March 2022 | By: Athisya Nadar

# Capstone Project notes

PGP in Data Science and Business Analytics

# Table of Contents

# Business Objective

## Customer Churn

An E Commerce company or DTH (you can choose either of these two domains) 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.

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

### 1) Introduction of the business problem

### a) Defining problem statement

Customer churn and engagement has become one of the top issues for most E Commerce company. It costs significantly more to acquire new customers than retain existing. It is of utmost important for a company to retain its customers.

We have data from an ecommerce company of 11260 customers. In this data-set we have a dependent variable “Churn” which works out to be 16.83% and various independent variables. Based on the data, we have to build a model to predict whether the customer will exit the company.

### b) Need of the study/project

Customer churn rate is a big concern for every company. Retaining existing customers by offering lucrative deal is cheaper than acquiring new customers. Since 1 account has more than 1 customer, our goal is to identify which variables have biggest impact on churn rate.

### c) Understanding business/social opportunity

To entice the customers, the goal of the business is to identify the features of a customer who may churn like previous interactions with the company, spend analysis, rewards and other patterns so that we create campaigns to cross sell, value add, loyalty program to help reduce the churn rate.

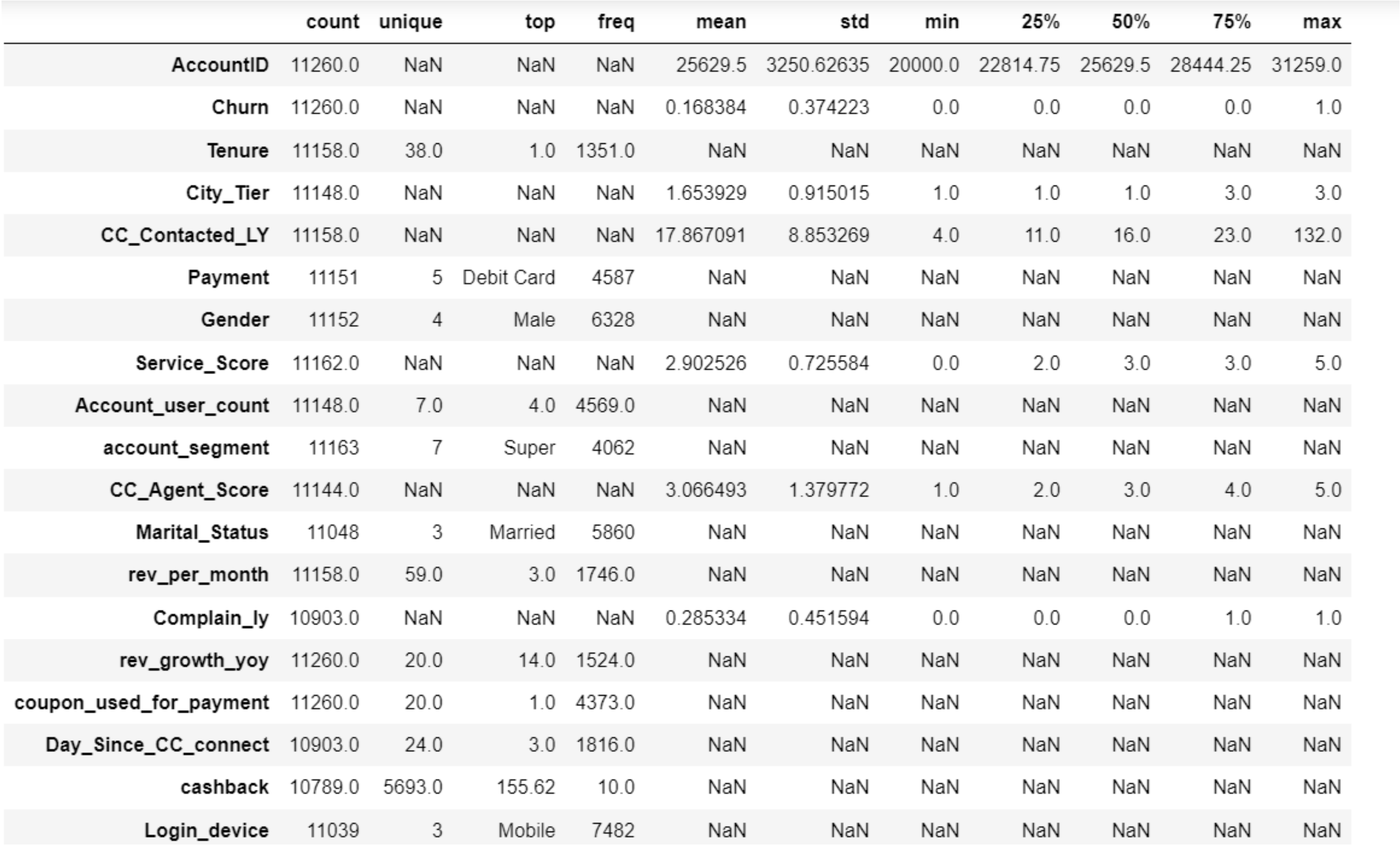
### 2. Data Report

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

Data is collected based on account unique identifier where we have data about Tenure, Tier of primary customer's city, How many times all the customers of the account has contacted customer care in last 12months, Preferred Payment mode of the customers in the account, Gender of the primary customer of the account, Satisfaction score given by customers of the account on service provided by company, Number of customers tagged with this account, Account segmentation on the basis of spend, Satisfaction score given by customers of the account on customer care service provided by company, Marital status of the primary customer of the account, Monthly average revenue generated by account in last 12 months, Any complaints has been raised by account in last 12 months, revenue growth percentage of the account (last 12 months vs last 24 to 13 month),How many times customers have used coupons to do the payment in last 12 months, Number of days since no customers in the account has contacted the customer care, Monthly average cashback generated by account in last 12 months, Preferred login device of the customers in the account

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

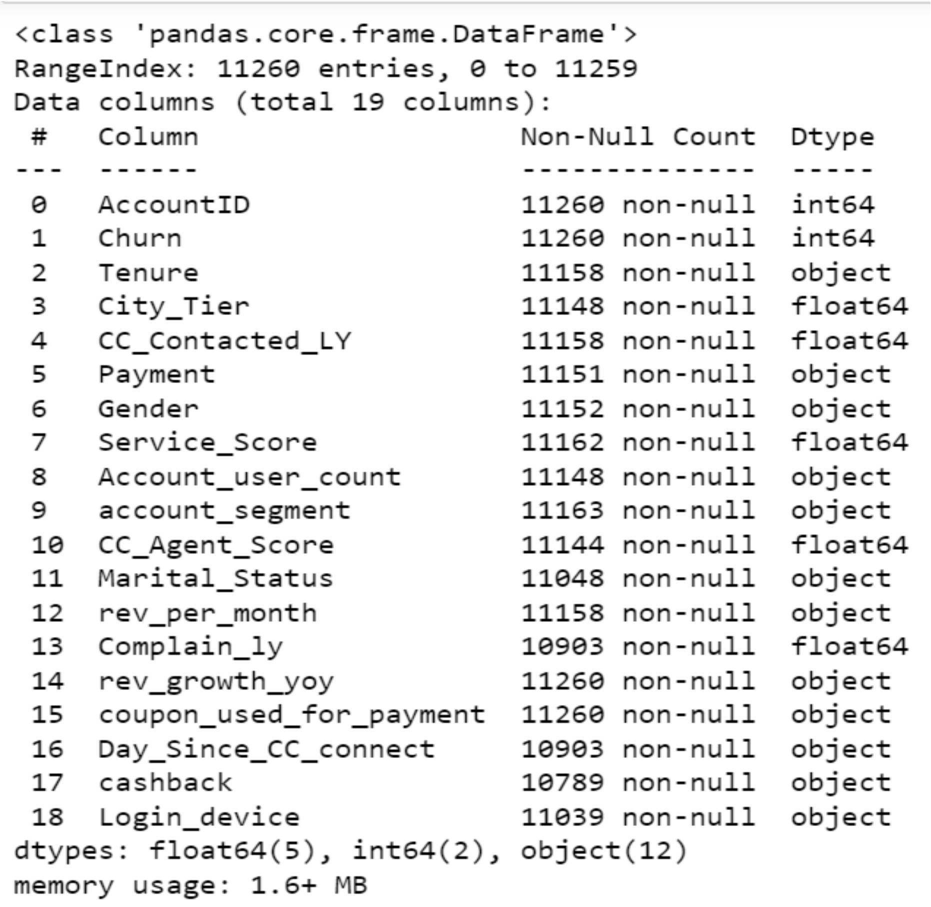
**The data consists of 11260 rows and 19 columns**



| | 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 |

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

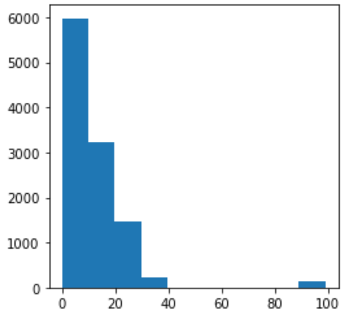
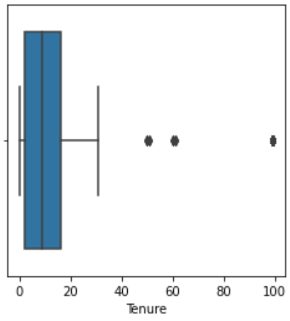
we have not renamed any column, but we can clearly see missing values in few columns and the data types of variables include 5 float64 variables, 2 int64 variables, 12 object variables.

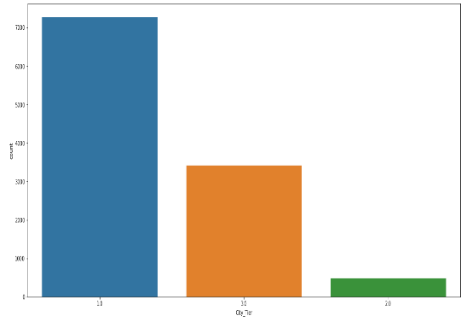
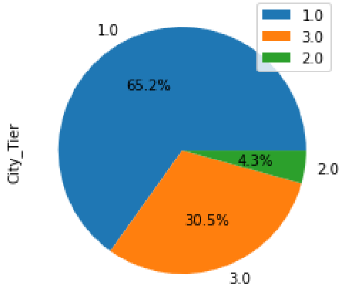
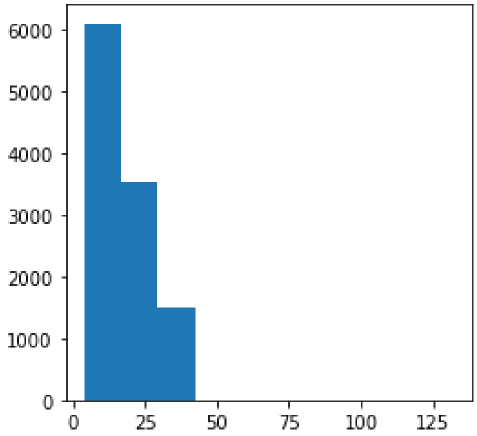
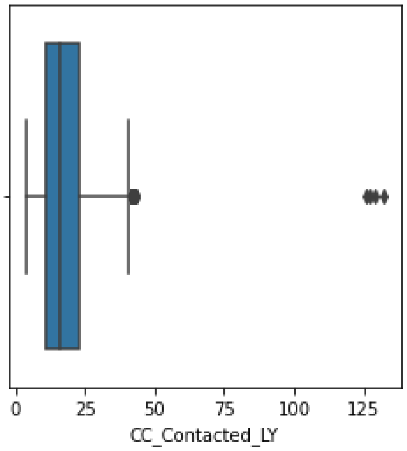
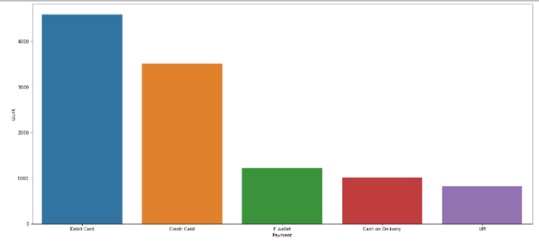
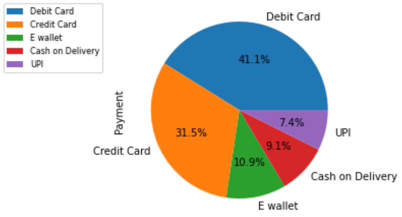
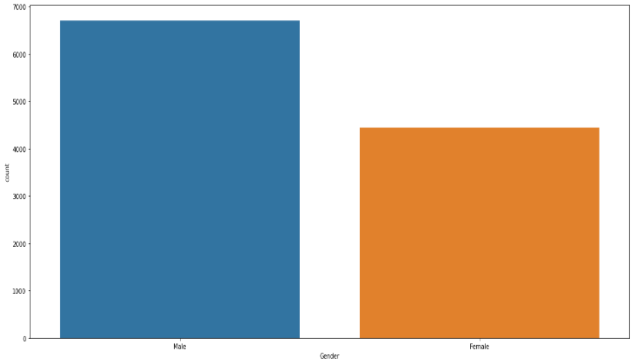
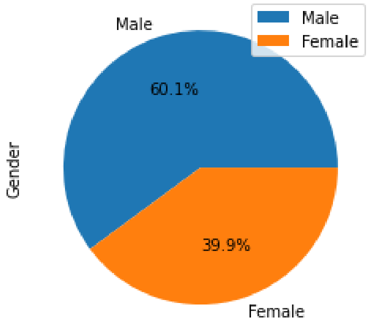


