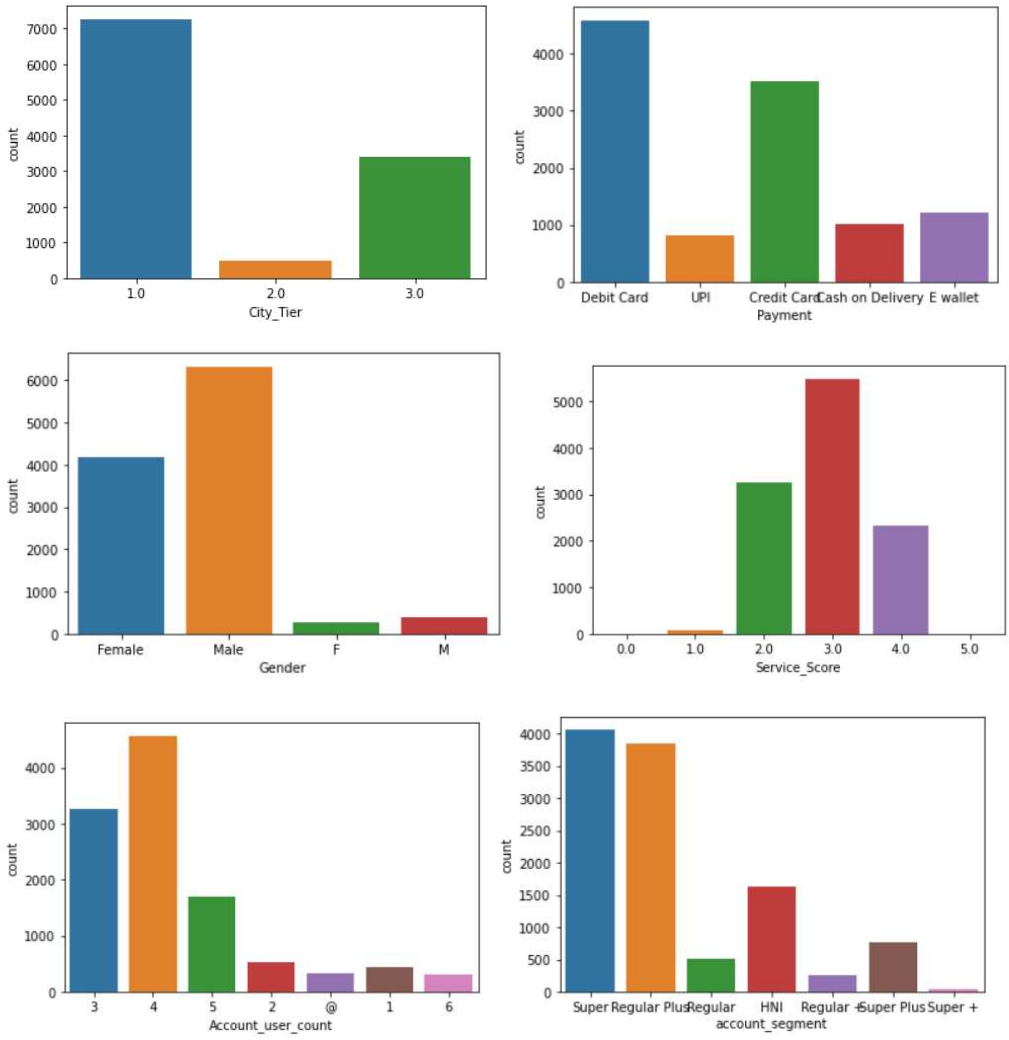
	outlier %
AccountID	0.00
Account_user_count	0.00
CC_Agent_Score	0.00
CC_Contacted_LY	0.37
Churn	16.84
City_Tier	0.00
Complain_ly	0.00
Day_Since_CC_connect	0.00
Gender	0.00
Login_device	0.00
Marital_Status	0.00
Payment	0.00
Service_Score	0.12
Tenure	0.00
account_segment	0.00
cashback	0.00
coupon_used_for_payment	0.00
rev_growth_yoy	0.00
rev_per_month	0.00

Table 5: - Showing Outliers in data

- None of the variables shows normal distribution and are skewed in nature.

	Kurtosis	Skewness
AccountID	-1.20	0.00
Churn	1.14	1.77
City_Tier	-1.40	0.74
CC_Contacted_LY	8.23	1.42
Service_Score	-0.67	0.00
CC_Agent_Score	-1.12	-0.14
Complain_ly	-1.10	0.95

Table 6:- Showing skewness and kurtosis



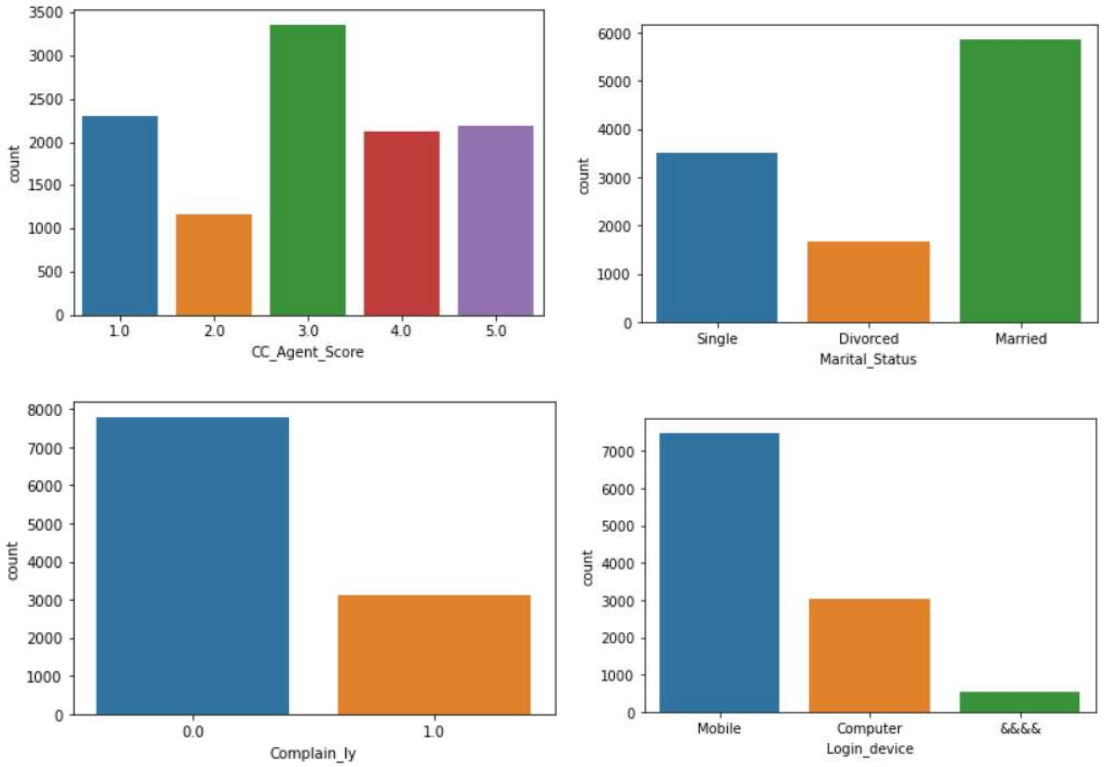
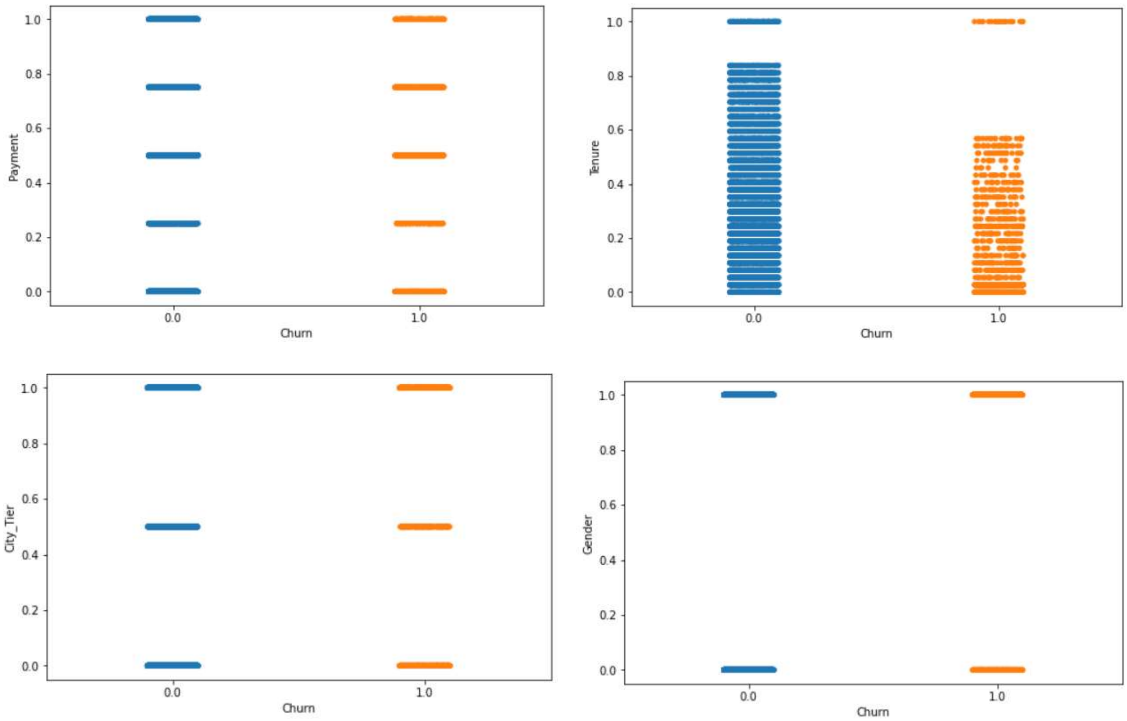