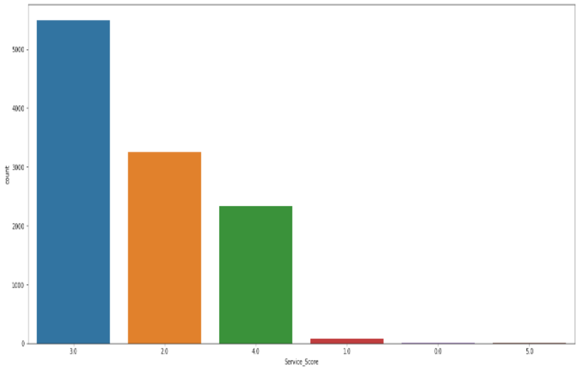
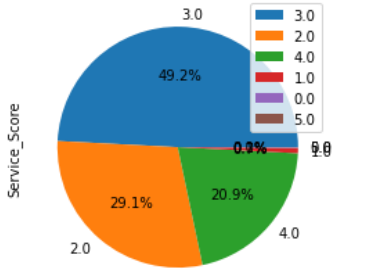
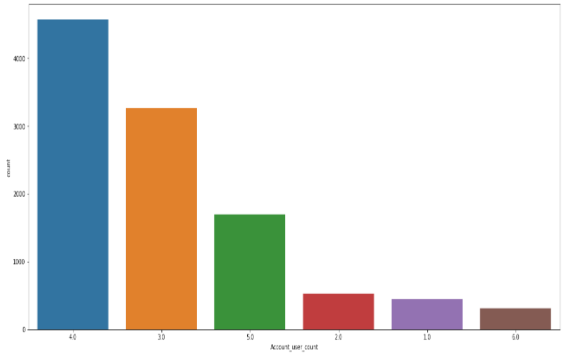
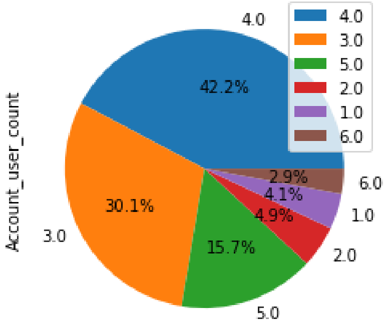
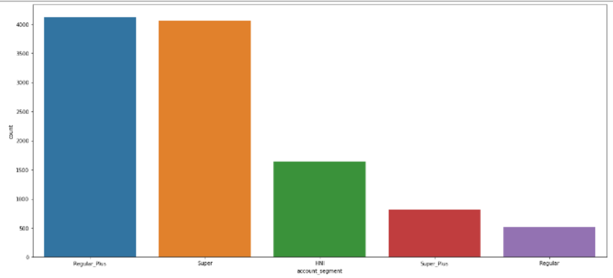
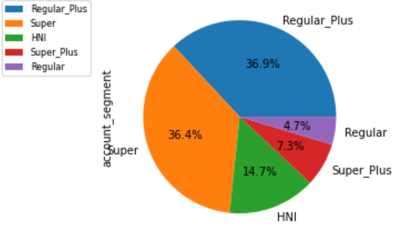
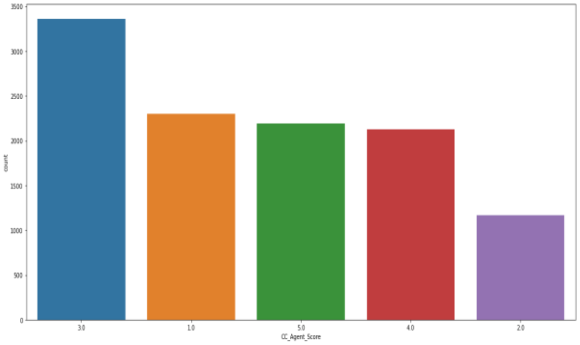
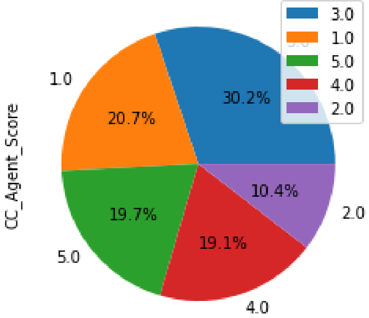
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 11260 entries, 0 to 11259
Data columns (total 19 columns):
 #   Column                   Non-Null Count  Dtype  
---  ------                   --------------  -----  
 0   AccountID                11260 non-null  int64  
 1   Churn                    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
```

## 3. Exploratory Data Analysis

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

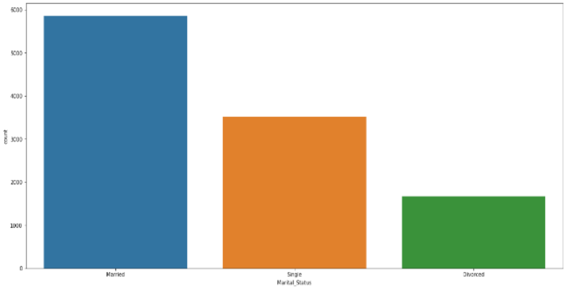
| | | |
|---|---|---|
| **AccountID** | account unique identifier ranges from  20000.0 to  31259.0 | |
| **Churn** | account churn flag (Target)<br>the data shows 0.831616 and 0.168384 as churn rate<br>9364 ,1896 is churn count | |
| **Tenure**<br>Description of Tenure<br>----------------------<br>count    11042.000000<br>mean        11.025086<br>std         12.879782<br>min          0.000000<br>25%          2.000000<br>50%          9.000000<br>75%         16.000000<br>max         99.000000 |  |  |

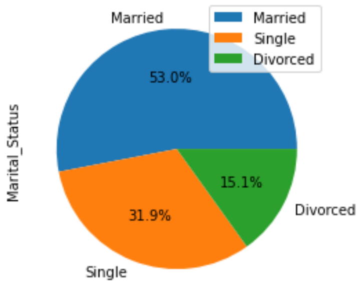
| | | |
|---|---|---|
| **City_Tier**<br>**1.0 7263**<br>**3.0 3405**<br>**2.0 480** |  |  |
| **CC_Contacted_LY**<br>Description of CC_Contacted_LY<br>------------------------------<br>count 11158.000000<br>mean 17.867091<br>std 8.853269<br>min 4.000000<br>25% 11.000000<br>50% 16.000000<br>75% 23.000000<br>max 132.000000 |  |  |
| **Payment**<br>**Debit Card 4587**<br>**Credit Card 3511**<br>**E wallet 1217**<br>**Cash on Delivery 1014**<br>**UPI 822** |  |  |
| **Gender**<br>**Male 6704**<br>**Female 4448** |  |  |

| | | |
|---|---|---|
| **Service_Score:** Satisfaction score given by customers of the account on service provided by company<br><br>**0.0 8**<br>**1.0 77**<br>**2.0 3251**<br>**3.0 5490**<br>**4.0 2331**<br>**5.0 5** |  |  |
| **Account_user_count**<br><br>**1.0 446**<br>**2.0 526**<br>**3.0 3261**<br>**4.0 4569**<br>**5.0 1699**<br>**6.0 315** |  |  |
| **account_segment**<br><br>**Regular 520**<br>**Regular_Plus 4124**<br>**Super 4062**<br>**Super_Plus 818**<br>**HNI 1639** |  |  |
| **CC_Agent_Score**<br><br>**1.0 2302**<br>**2.0 1164**<br>**3.0 3360**<br>**4.0 2127**<br>**5.0 2191** |  |  |

## Marital_Status

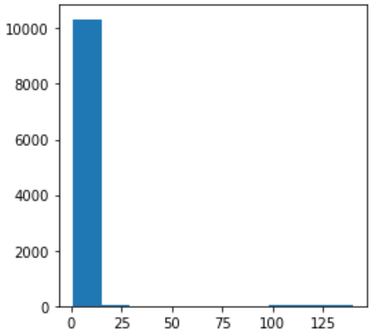
**Married 5860**
**Single 3520**
**Divorced 1668**

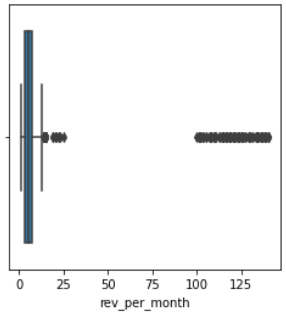




## rev_per_month

Monthly average revenue generated by account in last 12 months

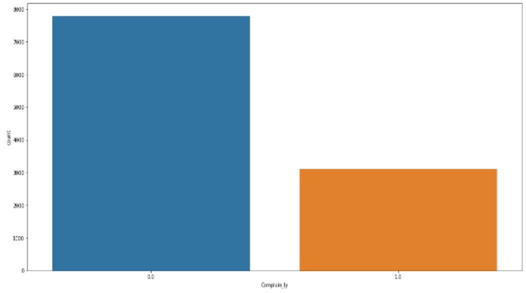


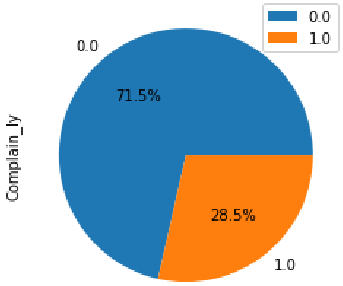


## Complain_ly

Any complaints has been raised by account in last 12 months

**0.0 7792**
**1.0 3111**

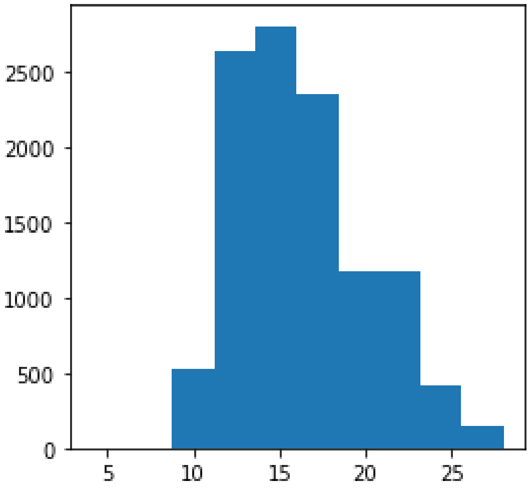


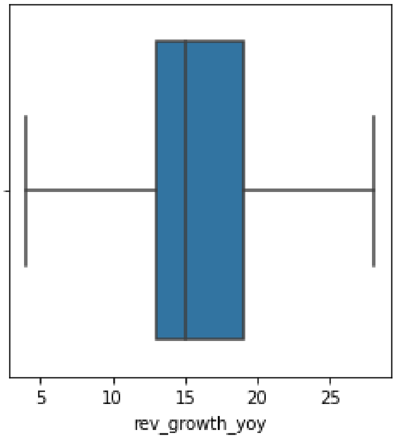


## rev_growth_yoy

revenue growth percentage of the account (last 12 months vs last 24 to 13 month)

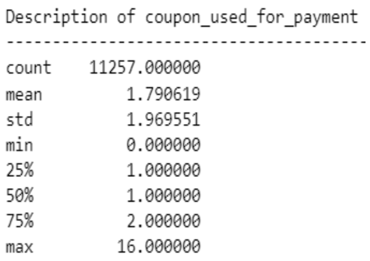
```
Description of rev_growth_yoy
------------------------------
count    11257.000000
mean        16.193391
std          3.757721
min          4.000000
25%         13.000000
50%         15.000000
75%         19.000000
max         28.000000
```



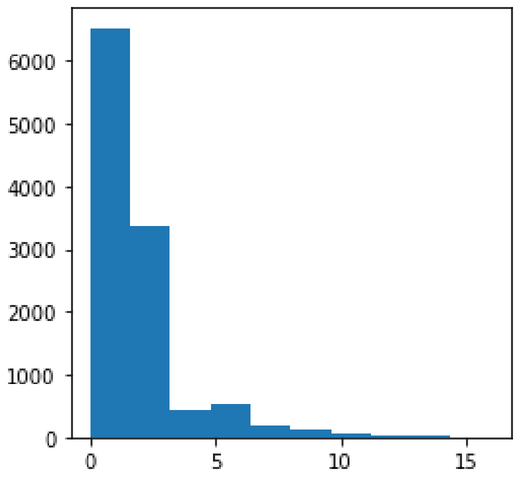


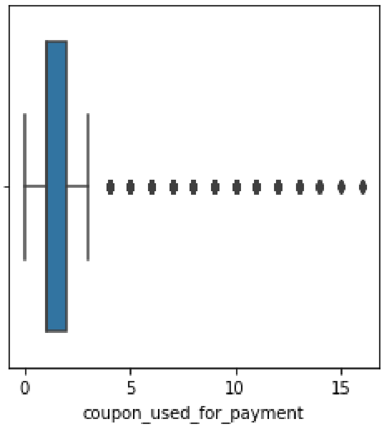
### coupon_used_for_payment

How many times customers have used coupons to do the payment in last 12 months



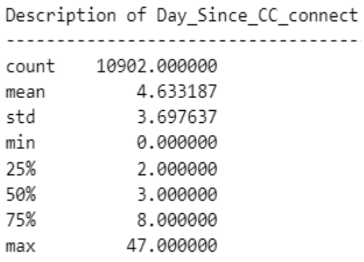
```
Description of coupon_used_for_payment
----------------------------------------
count    11257.000000
mean         1.790619
std          1.969551
min          0.000000
25%          1.000000
50%          1.000000
75%          2.000000
max         16.000000
```



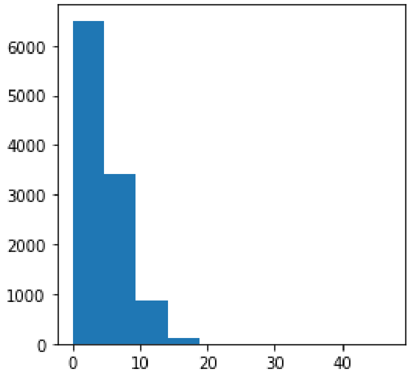


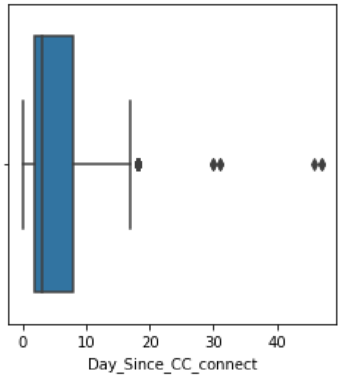
### Day_Since_CC_connect

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

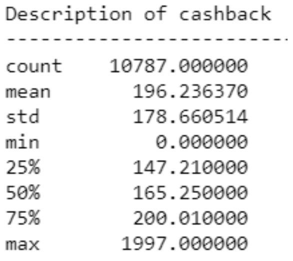