Fig 2: - Count plot of categorical variables



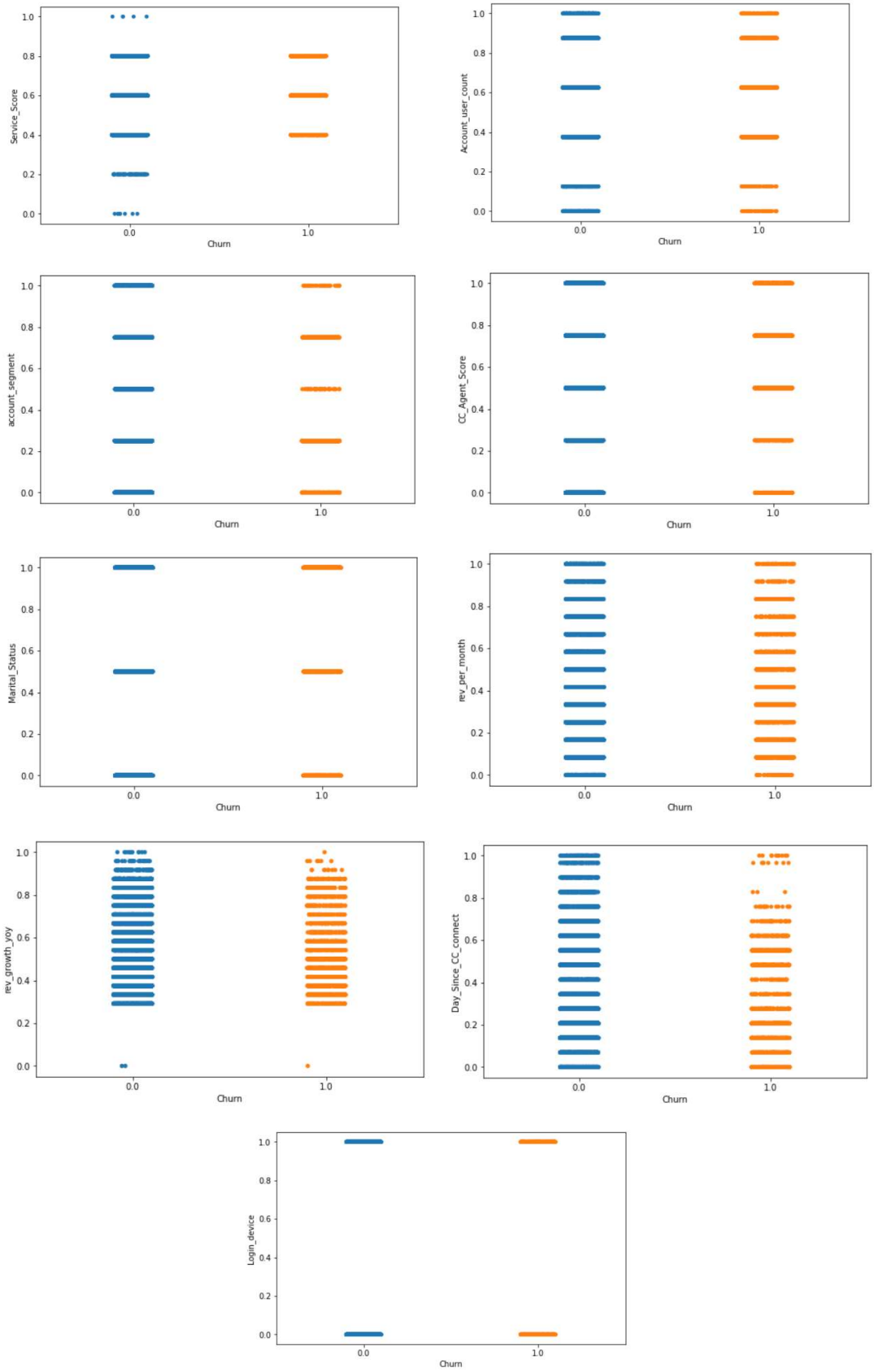


Fig 3: - Strip plot of across variables

Inferences from count plot: -

- Maximum customers are form city tire type “1”, which indicates the high number of population density in this city type.
- Maximum number of customers perfer debit and credit card as their preferred mode of payment.
- The ratio of male customers are higher when compared to female.
- Average service score given by a customer for the service provided is around “3” which shows the area of improvements.
- Most of the customers are into “Super+” segment and least number of customers are into “Regular” segment.
- Most of the customers availing servives are “Married”.
- Most of the customer perfer “Mobile” as the device to avail services.

Bi-variate Analysis: -

- Pair plot across all categorical data and its impact towards the target variable.