```
Description of Day_Since_CC_connect
----------------------------------
count    10902.000000
mean         4.633187
std          3.697637
min          0.000000
25%          2.000000
50%          3.000000
75%          8.000000
max         47.000000
```

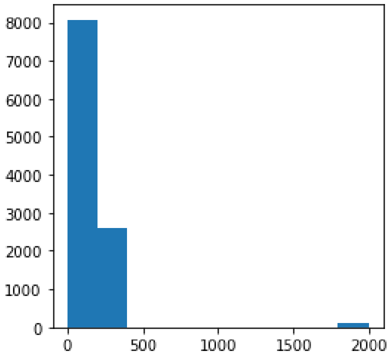


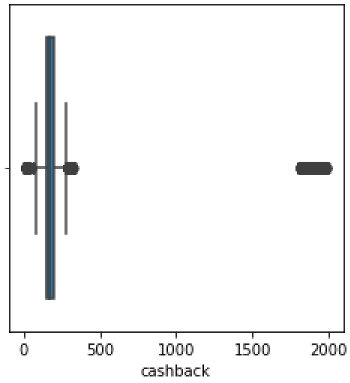


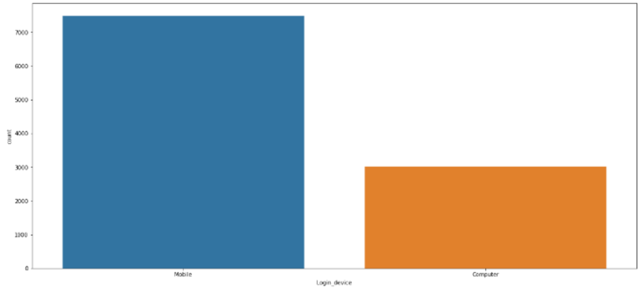
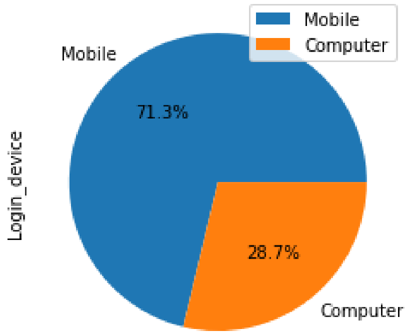
### Cashback



```
Description of cashback
-------------------------
count    10787.000000
mean       196.236370
std        178.660514
min          0.000000
25%        147.210000
50%        165.250000
75%        200.010000
max       1997.000000
```



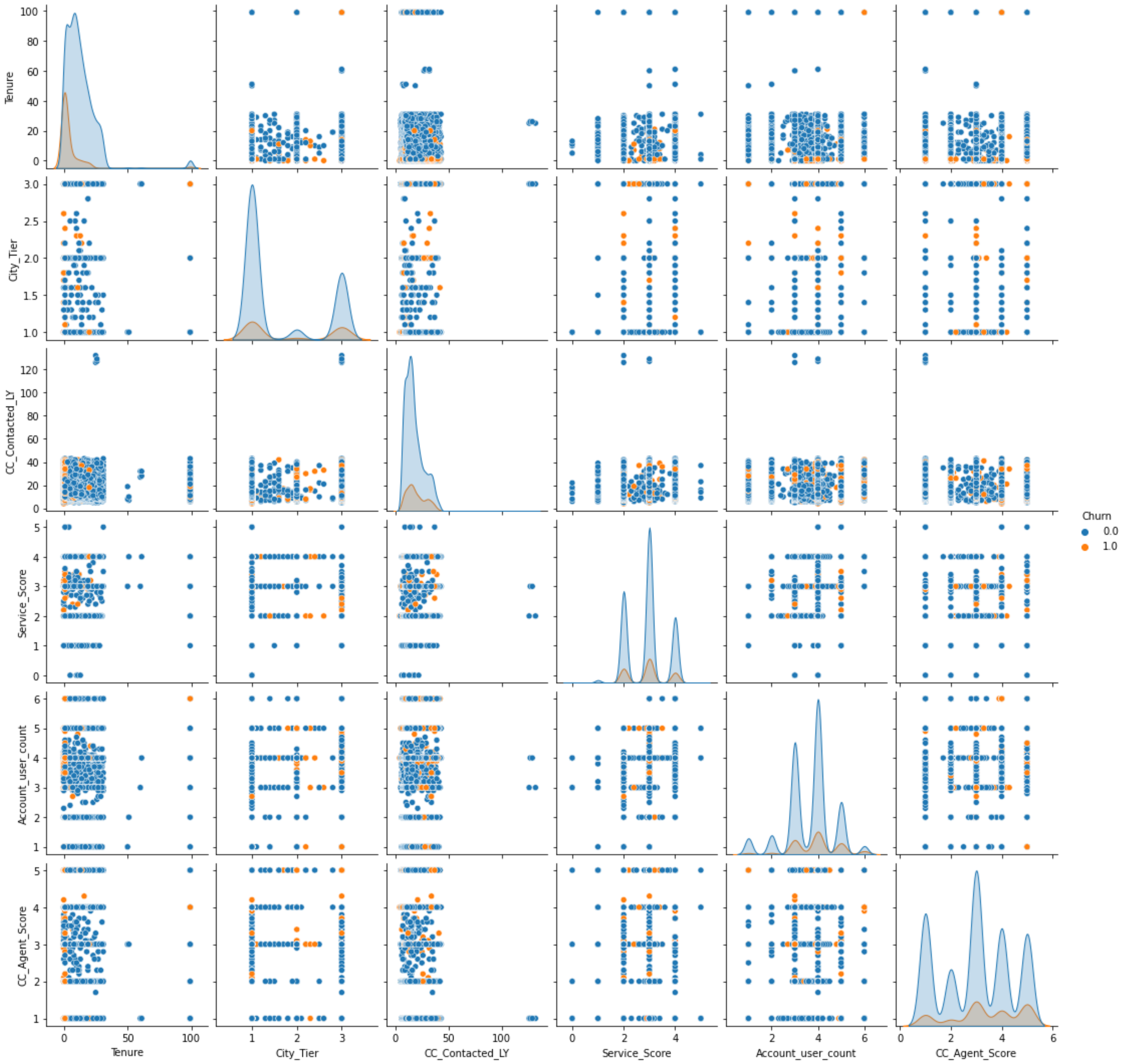


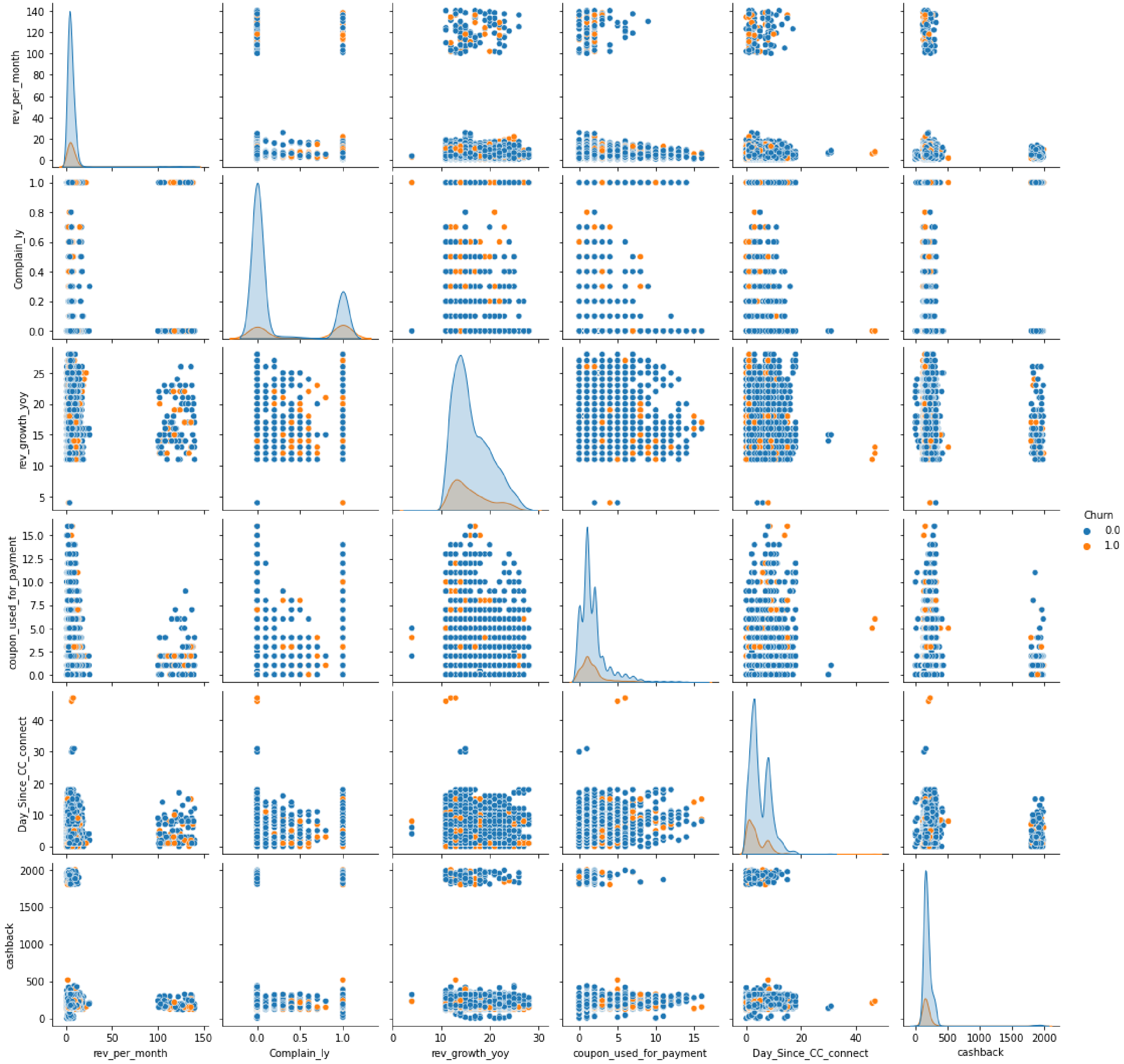
| | | |
|---|---|---|
| **Login_device**<br>**Mobile 7482**<br>**Computer 3018** |  |  |

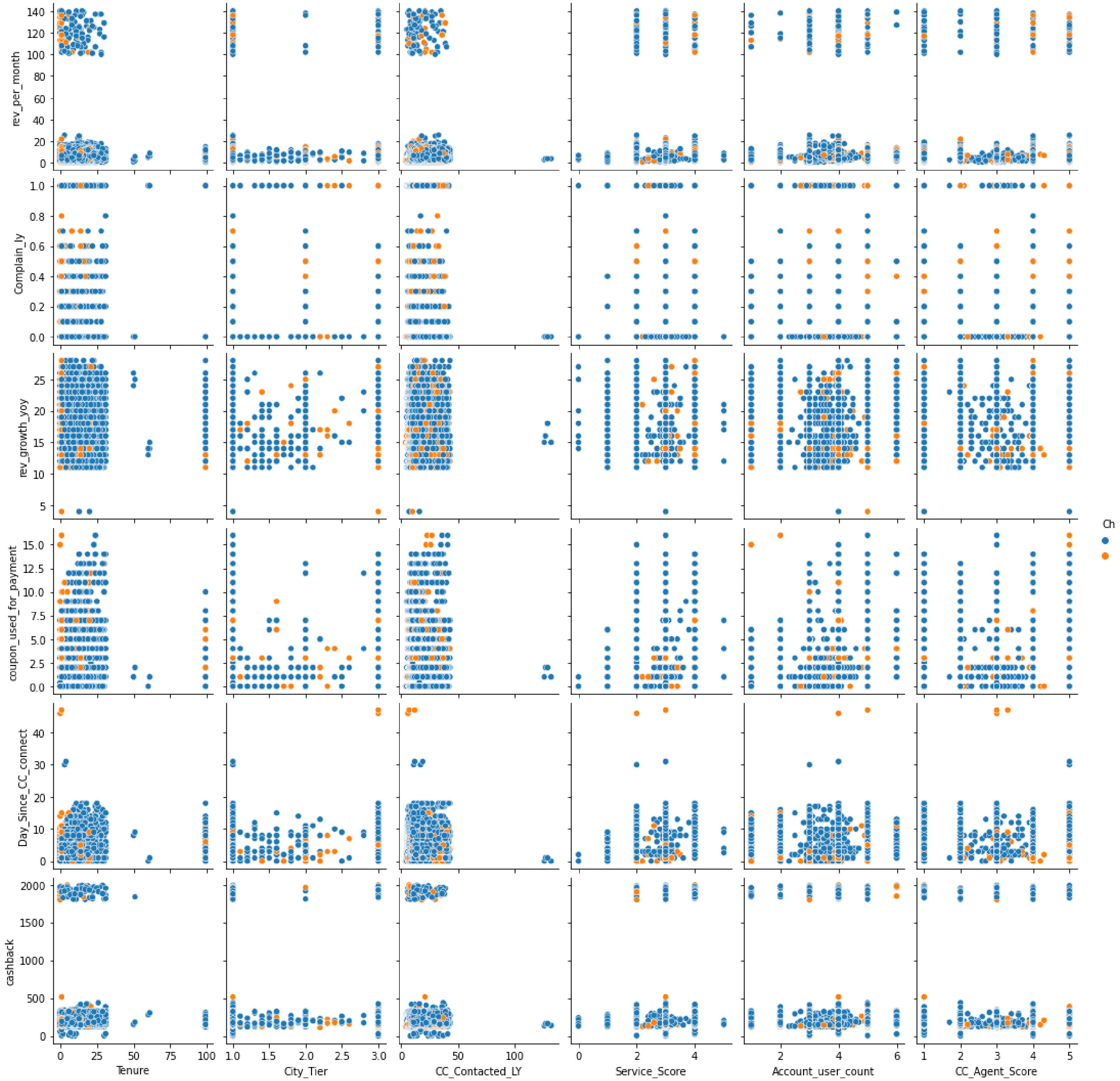
Observations from Univariate analysis:

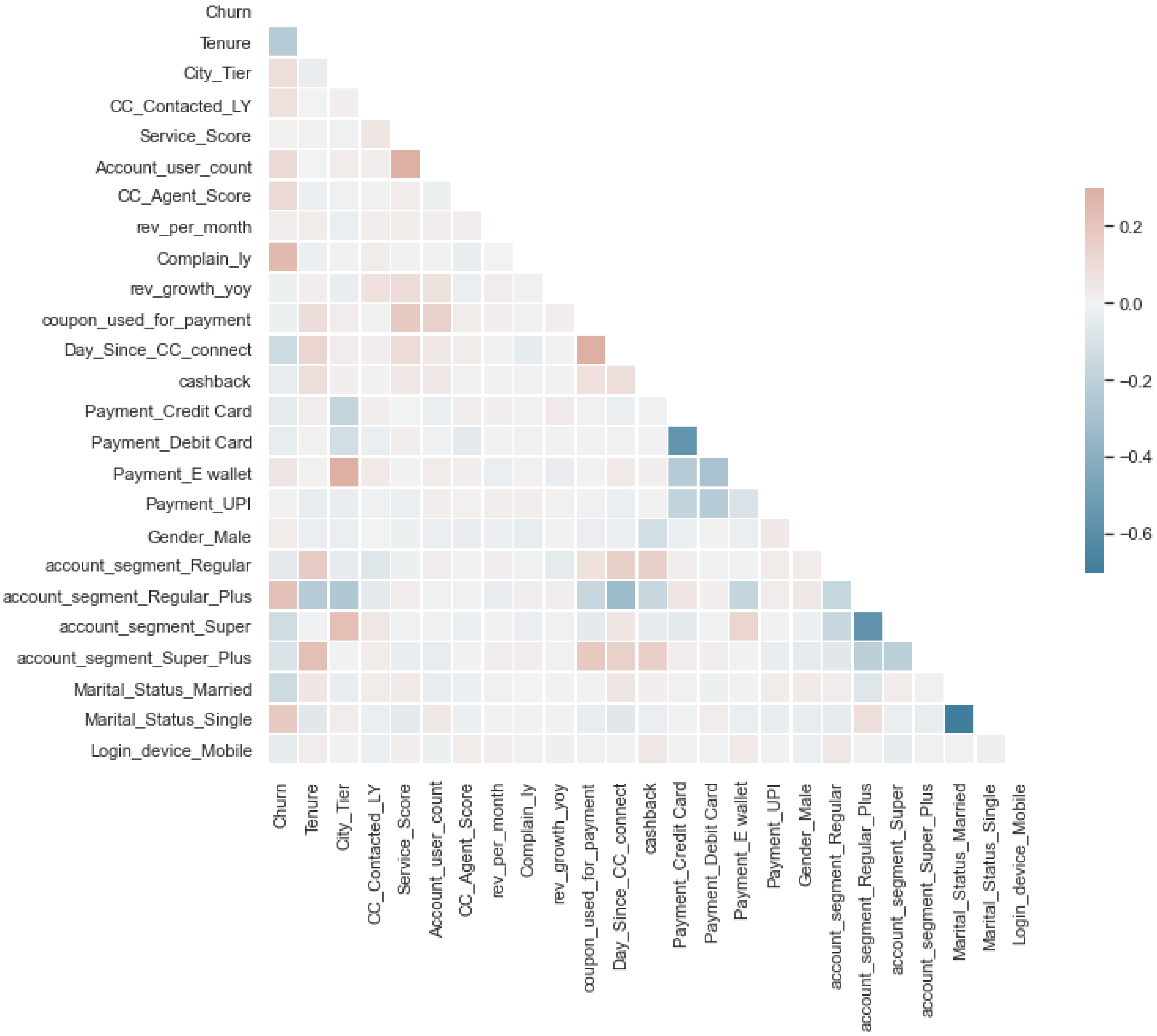
- Churn variable has 1896 customers who have churned and 9364 customers who have not churned which means churn rate is 16.8384%
- Tenure of account ranges from 0 to 99 months
- Tier of primary customer's city has 3 categories 1,2 and 3 with maximum count in tier 1 and minimum count in tier 2
- How many times all the customers of the account has contacted customer care in last 12months ranges from 4 to 132 with average being 17.86
- Preferred Payment mode of the customers in the account are Debit Card, Credit Card, E wallet, Cash on Delivery and UPI with maximum count for debit cards and minimum for UPI
- Gender of the primary customer of the account shows 6704 male and 4448 females which is 60.1% amle and 39.9% female
- Satisfaction score given by customers of the account on service provided by company ranges from 0 to 5 with majority rating of 3 and least count for 5. In other words, 49.2% have rated the service 3
- Number of customers tagged with this account ranges from 1 to 6 with most frequent being 4
- Account segmentation on the basis of spend shows 5 categories which includes HNI , regular, regular plus, super and super plus with 36.9% in regular plus, 36.4% in super, 14.7% in HNI, 7.3% in super plus and 4.7% in regular
- Satisfaction score given by customers of the account on customer care service provided by company ranges from 1 to 5 with 30.2% customers rating the company 3 , 20.7% rating the company 1, 19.7% rating the company 5, 19.1% rating the company 4 and 10.4% rating the company 2
- Marital status of the primary customer of the account shows that 53% are married ,31.9% are single and 15.1% are divorced
- Monthly average revenue generated by account in last 12 months ranges from 1 to 140
- Any complaints has been raised by account in last 12 months. This is 71.5% vs 28.5%
- revenue growth percentage of the account (last 12 months vs last 24 to 13 month) ranges from 4 to 28
- How many times customers have used coupons to do the payment in last 12 months ranges from 0 to 16 with mean of all values being 1.79
- Number of days since no customers in the account has contacted the customer care ranges from 0 to 47 with mean of all counts being 4.62
- Monthly average cashback generated by account in last 12 months ranges from 0 to 1997 and mean is 196.05
- Preferred login device of the customers in the account has mobile and computer category where 71.3% is mobile and 28.7% belongs to computer

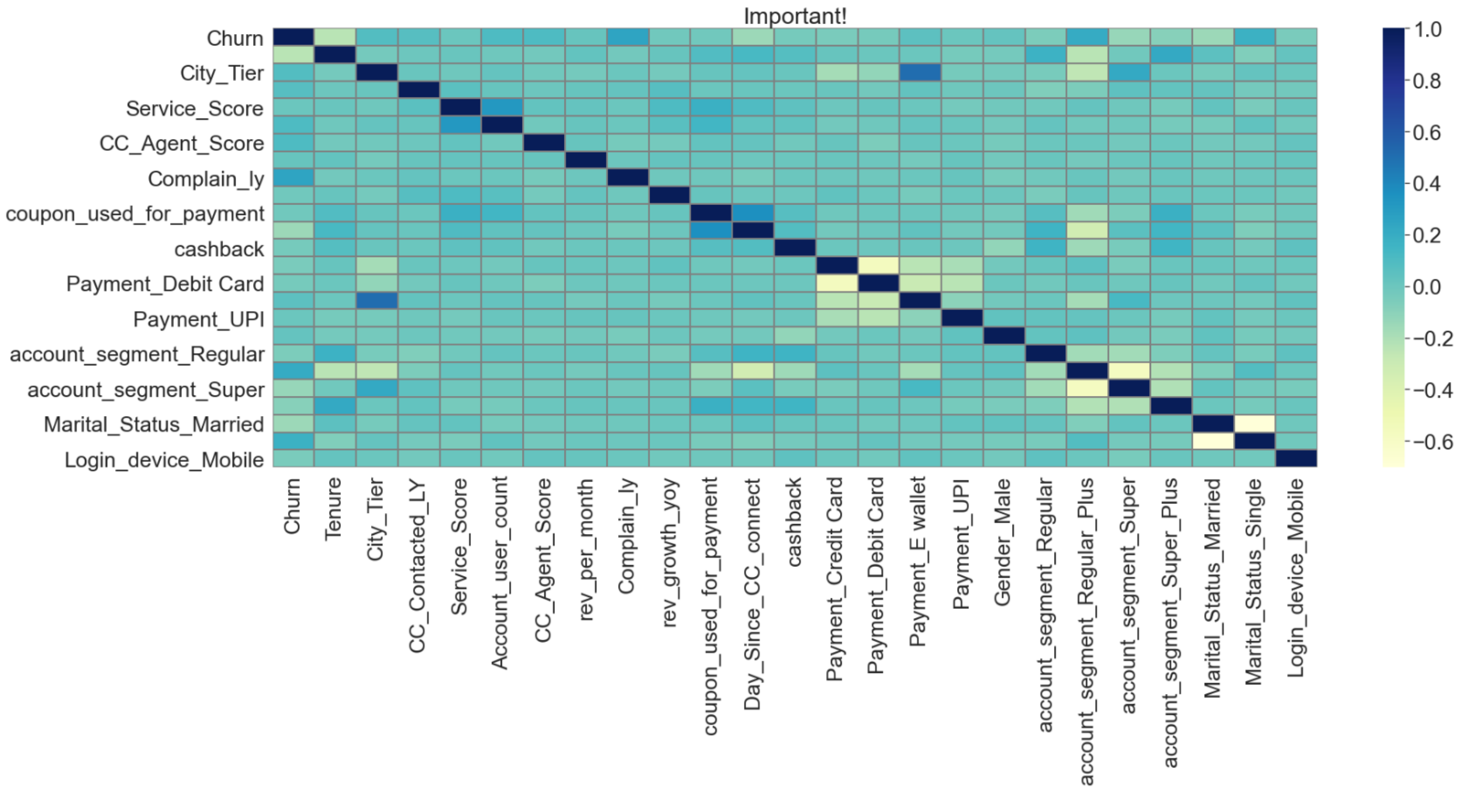
**b) Bivariate analysis (relationship between different variables, correlations)**

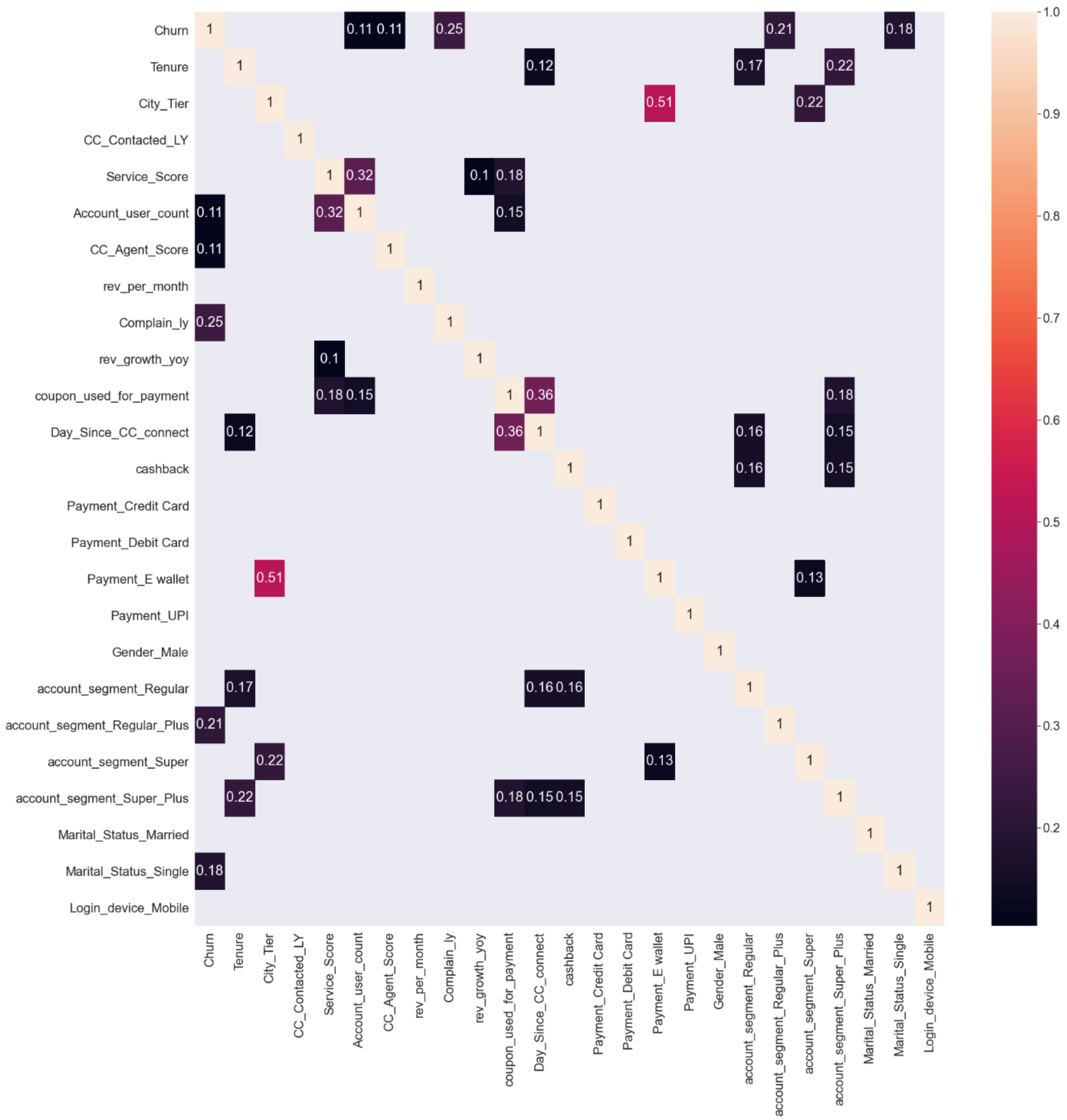

Tenure
City_Tier
CC_Contacted_LY
Service_Score
Account_user_count
CC_Agent_Score
Churn
0.0
1.0


rev_per_month
Complain_ly
rev_growth_yoy
coupon_used_for_payment
Day_Since_CC_connect
cashback
Churn
0.0
1.0
140
120
100
80
60
40
20
0
1.0
0.8
0.6
0.4
0.2
0.0
25
20
15
10
5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
40
30
20
10
0
2000
1500
1000
500
0
0
50
100
150
0.0
0.5
1.0
0
10
20
30
0
5
10
15
0
20
40
0
500
1000
1500
2000


rev_per_month
0
20
40
60
80
100
120
140
Complain_ly
0.0
0.2
0.4
0.6
0.8
1.0
rev_growth_yoy
5
10
15
20
25
coupon_used_for_payment
0.0
2.5
5.0
7.5
10.0
12.5
15.0
Day_Since_CC_connect
0
10
20
30
40
cashback
0
500
1000
1500
2000
Ch
0
25
50
75
100
Tenure
1.0
1.5
2.0
2.5
3.0
City_Tier
0
50
100
CC_Contacted_LY
0
2
4
Service_Score
2
4
6
Account_user_count
1
2
3
4
5
CC_Agent_Score


Churn
Tenure
City_Tier
CC_Contacted_LY
Service_Score
Account_user_count
CC_Agent_Score
rev_per_month
Complain_ly
rev_growth_yoy
coupon_used_for_payment
Day_Since_CC_connect
cashback
Payment_Credit Card
Payment_Debit Card
Payment_E wallet
Payment_UPI
Gender_Male
account_segment_Regular
account_segment_Regular_Plus
account_segment_Super
account_segment_Super_Plus
Marital_Status_Married
Marital_Status_Single
Login_device_Mobile
Churn
Tenure
City_Tier
CC_Contacted_LY
Service_Score
Account_user_count
CC_Agent_Score
rev_per_month
Complain_ly
rev_growth_yoy
coupon_used_for_payment
Day_Since_CC_connect
cashback
Payment_Credit Card
Payment_Debit Card
Payment_E wallet
Payment_UPI
Gender_Male
account_segment_Regular
account_segment_Regular_Plus
account_segment_Super
account_segment_Super_Plus
Marital_Status_Married
Marital_Status_Single
Login_device_Mobile
0.2
0.0
−0.2
−0.4
−0.6


Important!
Churn
City_Tier
Service_Score
CC_Agent_Score
Complain_ly
coupon_used_for_payment
cashback
Payment_Debit Card
Payment_UPI
account_segment_Regular
account_segment_Super
Marital_Status_Married
Login_device_Mobile
Churn
Tenure
City_Tier
CC_Contacted_LY
Service_Score
Account_user_count
CC_Agent_Score
rev_per_month
Complain_ly
rev_growth_yoy
coupon_used_for_payment
Day_Since_CC_connect
cashback
Payment_Credit Card
Payment_Debit Card
Payment_E wallet
Payment_UPI
Gender_Male
account_segment_Regular
account_segment_Regular_Plus
account_segment_Super
account_segment_Super_Plus
Marital_Status_Married
Marital_Status_Single
Login_device_Mobile
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6



Observations from bi-variate analysis

- Payment_E wallet and City_Tier show correlation of 0.510734
- coupon_used_for_payment and Day_Since_CC_connect show correlation of 0.362793
- Service_Score and Account_user_count show correlation of 0.320321
- Complain_ly and Churn show correlation of 0.249521
- City_Tier and account_segment_Super show correlation of 0.224018
- Tenure and account_segment_Super_Plus show correlation of 0.223684
- Churn and account_segment_Regular_Plus show correlation of 0.213095

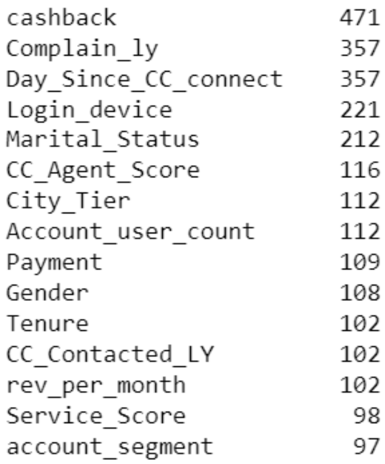
After plotting the pairplot and heatmap we can see that Payment E wallet and City Tier show more than 0.5 correlation

### a) Removal of unwanted variables (if applicable)

We are not going to remove any variables because we have used knn imputation for missing values and dummy encoding for categorical variables. However, anomalies in records we have first imputed with null value then treated them with knn imputation.

### b) Missing Value treatment (if applicable)

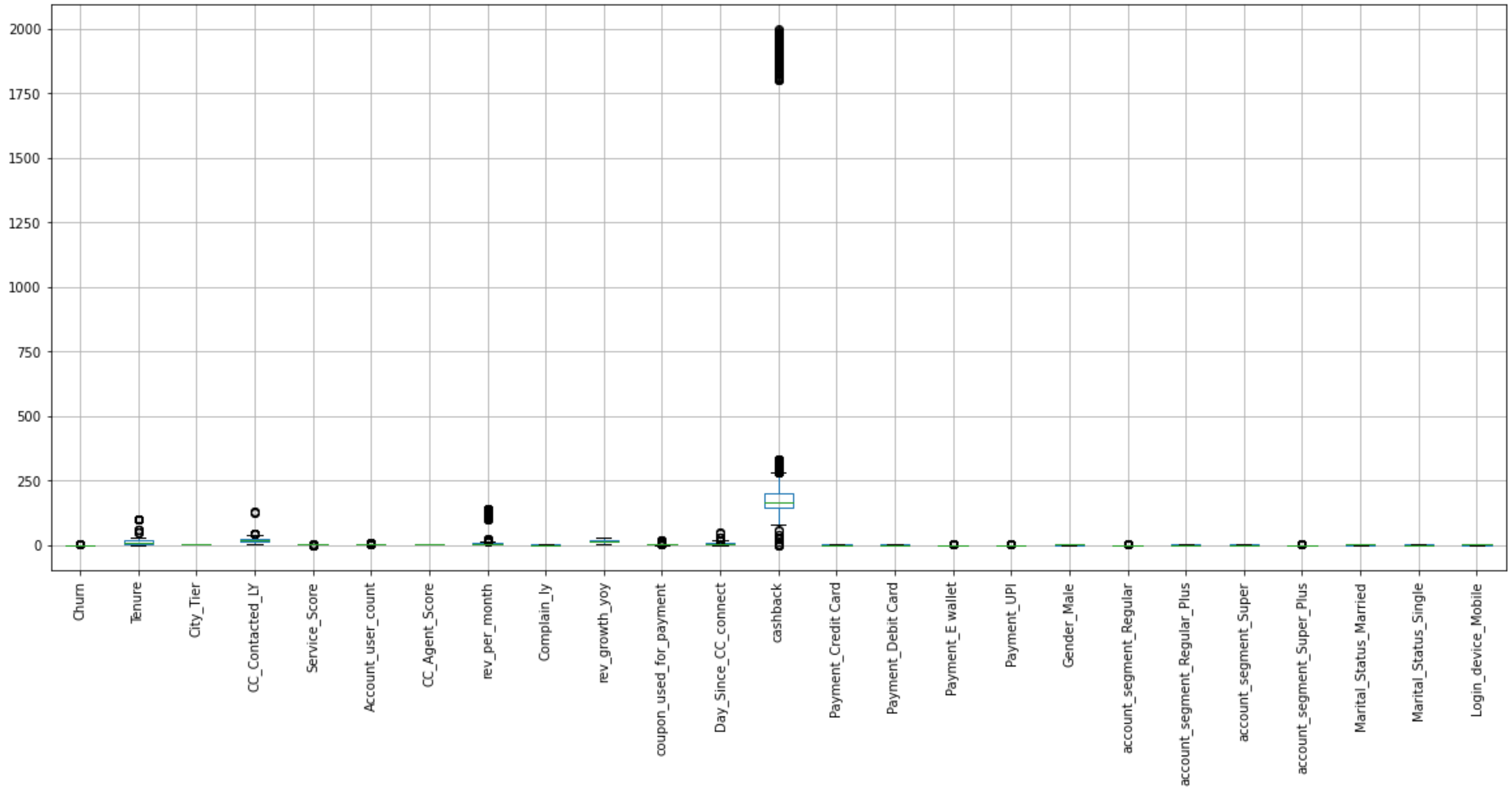
We are going to impute the missing values with knn imputation which is a method that imputes missing values based on the neighboring 10 values. The idea in kNN methods is to identify 'k' samples **in** the dataset that are similar or close in the space. Then we use these 'k' samples to estimate the value of the missing data points. Each sample's missing values are imputed using the mean value of the 'k'-neighbors found in the dataset. We have used KNNImputer from klearn.impute package.

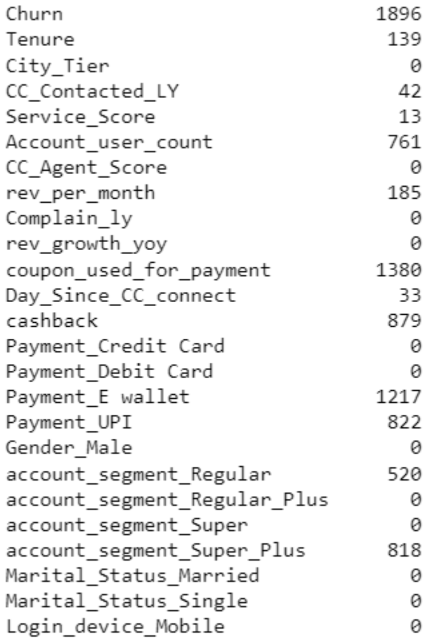