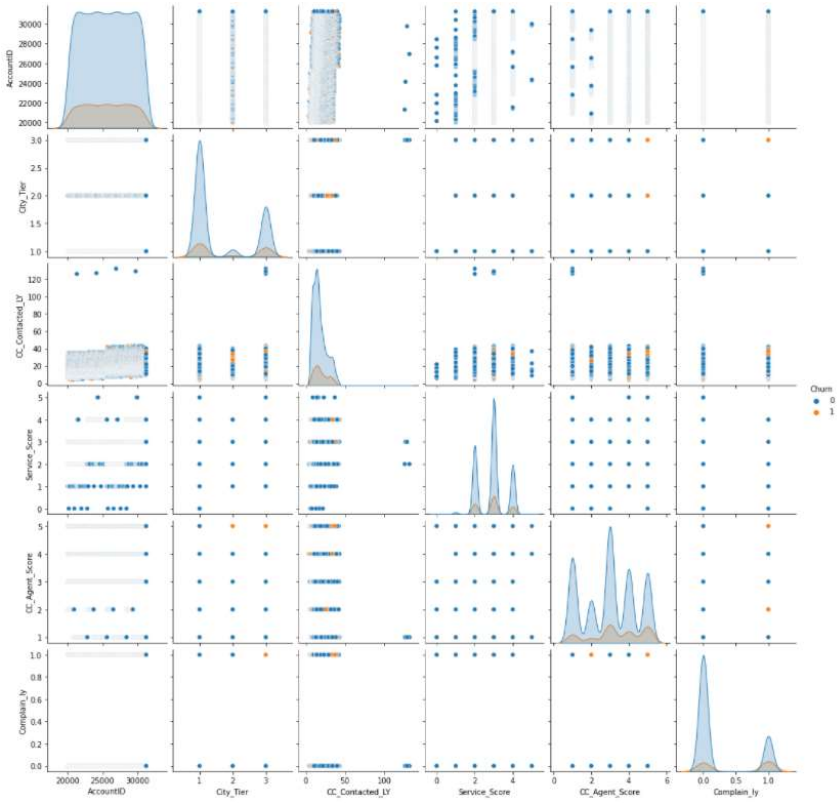
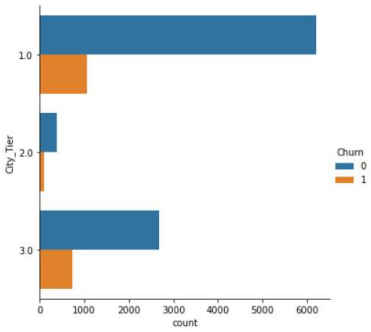
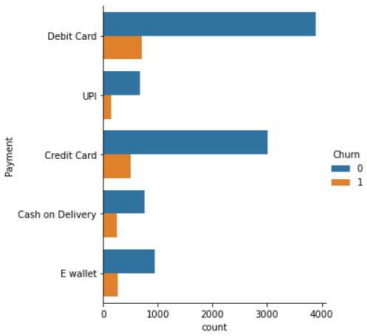
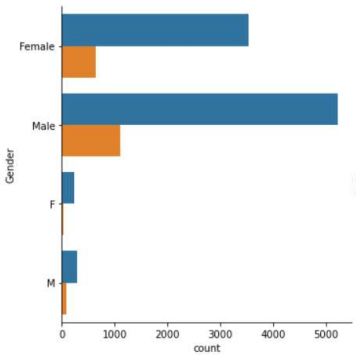


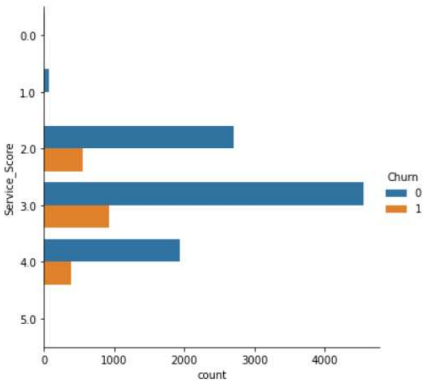
fig 4: - pairplot across categorical variables

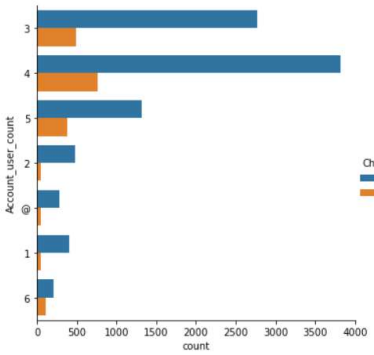
- The pair-plot shown above indiates that the independent variable are week or poor predictors of target variable as we the density of independent variable overlaps with the density of target variable.

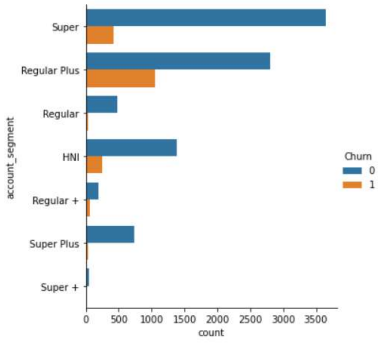

City_Tier
1.0
2.0
3.0
0
1000
2000
3000
4000
5000
6000
count
Churn
0
1

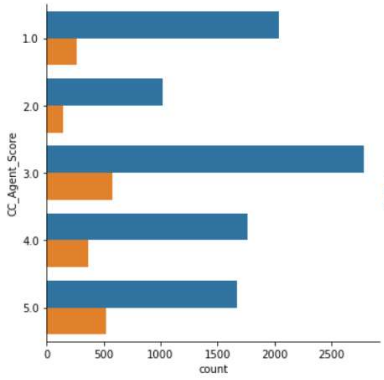

Payment
Debit Card
UPI
Credit Card
Cash on Delivery
E wallet
0
1000
2000
3000
4000
count
Churn
0
1

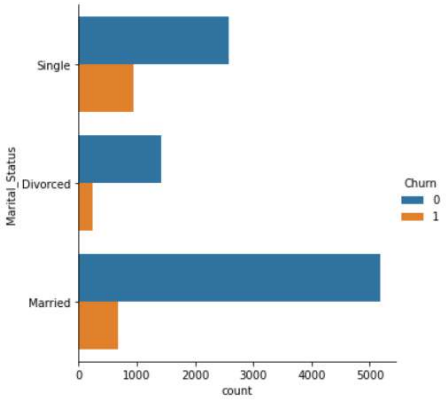

Gender
Female
Male
F
M
0
1000
2000
3000
4000
5000
count
Churn
0
1


Service_Score
0.0
1.0
2.0
3.0
4.0
5.0
0
1000
2000
3000
4000
count
Churn
0
1


Account_user_count
3
4
5
2
@
1
6
0
500
1000
1500
2000
2500
3000
3500
4000
count
Churn
0
1


account_segment
Super
Regular Plus
Regular
HNI
Regular +
Super Plus
Super +
0
500
1000
1500
2000
2500
3000
3500
count
Churn
0
1


CC_Agent_Score
1.0
2.0
3.0
4.0
5.0
0
500
1000
1500
2000
2500
count
Churn
0
1


Marital_Status
Single
Divorced
Married
0
1000
2000
3000
4000
5000
count
Churn
0
1

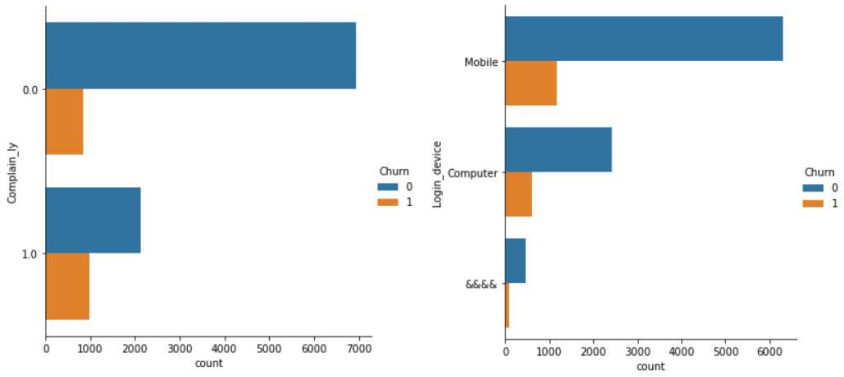


Fig 5: - Contribution of categorical variable towards chrun

- City_tier “1” has shown maximum churning when compared with “2” and “3”.
- Customer with prefreffred mode of payment as “debit card” and “credit card” are more prone to churn.
- Customers with gender as “Male” are showing more Churn ratio as compared to female.
- Customers into “Regular Plus” segment showing more churn.
- Single customers are more tend to churn when compared with divorced and married.
- Customers using the service over mobile shows more churn.