```
cashback                471
Complain_ly             357
Day_Since_CC_connect    357
Login_device            221
Marital_Status          212
CC_Agent_Score          116
City_Tier               112
Account_user_count      112
Payment                 109
Gender                  108
Tenure          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

### d) Outlier treatment (if required)

We have not treated outlier since it will affect our results significantly





```
Churn                           1896
Tenure                           139
City_Tier                          0
CC_Contacted_LY                   42
Service_Score                     13
Account_user_count               761
CC_Agent_Score                     0
rev_per_month                    185
Complain_ly                        0
rev_growth_yoy                     0
coupon_used_for_payment         1380
Day_Since_CC_connect              33
cashback                         879
Payment_Credit Card                0
Payment_Debit Card                 0
Payment_E wallet                1217
Payment_UPI                      822
Gender_Male                        0
account_segment_Regular          520
account_segment_Regular_Plus       0
account_segment_Super              0
account_segment_Super_Plus       818
Marital_Status_Married             0
Marital_Status_Single              0
Login_device_Mobile                0
```

### e) Variable transformation (if applicable)

Dummy Encoding

Dummy coding scheme is similar to one-hot encoding. This categorical data encoding method transforms the categorical variable into a set of binary variables (also known as dummy variables). In the case of one-hot encoding, for N categories in a variable, it uses N binary variables. The dummy encoding is a small improvement over one-hot-encoding. Dummy encoding uses N-1 features to represent N labels/categories.