<u>Correlation among variable:-</u>

We have performed correlation between variables after treatuing bad data and missing values. We have also converted into interger data types to check on correlation as data type as categorical wont show in the pictures below.

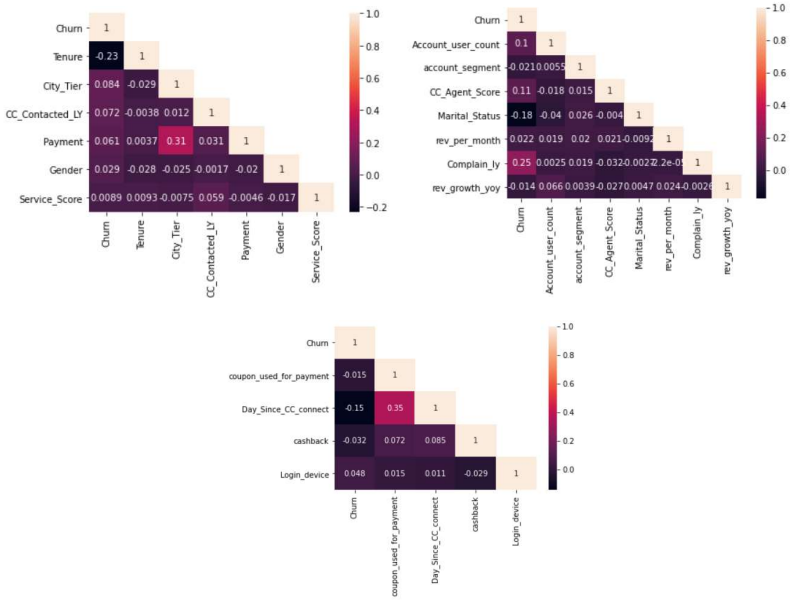


Fig 6: - Correlation among variables

Inferences from correlation: -

- Variable “Tenure” shows high co-relation with Churn.
- Variable “Marital Status” shows high co-relation with churn.
- Variable “complain_ly” shows high- correlation with churn.

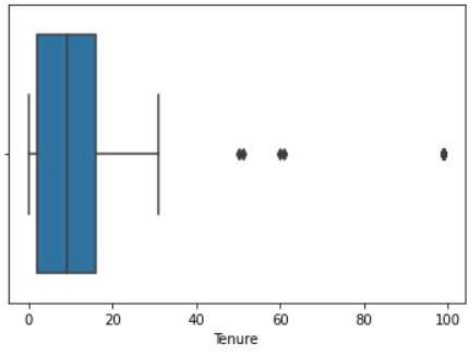
Removal of unwanted variables: - After in-depth understanding of data we conclude that removal of variables is not required at this stage of project. We can remove the variable “AccountID” which denotes a unique ID assigned to unique customers. However, removing them will lead to 8 duplicate rows. Rest all the variables looks important looking at the univariate and bi-variate analysis.

Outlier treatment: -

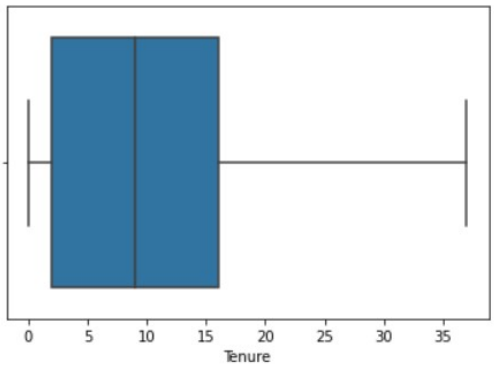
This dataset is the mix of continuous as well as categorical variables. It doesn’t make nay sense if we perform outlier treatment on categorical variable as each category denotes a type of customer. So, we are performing outlier treatment only for variables continuous in nature.

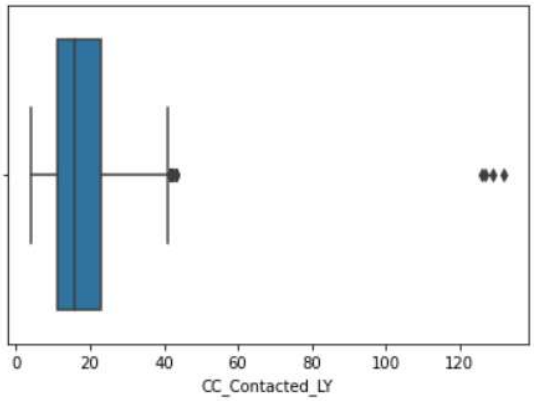
- Used box plot to determine the presence if outlier in a variable.
- The dots outside the upper limit of a quantile represents the outlier in the variable.
- We have 8 continuous variables in the dataset namely, “Tenure”, “CC_Contacted_LY”, “Account_user_count”, “cashback”, “rev_per_month”, “Day_Since_CC_connect”, “coupon_used_for_payment” and “rev_growth_yoy”.
- We have used upper limit and lower limit to remove outliers. Below is the pictorial representation of variables before and after outlier treatment.

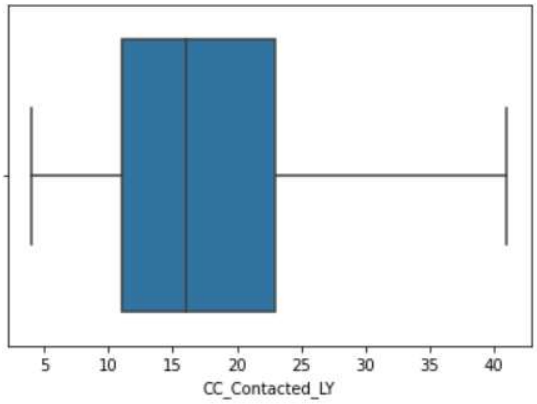
Before

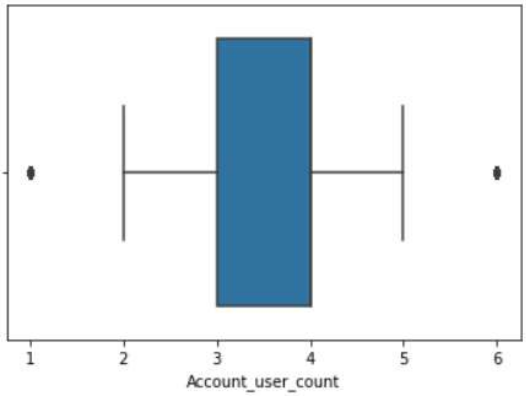


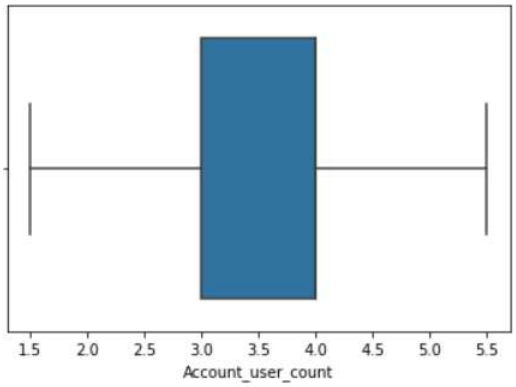
After

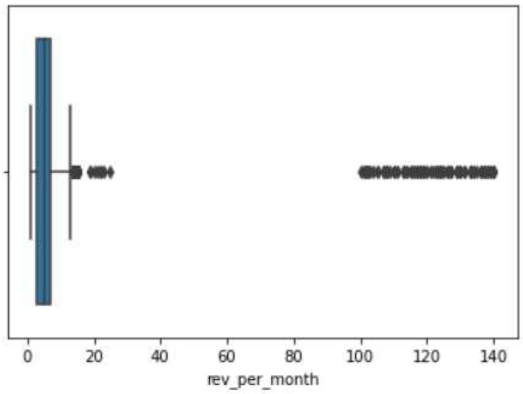


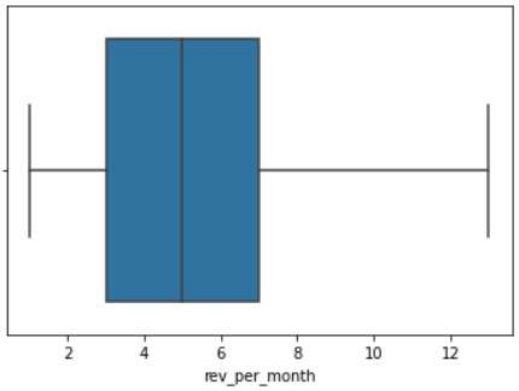

0
20
40
60
80
100
120
CC_Contacted_LY

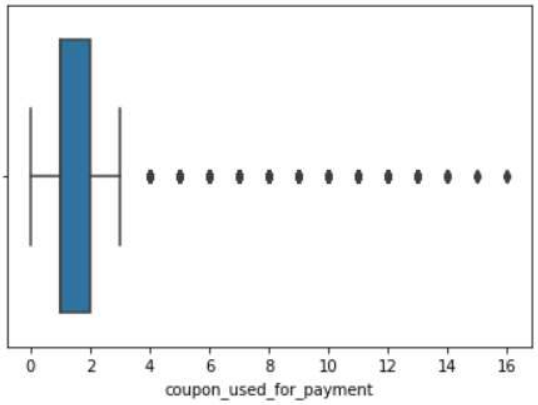

5
10
15
20
25
30
35
40
CC_Contacted_LY

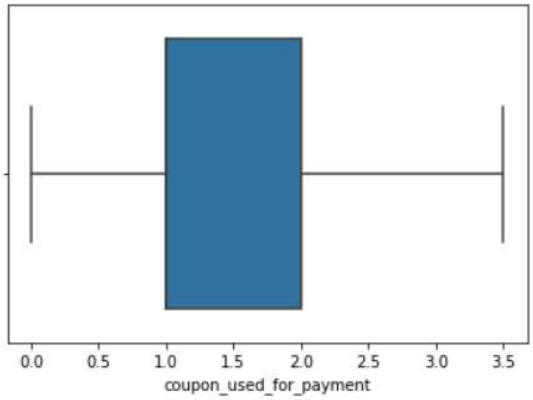

1
2
3
4
5
6
Account_user_count


1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0
5.5
Account_user_count


0
20
40
60
80
100
120
140
rev_per_month


2
4
6
8
10
12
rev_per_month


0
2
4
6
8
10
12
14
16
coupon_used_for_payment


0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
coupon_used_for_payment

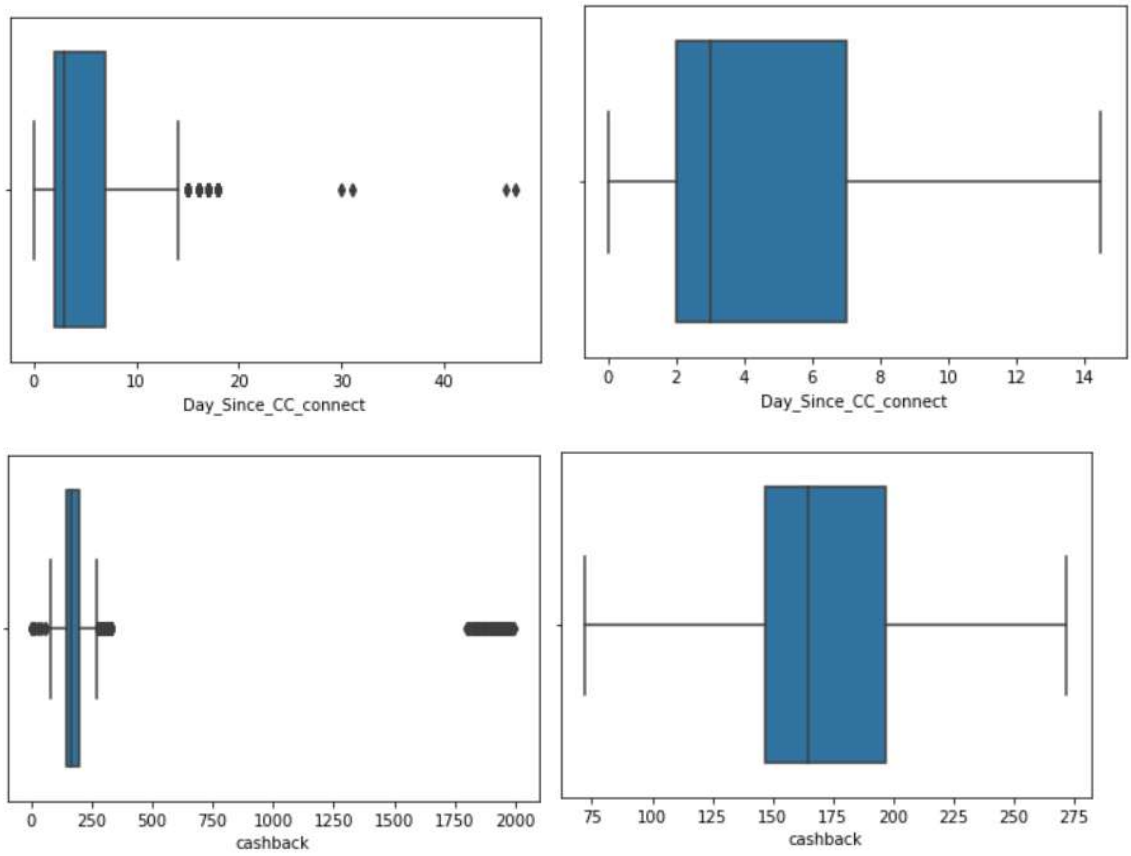


Fig 7: - Before and after outlier treatement

Missing Value treatment and variable transformation: -

- Out of 19 variables we have data anomalies present in 17 variable and null values in 15 variables.
- Using “Median” to impute null values where variable is continuous in nature because Median is less prone to outliers when compared with mean.
- Using “Mode: to impute null values where variables are categorical in nature.
- We have treated null values variable by variable as each and every variable is unique in its nature.

Treating Variable “Tenure”

- We look at the unique observations in the variable and see that we have “#” and “nan” present in the data. Where “#” is a anomaly and “nan” represents null value.

```
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)
```

Fig 8: - before treatment

- Replacing “#” with “nan” and further we replace “nan” with calculated median of the variable and now we don’t see any presence of bad data and null values.