### f) Addition of new variables (if required)

After dummy encoding, we have 25 columns, since we have categorical columns that needs to be coded before modelling

## 4. Business insights from EDA

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

If your data is not split 50/50 then it is not “balanced”. We have 16.8384 % churned customers and 83.1616% active customers. This is example of unbalanced data. There is no standard threshold for how your classes should be split. Depending on the type of business problem you are solving and the type of industry, the split might become more meaningful. For example, an 80/20 split for a customer retention problem might be acceptable, but not so much for a healthcare problem.

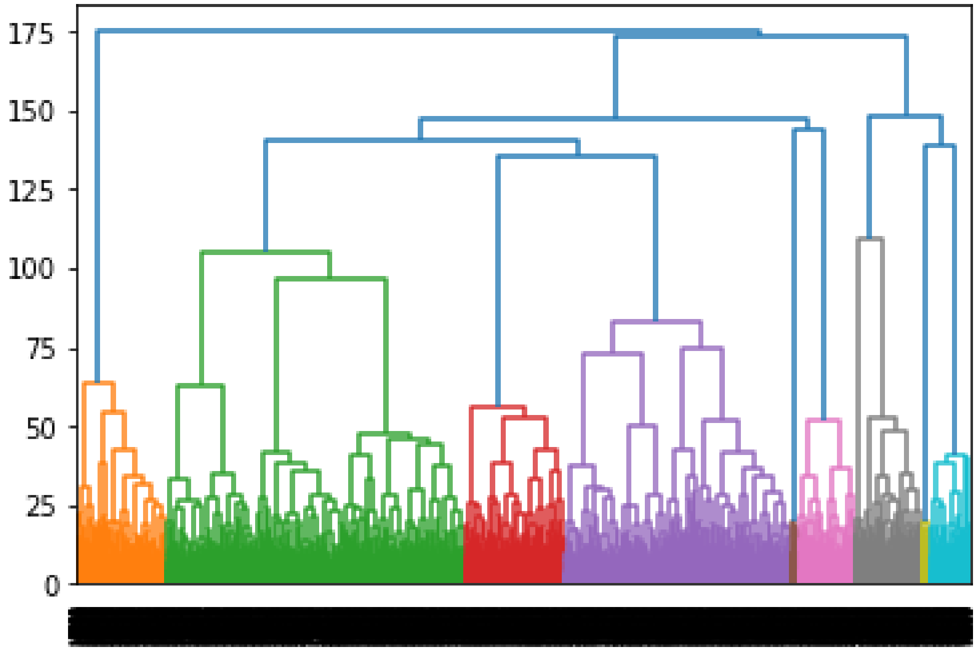
Unbalanced data can affect accuracy, precision and recall. Because there were so many customers who did not churn, the high number of true negatives made the accuracy be very high, which would be misleading since we wanted to focus on those customers that did leave, and try to discover what we could do to prevent that from happening with other customers.

How to balance the data?

- Under sampling: Under sampling consists of deleting observations from your over-represented class. In this case, those would be the active customers.
- Oversampling: Oversampling consists of adding copies of observations from your under-represented class. In this case, those would be the churned customers.
- SMOTE: THIS is better! SMOTE stands for Synthetic Minority Over-sampling Technique. This method creates synthetic samples of your data, so rather than taking copies of observations, SMOTE uses a distance measure to create synthetic samples of data points that would not be far from your data points.

SMOTE is the best method to balance the data.

### b) Any business insights using clustering (if applicable)



We have performed clustering,  we do see that there are some visible clusters which could affect our analysis.

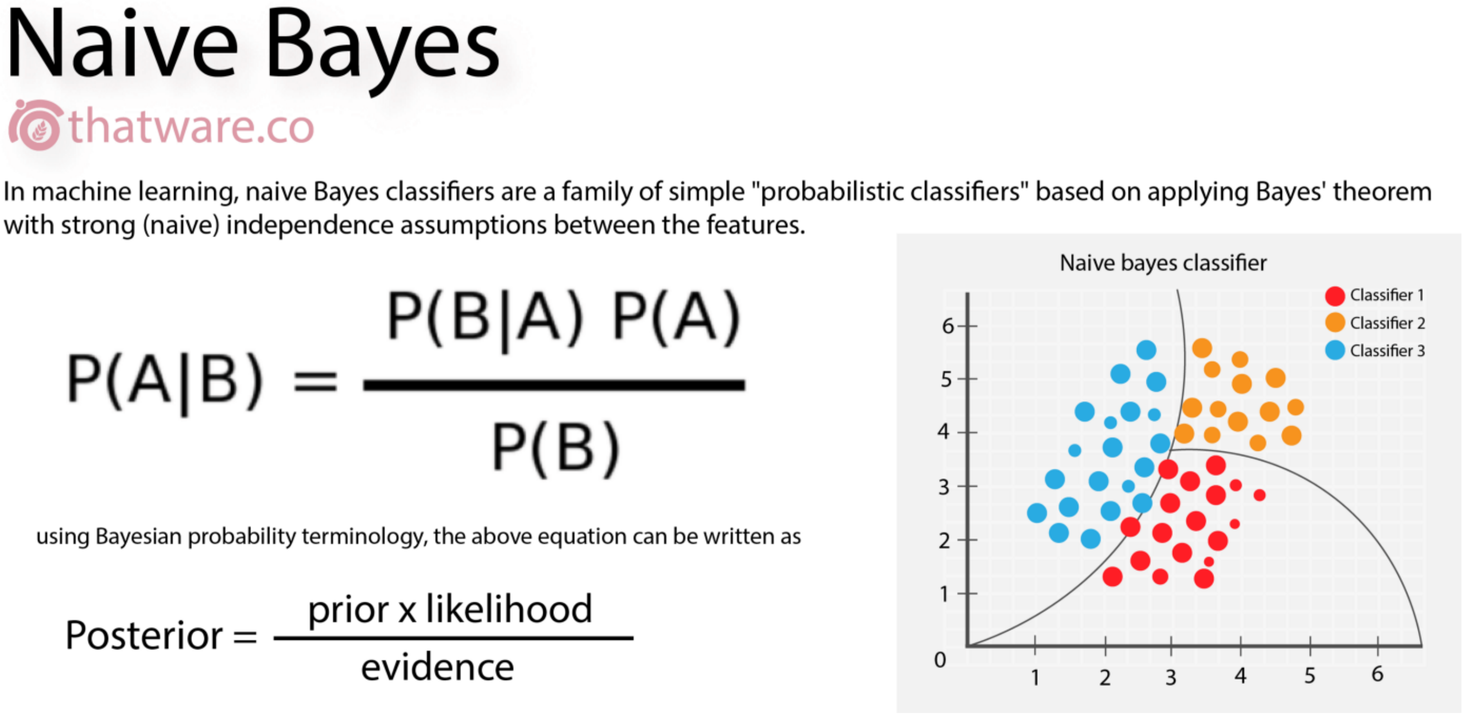
**c) Any other business insights**

- Churn rate is the percentage of subscribers to a service that discontinue their subscription to that service in a given time period. In order for a company to expand its clientele, its growth rate (i.e. its number of new customers) must exceed its churn rate.
- Churn rates are often used to indicate the strength of a company’s customer service division and its overall growth prospects. Lower churn rates suggest a company is, or will be, in a better or stronger competitive state. Customer loss impacts carriers significantly as they often make a significant investment to acquire customers.
- The ability to predict that a particular customer is at a high risk of churning, while there is still time to do something about it, represents a huge additional potential revenue source for every online business.

**1). Model building and interpretation.**

a. Build various models (You can choose to build models for either or all of descriptive, predictive or prescriptive purposes)

First let us begin with Naïve Bayes classifier model. Naïve Bayes Classifier is one of the simple and most effective Classification algorithms which helps in building the fast machine learning models that can make quick predictions. It is a probabilistic classifier, which means it predicts on the basis of the probability of an object. We have used GaussianNB algorithm and metrics from sklearn.naive_bayes package to build this model.



Naive Bayes

thatware.co

In machine learning, naive Bayes classifiers are a family of simple "probabilistic classifiers" based on applying Bayes' theorem with strong (naive) independence assumptions between the features.

$$P(A|B) = \frac{P(B|A)\ P(A)}{P(B)}$$

using Bayesian probability terminology, the above equation can be written as

$$\text{Posterior} = \frac{\text{prior x likelihood}}{\text{evidence}}$$

1. <u>Naive Bayes Model</u>

**After running the naïve bayes model on training data, we have got the following confusion matrix:**

```
[[5572  984]
 [ 500  826]]
```

```
True Positive: 5572 active customers are classified correctly
True Negative: 826 churned customers are classified correctly
False Positive: 500 churned customers are classified as active customer incorrectly
False Negative: 984 active customers are classified as churned customer incorrectly
```

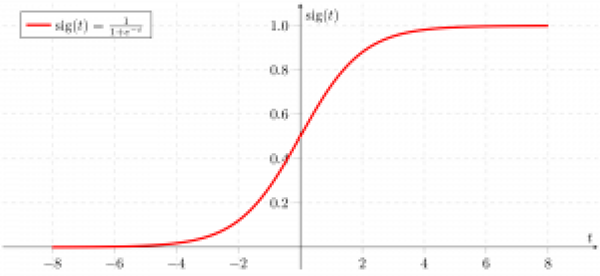
**After running the naïve bayes model on test data, we have got the following confusion matrix:**

```
[[2376  432]
 [ 245  325]]
```

```
True Positive: 2376 active customers are classified correctly
True Negative: 325 churned customers are classified correctly
False Positive: 245 churned customers are classified as active customer incorrectly
False Negative: 432 active customers are classified as churned customer incorrectly
```

### 2. Logistic Regression

Next we have tried LogisticRegression package from sklearn.linear_model library. Logistic regression models the data using the sigmoid function.



**After running the logistic regression model on training data, we have got the following confusion matrix:**

```
[[6368  188]
 [ 785  541]]
```

```
True Positive: 6368 active customers are classified correctly
True Negative: 541 churned customers are classified correctly
False Positive: 785 churned customers are classified as active customer incorrectly
False Negative: 188 active customers are classified as churned customer incorrectly
```

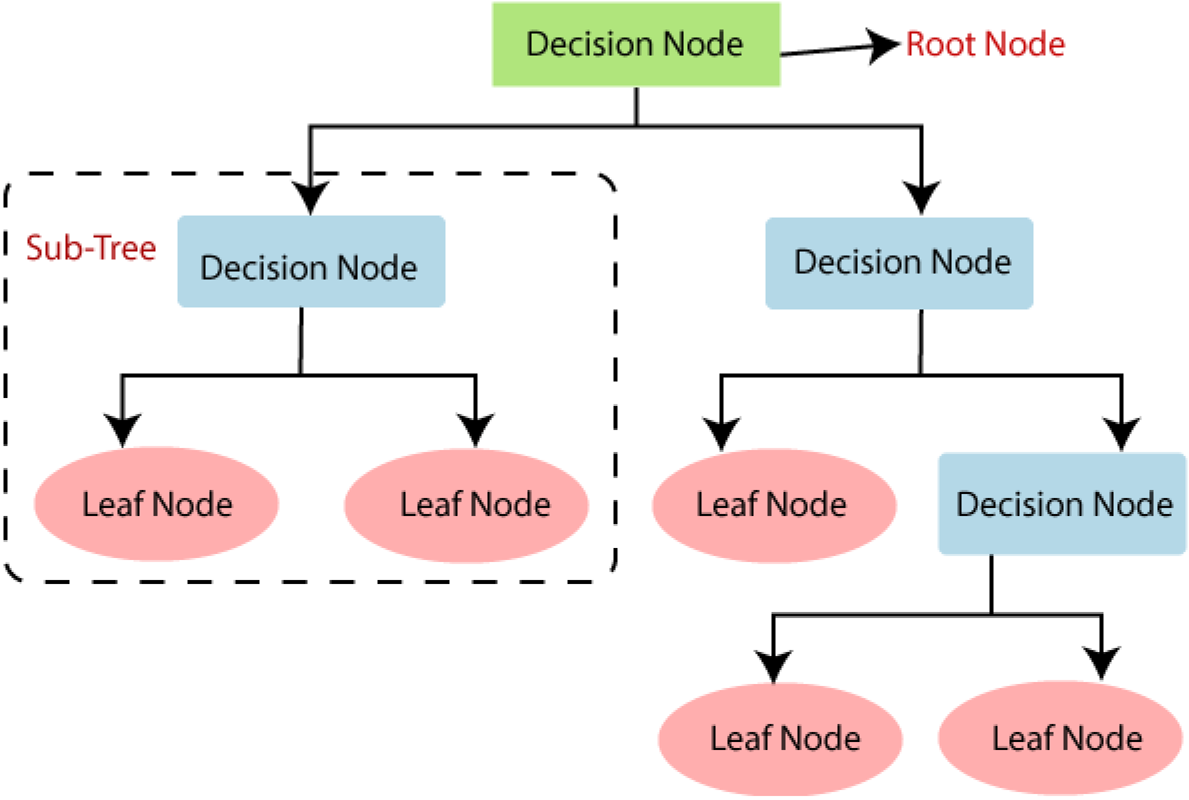
**After running the logistic regression model on test data, we have got the following confusion matrix:**

```
[[2723   85]
 [ 335  235]]
```

```
True Positive: 2723 active customers are classified correctly
True Negative: 235 churned customers are classified correctly
False Positive: 335 churned customers are classified as active customer incorrectly
False Negative: 85 active customers are classified as churned customer incorrectly
```

### 3. Decision Tree

Next, we have tried decision tree algorithm which is actually a white box model. We have used  DecisionTreeClassifier package  from sklearn.tree library to classify the data.  In Decision Trees, for predicting a class label for a record we start from the root of the tree. We compare the values of the root attribute with the record's attribute. On the basis of comparison, we follow the branch corresponding to that value and jump to the next node.