- Converted data type to integer, because IDE has recognized it as object data type due presence of bad data.

```
<IntegerArray>
[ 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, 27, 12, 21, 17, 50, 60, 31, 51, 61]
Length: 37, dtype: Int64
```

Fig 9: - after treatment

Treating Variable "City_Tier"

- We look at the unique observations in the variable and see presence of null value as shown below.

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

Fig 10: - before treatment

- we are replacing "nan" with calculated mode of the variable and now we don't see any presence of null values.
- Converted data type to integer, because IDE has recognized it as object data type due presence of bad data.

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

Fig 11: - after treatment

Treating Variable "CC_Contacted_LY"

- We look at the unique observations in the variable and see presence of null value as shown below.

```
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.])
```

Fig 12: - before treatment

- we are replacing "nan" with calculated Median of the variable and now we don't see any presence of null values.
- Converted data type to integer, because IDE has recognized it as object data type due presence of bad data.

```
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, 34, 5, 4, 41.0, 7, 36,
       38, 37, 39, 40], dtype=object)
```

Fig 13: - after treatment

Treating Variable "Payment"

- We look at the unique observations in the variable and see presence of null value as shown below.

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

Fig 14: - before treatment

- we are replacing "nan" with calculated Mode of the variable and now we don't see any presence of null values.
- Also performed label encoding for the observations. Where 1 = Debit card, 2 = UPI, 3 = credit card, 4 = cash on delivery and 5 = e-wallet. Then converting them to integer data type as it will be used for further model building.

```
array(['1', '2', '3', '4', '5']
```

Fig 15: - after treatment

Treating Variable "Gender"

- We look at the unique observations in the variable and see presence of null value and multiple abbreviations of the same observations as shown below.

```
array(['Female', 'Male', 'F', nan, 'M']
```

Fig 16: - before treatment

- we are replacing "nan" with calculated Mode of the variable and now we don't see any presence of null values.
- Also performed label encoding for the observations. Where 1 = Female card and 2 = Male. Then converting them to integer data type as it will be used for further model building.

```
array(['1', '2'],
```

Fig 17: - after treatment

Treating Variable "Service_Score"

- We look at the unique observations in the variable and see presence of null value as shown below.

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

Fig 18: - before treatment

- we are replacing "nan" with calculated Mode of the variable and now we don't see any presence of null values.
- Then converting them to integer data type as it will be used for further model building.

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

Fig 19: - after treatment

Treating Variable "Account_user_count"

- We look at the unique observations in the variable and see presence of null value as well "@" as bad data, shown below.

```
array([3, 4, nan, 5, 2, '@', 1, 6]
```

Fig 20: - before treatment

- Replacing "@" with "nan" and further we replace "nan" with calculated median of the variable and now we don't see any presence of bad data and null values.
- Then converting them to integer data type as it will be used for further model building.

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

Fig 21: - after treatment

Treating Variable "account_segment"

- We look at the unique observations in the variable and see presence of null value as well different denotations for the same type of observations, shown below.

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

Fig 22: - before treatment

- Replacing "nan" with calculated Mode of the variable and also labelled different account segments, where in 1 = Super, 2 = Regular Plus, 3 = Regular, 4 = HNI and 5 = Super Plus and now we don't see any presence of bad data and null values.
- Then converting them to integer data type as it will be used for further model building.

```
array(['1', '2', '3', '4', '5']
```

Fig 23: - after treatment

Treating Variable "CC_Agent_Score"

- We look at the unique observations in the variable and see presence of null value as shown below.

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

Fig 24: - before treatment

- Replacing "nan" with calculated Mode of the variable and now we don't see any presence of bad data and null values.
- Then converting them to integer data type as it will be used for further model building.

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

Fig 25: - after treatment

Treating Variable "Marital_Status"

- We look at the unique observations in the variable and see presence of null value as shown below.

```
array(['Single', 'Divorced', 'Married', nan],
```

Fig 26: - before treatment

- Replacing "nan" with calculated Mode of the variable and also labelled the observations. Where in 1 = Single, 2 = Divorced and 3 = Married and now we don't see any presence of bad data and null values.
- Then converting them to integer data type as it will be used for further model building.

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

Fig 27: - after treatment

Treating Variable "rev_per_month"

- We look at the unique observations in the variable and see presence of null value as well as presence of "+" which denoted bad data. shown below.

```
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)
```

Fig 28: - before treatment

- Replacing "+" with "nan" and further we replace "nan" with calculated median of the variable and now we don't see any presence of bad data and null values.
- Then converting them to integer data type as it will be used for further model building.

```
array([  9.,   7.,   6.,   8.,   3.,   2.,   4.,  10.,   1.,   5., 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.])
```

Fig 29: - after treatment

Treating Variable "Complain_ly"

- We look at the unique observations in the variable and see presence of null value as shown below.

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

Fig 30: - before treatment

- Replacing "nan" with calculated Mode of the variable and now we don't see any presence of null values.
- Then converting them to integer data type as it will be used for further model building.

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

Fig 31: - after treatment

Treating Variable "rev_growth_yoy"

- We look at the unique observations in the variable and see presence of "$" which denoted bad data. shown below.

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

Fig 32: - before treatment

- Replacing "$" with "nan" and further we replace "nan" with calculated median of the variable and now we don't see any presence of bad data and null values.
- Then converting them to integer data type as it will be used for further model building.

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

Fig 33: - after treatment

Treating Variable "coupon_used_for_payment"

- We look at the unique observations in the variable and see presence of "$", "*" and "#" which denoted bad data. shown below.

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

Fig 34: - before treatment

- Replacing "$", "*" and "#" with "nan" and further we replace "nan" with calculated median of the variable and now we don't see any presence of bad data and null values.
- Then converting them to integer data type as it will be used for further model building.

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

Fig 35: - after treatment

Treating Variable "Day_Since_CC_connect"

- We look at the unique observations in the variable and see presence of "$" which denoted bad data and also the presence of null values. shown below.

```
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)
```

Fig 36: - before treatment

- Replacing "$" with "nan" and further we replace "nan" with calculated median of the variable and now we don't see any presence of bad data and null values.
- Then converting them to integer data type as it will be used for further model building.

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

Fig 37: - after treatment

Treating Variable "cashback"

- We look at the unique observations in the variable and see presence of "$" which denoted bad data and also the presence of null values. shown below.

```
array([159.93, 120.9, nan, ..., 227.36, '$', .91, 191.42]
```

Fig 38: - before treatment

- Replacing "$" with "nan" and further we replace "nan" with calculated median of the variable and now we don't see any presence of bad data and null values.
- Then converting them to integer data type as it will be used for further model building.

```
array([159., 120., 165., 134., 129., 139., 122., 126., 272., 153., 133.,
       196., 157., 160., 149., 161., 203., 116., 206., 142., 172., 123.,
       189., 143., 208., 127., 194., 125., 124., 186., 130., 150., 111.,
       204., 131., 144., 195., 237., 267., 135., 152., 162., 168., 138.,
       166., 176., 121., 148., 193., 184., 199., 224., 235., 188., 221.,
        72., 179., 187., 132., 260., 137., 236., 164., 200., 209., 169.,
       268., 155., 140., 234., 218., 219., 156., 163., 145., 154., 147.,
       158., 114., 180., 136., 112., 220., 270., 175., 146., 174., 215.,
       171., 182., 259., 225., 167., 128., 266., 141., 243., 183., 265.,
       117., 241., 202., 190., 198., 232., 261., 118., 205., 254., 177.,
       110., 211., 248., 217., 178., 151., 216., 271., 263., 207., 238.,
       242., 197., 231., 239., 227., 233., 173., 119., 170., 185., 240.,
       247., 192., 113., 264., 115., 212., 201., 252., 229., 181., 257.,
       210., 269., 228., 214., 244., 253., 262., 191., 249., 213., 245.,
       250., 223., 230., 222., 256., 258., 246., 226.,  81., 251.])
```

Fig 39: - after treatment

Treating Variable "Login_device"

- We look at the unique observations in the variable and see presence of "&&&&" which denoted bad data and also the presence of null values. shown below.

```
array(['Mobile', 'Computer', '&&&&', nan
```

Fig 40: - before treatment

- Replacing "&&&&" with "nan" and further we replace "nan" with calculated Mode of the variable. Also, labelling the observations where in 1= Mobile and 2 = Computer and now we don't see any presence of bad data and null values.
- Then converting them to integer data type as it will be used for further model building.

```
array([1, 2]
```

Fig 41: - after treatment

Count of null values before and after treatment

Before		***After***	
AccountID	0	AccountID	0
Churn	0	Churn	0
Tenure	102	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	112	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	102	rev_per_month	0
Complain_ly	357	Complain_ly	0
rev_growth_yoy	0	rev_growth_yoy	0
coupon_used_for_payment	0	coupon_used_for_payment	0
Day_Since_CC_connect	357	Day_Since_CC_connect	0
cashback	471	cashback	0
Login_device	221	Login_device	0

Fig 42: - Before and after null value treatment

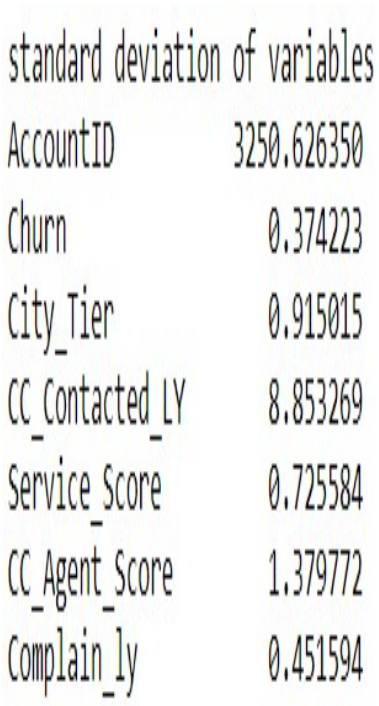
- We see NIL null values across variable which indicated that the data is now cleaned and we can move further for data transformation of required.

Variable transformation: -

- We see that the different variable have different dimensions. Like variable "Cashback" denotes currency where as "CC_Agent_Score" denotes rating provided by the customers. Due to which they differ in their statistical rating as well.
- Scaling would be required for this data set which in turn will normalize the date and standard deviation will be close to "0".
- Using MinMax scalar to perform normalization of data.

Standard Deviation Before and After Normalization: -

Before



standard deviation of variables	
AccountID	3250.626350
Churn	0.374223
City_Tier	0.915015
CC_Contacted_LY	8.853269
Service_Score	0.725584
CC_Agent_Score	1.379772
Complain_ly	0.451594

After

Churn	0.374223
Tenure	0.240241
City_Tier	0.456381
CC_Contacted_LY	0.231463
Payment	0.344845
Gender	0.488878
Service_Score	0.144495
Account_user_count	0.231069
account_segment	0.316751
CC_Agent_Score	0.343166
Marital_Status	0.447373
rev_per_month	0.239968
Complain_ly	0.447181
rev_growth_yoy	0.156553
coupon_used_for_payment	0.314928
Day_Since_CC_connect	0.240931
cashback	0.218847
Login_device	0.442952

Fig 43: - Before and after Normalization

- We see that the standard deviation of variables are now close to "0".
- Also converted variables to int data type which will help in further model building process.

Addition of new variables: -

At the current stage we don't see to create ay new variable as such. May be required at further stage of model building and can be created accordingly.

Business insights from EDA

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

- Dataset provided is imbalance in nature. The categorical count of our target variable "Churn" shows high variation in counts. We have count of "0" as 9364 and count of "1" as 1896.

Churn	AccountID
0	9364
1	1896

Table 44: - Imbalanced dataset

- This imbalance in dataset can be performed using SMOTE technique will generates additional datapoints to balance the data.
- We need to apply SMOTE only on to train dataset not on test dataset. divided data into train and test dataset in 70:30 ratio as a accepted market practice (can be changed later as instructed).

Before SMOTE

```
X_train (7882, 17)
X_test (3378, 17)
y_train (7882,)
y_test (3378,)
```

After SMOTE

```
X_train_res (13112, 17)
y_train_res (13112,)
```

Table 8: - Before and after SMOTE

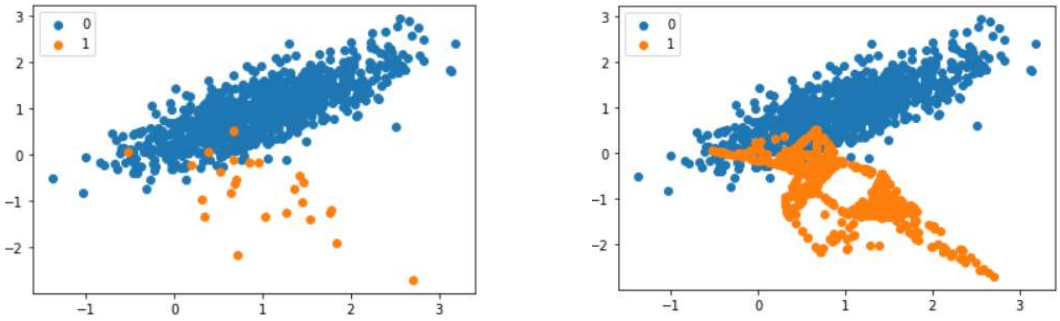


Fig 45: - Before and after SMOTE

- The increase in density of the orange dots indicates the increase in data points.

Any business insights using clustering

- Created 3 clusters using K-means cluster and segmenting customers into these 3 segments.
- Decided as 3 number of clusters based upon the inertia value.
- Maximum counts of customers are in 2nd cluster and least in 3rd cluster.

Any other business insights

- We see decent variations in data collection with a mixture of services provided along with rating provided by the customer and also about customer profile.
- Business needs to increase its visibility in tier 2 city and can acquire new customers.
- Business can promote payment via standing instruction in bank account or UPI which can be hassle free and safe for customers.
- There is need of improvement in service scores and have a lot of grey area left over. Business and roll out a survey for better understanding of customer's expectations.
- Business can train their customer care executive to provide better customer experience which in turn will improve their feedback scores.
- Can have curated plans for customers not only based on the spend they have but also the tenure they have spent with the business.
- Can have curated plan for married people something like a family floater.