**After running the decision tree model on training data, we have got the following confusion matrix:**

```
[[6556    0]
 [   0 1326]]
```

```
True Positive: 6556 active customers are classified correctly
True Negative: 1326 churned customers are classified correctly
False Positive: 0 churned customers are classified as active customer incorrectly
False Negative: 0 active customers are classified as churned customer incorrectly
```

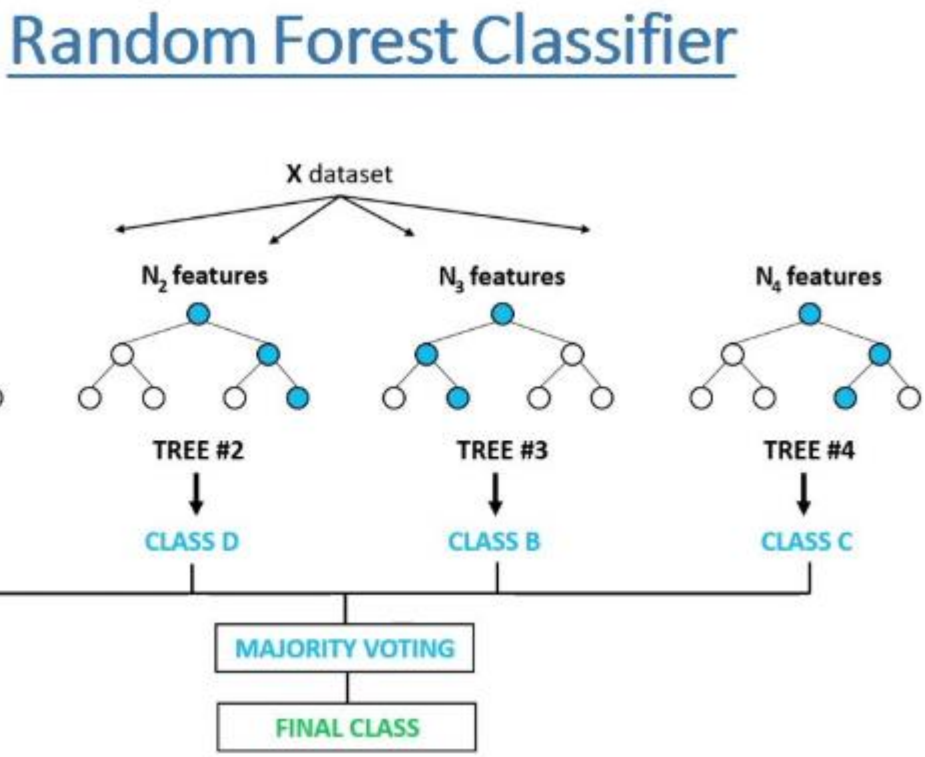
**After running the decision tree model on test data, we have got the following confusion matrix:**

```
[[2712   96]
 [ 104  466]]
```

```
True Positive: 2712 active customers are classified correctly
True Negative: 466 churned customers are classified correctly
False Positive: 104 churned customers are classified as active customer incorrectly
False Negative: 96 active customers are classified as churned customer incorrectly
```

### 4. Random Forest

Now let us try random forest model using RandomForestClassifier package from sklearn.ensemble library. The random forest is a classification algorithm consisting of many decisions trees. It uses bagging and feature randomness when building each individual tree to try to create an uncorrelated forest of trees whose prediction by committee is more accurate than that of any individual tree.



**After running the random forest model on training data, we have got the following confusion matrix:**

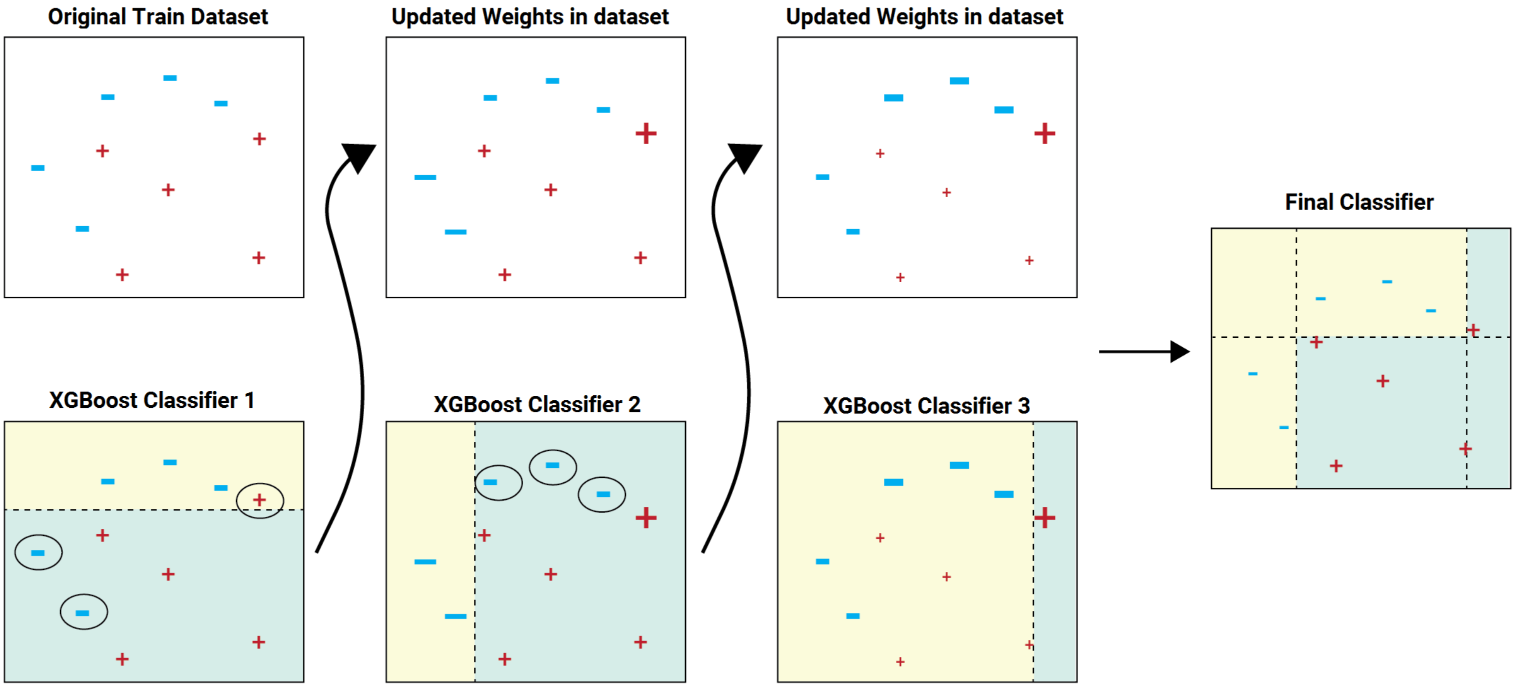
```
[[6556    0]
 [   0 1326]]

True Positive: 6556 active customers are classified correctly
True Negative: 1326 churned customers are classified correctly
False Positive: 0 churned customers are classified as active customer incorrectly
False Negative: 0 active customers are classified as churned customer incorrectly
```

**After running the random forest model on test data, we have got the following confusion matrix:**

```
[[2802    6]
 [  96  474]]

True Positive: 2802 active customers are classified correctly
True Negative: 474 churned customers are classified correctly
False Positive: 96 churned customers are classified as active customer incorrectly
False Negative: 6 active customers are classified as churned customer incorrectl
```

### 5. XGBoost

Finally, we have used XGBClassifier algorithm from xgboost library. XGBoost (eXtreme Gradient Boosting) is a popular supervised-learning algorithm used for regression and classification on large datasets. It uses sequentially-built shallow decision trees to provide accurate results and a highly-scalable training method that avoids overfitting.



**After running the XGBoost model on training data, we have got the following confusion matrix:**

```
[[6555    1]
 [   3 1323]]

True Positive: 6555 active customers are classified correctly
True Negative: 1323 churned customers are classified correctly
False Positive: 3 churned customers are classified as active customer incorrectly
False Negative: 1 active customers are classified as churned customer incorrectly
```

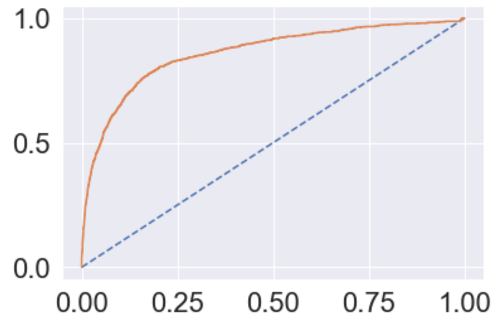
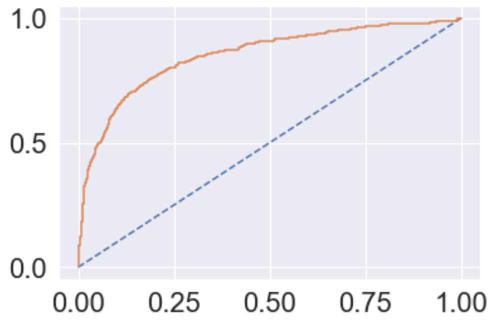
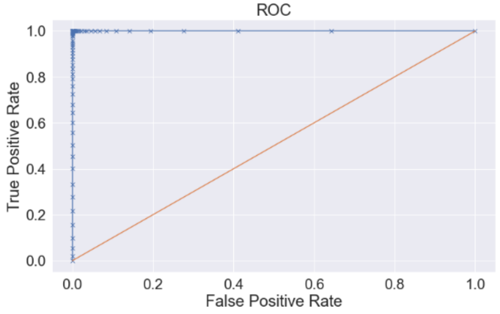
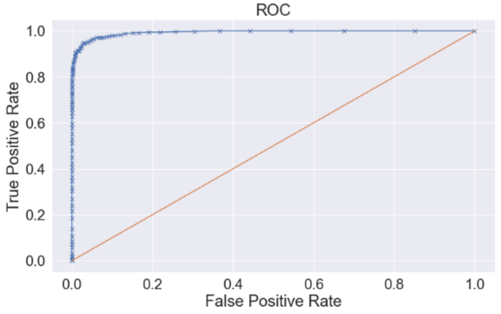
**After running the XGBoost model on test data, we have got the following confusion matrix:**

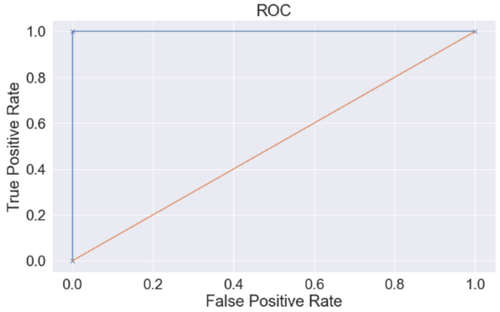
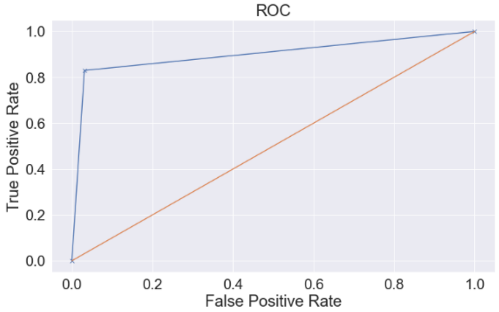
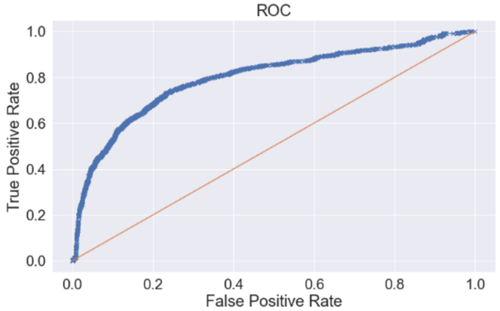
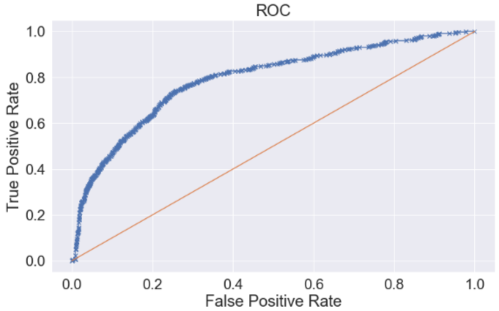
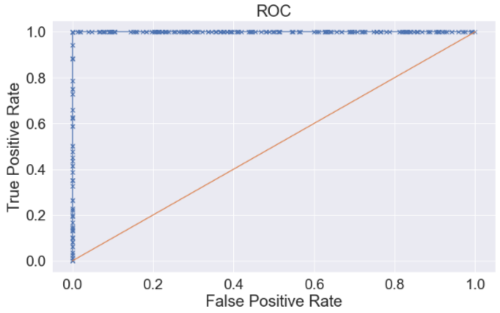
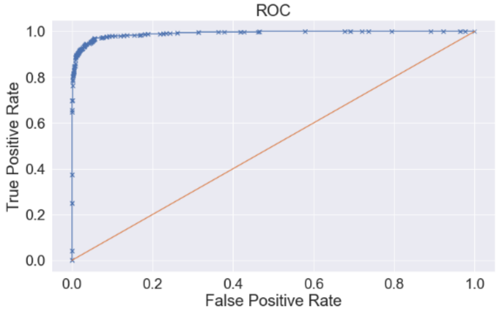
```
[[2785   23]
 [  75  495]]

True Positive: 2785 active customers are classified correctly
True Negative: 495 churned customers are classified correctly
False Positive: 75 churned customers are classified as active customer incorrectly
False Negative: 23 active customers are classified as churned customer incorrectly
```

b. Test your predictive model against the test set using various appropriate performance metrics