Appendix

Codes Snapshot: -

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

```
# loding data
churn  = pd.read_excel(r'C:\Users\abhay\Downloads\Customer Churn Data.xlsx', sheet_name = 'Data for DSBA')
```

```
# checking info of data
churn.info()
```

```
# checking shape of dataset
print("The shape of dataset is :{}".format(churn.shape))
```

```
# checking for duplicate values
print("Number of duplicate rows:",churn.duplicated().sum())
```

```
#checking for skewness
churn.hist(figsize=(20,15));
```

```
# kest check kurtosis and skewness of data
print("kurtosis and skewness of dataste is as below")
pd.DataFrame(data = [churn.kurtosis(), churn.skew()], index=['Kurtosis','Skewness']).T.round(2)
```

```
# plotting sns plot
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
# pair plot to check on data distribution and co-linearity
sns.pairplot(churn, hue = 'Churn', diag_kind='kde')
plt.show()
```

```
#checking if the data is balanced or not
churn.groupby(["Churn"]).count()
```

```
sns.countplot(churn["City_Tier"]);
```

```
sns.countplot(churn["Payment"]);
```

```
sns.countplot(churn["Gender"]);
```

```
sns.countplot(churn["Service_Score"]);
```

```
sns.countplot(churn["Account_user_count"]);
```

```
sns.countplot(churn["account_segment"]);
```

```
sns.countplot(churn["CC_Agent_Score"]);
```

```
sns.countplot(churn["Marital_Status"]);
```

```
sns.countplot(churn["Complain_ly"]);
```

```
sns.countplot(churn["Login_device"]);
```

```
sns.catplot(y="City_Tier", hue="Churn", kind="count", data=churn)
```

```
sns.catplot(y="Payment", hue="Churn", kind="count", data=churn)
```

```
sns.catplot(y="Gender", hue="Churn", kind="count", data=churn)
```

```
sns.catplot(y="Service_Score", hue="Churn", kind="count", data=churn)
```

```
sns.catplot(y="Account_user_count", hue="Churn", kind="count", data=churn)
```

```
sns.catplot(y="account_segment", hue="Churn", kind="count", data=churn)
```

```
sns.catplot(y="CC_Agent_Score", hue="Churn", kind="count", data=churn)
```

```
sns.catplot(y="Marital_Status", hue="Churn", kind="count", data=churn)
```

```
sns.catplot(y="Complain_ly", hue="Churn", kind="count", data=churn)
```

```
sns.catplot(y="Login_device", hue="Churn", kind="count", data=churn)
```

```
# plotting heatmap of correlation
cor = churn.corr()
mask = np.array(cor)
mask[np.tril_indices_from(mask)] = False
fig,ax= plt.subplots()
fig.set_size_inches(10,10)
sns.heatmap(cor, mask=mask, vmax=1, square=True, annot=True)
plt.show()
```

```
churn['account_segment'] = churn['account_segment'].replace('Super','1')
churn['account_segment'] = churn['account_segment'].replace('Regular Plus','2')
churn['account_segment'] = churn['account_segment'].replace('Regular +','2')
churn['account_segment'] = churn['account_segment'].replace('Regular','3')
churn['account_segment'] = churn['account_segment'].replace('HNI','4')
churn['account_segment'] = churn['account_segment'].replace('Super Plus','5')
churn['account_segment'] = churn['account_segment'].replace('Super +','5')
```

```
# lets check the percentage of outlier in each column
Q1 = churn.quantile(0.25)
Q3 = churn.quantile(0.75)
IQR = Q3 - Q1
pd.DataFrame((((churn < (Q1 - 1.5 * IQR)) | (churn > (Q3 + 1.5 * IQR))).sum()/churn.shape[0]*100),
             columns = ['outlier %'], index = None). round(2)
```

```
# kest check kurtosis and skewness of data
print("kurtosis and skewness of dataste is as below")
pd.DataFrame(data = [churn.kurtosis(), churn.skew()], index=['Kurtosis','Skewness']).T.round(2)
```

```
# lets check the percentage of outlier in each column
Q1 = churn.quantile(0.25)
Q3 = churn.quantile(0.75)
IQR = Q3 - Q1
pd.DataFrame((((churn < (Q1 - 1.5 * IQR)) | (churn > (Q3 + 1.5 * IQR))).sum()/churn.shape[0]*100),
             columns = ['outlier %'], index = None). round(2)
```

```
from sklearn.preprocessing import MinMaxScaler

churn['Scaled_Churn'] = MinMaxScaler().fit_transform(churn[['Churn']])
churn['Scaled_Tenure'] = MinMaxScaler().fit_transform(churn[['Tenure']])
churn['Scaled_City_Tier'] = MinMaxScaler().fit_transform(churn[['City_Tier']])
churn['Scaled_CC_Contacted_LY'] = MinMaxScaler().fit_transform(churn[['CC_Contacted_LY']])
churn['Scaled_Payment'] = MinMaxScaler().fit_transform(churn[['Payment']])
churn['Scaled_Gender'] = MinMaxScaler().fit_transform(churn[['Gender']])
churn['Scaled_Service_Score'] = MinMaxScaler().fit_transform(churn[['Service_Score']])
churn['Scaled_Account_user_count'] = MinMaxScaler().fit_transform(churn[['Account_user_count']])
churn['Scaled_account_segment'] = MinMaxScaler().fit_transform(churn[['account_segment']])
churn['Scaled_CC_Agent_Score'] = MinMaxScaler().fit_transform(churn[['CC_Agent_Score']])
churn['Scaled_Marital_Status'] = MinMaxScaler().fit_transform(churn[['Marital_Status']])
churn['Scaled_rev_per_month'] = MinMaxScaler().fit_transform(churn[['rev_per_month']])
churn['Scaled_Complain_ly'] = MinMaxScaler().fit_transform(churn[['Complain_ly']])
churn['Scaled_rev_growth_yoy'] = MinMaxScaler().fit_transform(churn[['rev_growth_yoy']])
churn['Scaled_coupon_used_for_payment'] = MinMaxScaler().fit_transform(churn[['coupon_used_for_payment']])
churn['Scaled_Day_Since_CC_connect'] = MinMaxScaler().fit_transform(churn[['Day_Since_CC_connect']])
churn['Scaled_cashback'] = MinMaxScaler().fit_transform(churn[['cashback']])
churn['Scaled_Login_device'] = MinMaxScaler().fit_transform(churn[['Login_device']])
```

```
churn_scaled = pd.DataFrame({
'Churn': churn['Scaled_Churn'] ,
'Tenure': churn['Scaled_Tenure'] ,
'City_Tier': churn['Scaled_City_Tier'] ,
'CC_Contacted_LY': churn['Scaled_CC_Contacted_LY'] ,
'Payment': churn['Scaled_Payment'] ,
'Gender': churn['Scaled_Gender'] ,
'Service_Score': churn['Scaled_Service_Score'] ,
'Account_user_count': churn['Scaled_Account_user_count'] ,
'account_segment': churn['Scaled_account_segment'] ,
'CC_Agent_Score': churn['Scaled_CC_Agent_Score'] ,
'Marital_Status': churn['Scaled_Marital_Status'] ,
'rev_per_month': churn['Scaled_rev_per_month'] ,
'Complain_ly': churn['Scaled_Complain_ly'] ,
'rev_growth_yoy': churn['Scaled_rev_growth_yoy'] ,
'coupon_used_for_payment': churn['Scaled_coupon_used_for_payment'] ,
'Day_Since_CC_connect': churn['Scaled_Day_Since_CC_connect'] ,
'cashback': churn['Scaled_cashback'] ,
'Login_device': churn['Scaled_Login_device'] })
churn_scaled
```

```
# plotting sns plot
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
# pair plot to check on data distribution and co-linearity
sns.pairplot(churn_scaled, hue = 'Churn', diag_kind='kde')
plt.show()
```

```
from numpy import where
import matplotlib.pyplot as plt

from collections import Counter
from sklearn.datasets import make_classification
from imblearn.over_sampling import SMOTE
```

```
X, y = make_classification(n_samples=1000, n_features=2, n_redundant=0, n_clusters_per_class=1, weights=[0.975], flip_y=0,
counter=Counter(y)
counter
```

```
from collections import Counter
for label, _ in counter.items():
    row_ix = where(y == label)[0]
    plt.scatter(X[row_ix, 0], X[row_ix, 1], label=str(label))
plt.legend()
plt.show()
```

End of Project Note - 1