To gauge the models we have built for churn analysis, we have created a table with accuracy,precision,recall,F1 score, AUC so that we can understand the metrices better.

| | accuracy | | precision | | recall | | F1 score | | Fit |
|---|---|---|---|---|---|---|---|---|---|
| | train | test | train | test | train | test | train | test | |
| **XGBoost** | 1.00 | **0.98** | 1.00 | 0.96 | 1.00 | **0.90** | 1.00 | **0.93** | robust |
| **Logistic Regression** | 0.89 | 0.89 | 0.77 | 0.76 | 0.51 | 0.49 | 0.62 | 0.59 | robust |
| **Random Forest** | 1.00 | **0.97** | 1.00 | **0.99** | 1.00 | 0.85 | 1.00 | 0.91 | robust |
| **Decision Tree** | 1.00 | 0.94 | 1.00 | 0.86 | 1.00 | 0.81 | 1.00 | 0.83 | robust |
| **Naive Bayes Model** | 0.73 | 0.65 | 0.69 | 0.30 | 0.83 | 0.82 | 0.75 | 0.44 | robust |

| | | |
|---|---|---|
| Logistic regression |  <br>AUC: 0.860 |  <br>AUC: 0.860 |
| Random forest |  <br>Area under Curve is 0.9999999999999999 |  <br>Area under Curve is 0.9931945694007097 |

| | | |
|---|---|---|
| Decision Tree |  <br>Area under Curve is 1.0 |  <br>Area under Curve is 0.8992427650322388 |
| Naive Bayes Model |  <br>Area under Curve is 0.7980243535908754 |  <br>Area under Curve is 0.7915810716249312 |
| XGBoost |  <br>Area under Curve is 0.9999994248415093 |  <br>Area under Curve is 0.9911093617233968 |

After analyzing the values, we can say that XGBoost is the best performing model out of all the 5 models.

When we look at accuracy XGBoost has given us the best accuracy of 98%, but we got 99% precision for random forest model. We have achieved 90% recall and 93% F1-score using  XGBoost model. AUC is almost same for both Random forest and XGBoost model. In conclusion, we can say the XGBoost model has yielded outstanding results across all metrices.

c. Interpretation of the model(s)

We can see that clearly XGBoost has outperformed among all the 5 models in terms of recall, accuracy, f1-score. Random forest gave us better precision and AUC too but still when we compare all other metrices XGBoost is a better performing model.  can our model accurately detect churn to help retain these customers?

Customer churn, also known as attrition, occurs when a customer stop doing business with a company. Understanding and detecting churn is the first step to retaining these customers and improving the company’s offerings. In test data, out of 570 customers, we have predicted 495 customers correctly.

The decision tree model shows that the most important features are Tenure, cashback, Day_Since_CC_connect, CC_Agent_Score, rev_per_month, Complain_ly, rev_growth_yoy which contribute in deciding churn.

The business need to look into Tenure of account, Monthly average cashback generated by account in last 12 months, Number of days since no customers in the account has contacted the customer care, Any complaints has been raised by account in last 12 months, Satisfaction score given by customers of the account on customer care service provided by company, revenue growth percentage of the account (last 12 months vs last 24 to 13 month) to ensure that the customer stays with the company.

**2). Model Tuning and business implication**

a. Ensemble modelling, wherever applicable

Ensemble modeling is a process where multiple diverse models are created to predict an outcome, either by using many different modeling algorithms or using different training data sets. The ensemble model then aggregates the prediction of each base model and results in once final prediction for the unseen data.

Common Ensemble Techniques include bagging and boosting.

**Bagging** reduces chances of over fitting by training each model only with a randomly chosen subset of the training data. Training can be done in parallel. Essentially trains a large number of “strong” learners in parallel (each model is an over fit for that subset of the data). Combines (averaging or voting) these learners together to "smooth out" predictions.

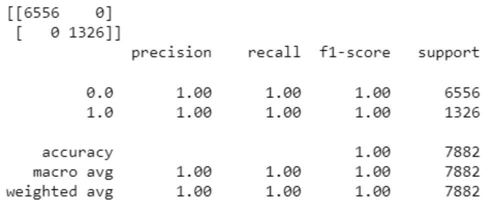
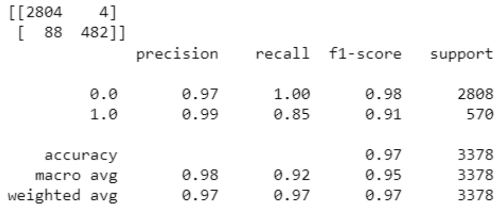
**Random forest** is an ensemble of decision tree algorithms. It is an extension of bootstrap aggregation (bagging) of decision trees and can be used for classification and regression problems. Random forest has yielded excellent results. In the test data, out of 570 churned customers we were able to predict 474 correctly.

**Boosting** trains a large number of "weak" learners in sequence. A weak learner is a simple model that is only slightly better than random (eg. One depth decision tree). Miss-classified data weights are increased for training the next model. So, training has to be done in sequence. Boosting then combines all the weak learners into a single strong learner.

**XGBoost** is a scalable and accurate implementation of gradient boosting machines and it has proven to push the limits of computing power for boosted trees algorithms as it was built and developed for the sole purpose of model performance and computational speed. We have used extreme gradient boosting which has given us the best recall for predicting churn.

b. Any other model tuning measures (if applicable)

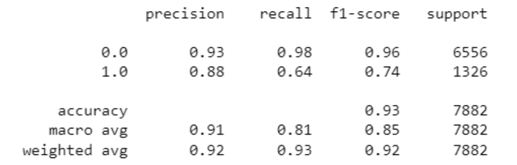
We have used grid search to tune our models. First, we extract GridSearchCV package from sklearn library then we have used it to generate hyper parameters for decision tree, random forest and logistic regression.

| | **train** | **test** |
|---|---|---|
| **random forest(before)** |  |  |

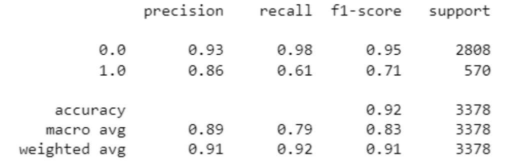
```
[[6556    0]
 [   0 1326]]
              precision    recall  f1-score   support

         0.0       1.00      1.00      1.00      6556
         1.0       1.00      1.00      1.00      1326

    accuracy                           1.00      7882
   macro avg       1.00      1.00      1.00      7882
weighted avg       1.00      1.00      1.00      7882
```

```
[[2804    4]
 [  88  482]]
              precision    recall  f1-score   support

         0.0       0.97      1.00      0.98      2808
         1.0       0.99      0.85      0.91       570

    accuracy                           0.97      3378
   macro avg       0.98      0.92      0.95      3378
weighted avg       0.97      0.97      0.97      3378
```

**random forest (after grid search)**



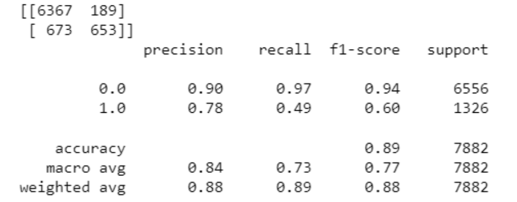
```
              precision    recall  f1-score   support

         0.0       0.93      0.98      0.96      6556
         1.0       0.88      0.64      0.74      1326

    accuracy                           0.93      7882
   macro avg       0.91      0.81      0.85      7882
weighted avg       0.92      0.93      0.92      7882
```



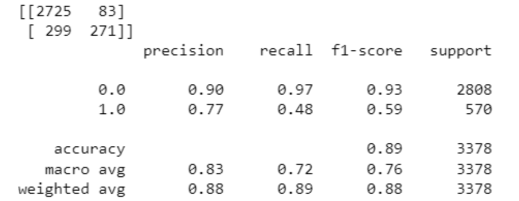
```
              precision    recall  f1-score   support

         0.0       0.93      0.98      0.95      2808
         1.0       0.86      0.61      0.71       570

    accuracy                           0.92      3378
   macro avg       0.89      0.79      0.83      3378
weighted avg       0.91      0.92      0.91      3378
```

Random forest can be said to have overfit before we ran gridsearch. 100% accuracy on training data is not necessarily a problem. We have achieved better results after grid search because training data shows 93% accuracy and test data gave 92% accuracy. However recall is not that great only 64% on train and 61% on test. Hence we can say that the default results are good enough and hyperparameter tuning can't make it better. Precision,AUC and f1 score have reduced after gridsearch.

**Logistic regression(before)**

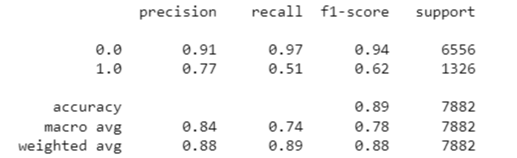


```
[[6367  189]
 [ 673  653]]
              precision    recall  f1-score   support

         0.0       0.90      0.97      0.94      6556
         1.0       0.78      0.49      0.60      1326

    accuracy                           0.89      7882
   macro avg       0.84      0.73      0.77      7882
weighted avg       0.88      0.89      0.88      7882
```

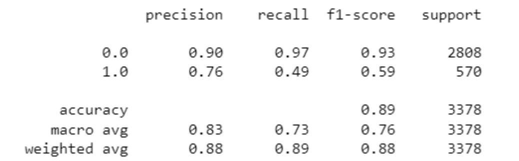


```
[[2725   83]
 [ 299  271]]
              precision    recall  f1-score   support

         0.0       0.90      0.97      0.93      2808
         1.0       0.77      0.48      0.59       570

    accuracy                           0.89      3378
   macro avg       0.83      0.72      0.76      3378
weighted avg       0.88      0.89      0.88      3378
```

**Logistic regression (after grid search)**



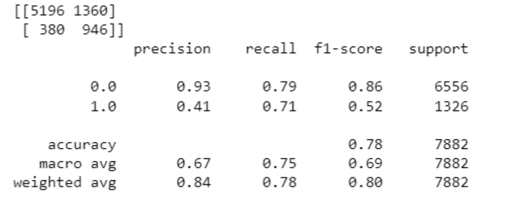
```
              precision    recall  f1-score   support

         0.0       0.91      0.97      0.94      6556
         1.0       0.77      0.51      0.62      1326

    accuracy                           0.89      7882
   macro avg       0.84      0.74      0.78      7882
weighted avg       0.88      0.89      0.88      7882
```



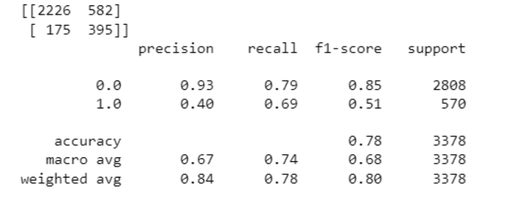
```
              precision    recall  f1-score   support

         0.0       0.90      0.97      0.93      2808
         1.0       0.76      0.49      0.59       570

    accuracy                           0.89      3378
   macro avg       0.83      0.73      0.76      3378
weighted avg       0.88      0.89      0.88      3378
```

Logistic regression is a bad model because though the accuracy before and after grid search is 89%, still recall is only 51% on train and 49% on test even after grid search. But we can agree that grid search has slightly improved the model performance. Precision has reduced after grid search, but f1 score has increased slightly on training data. AUC is same 0.88 before and after gridsearch.

**Naive Bayes(before)**

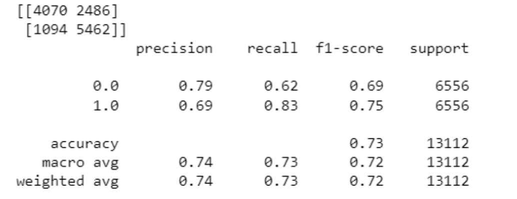


```
[[5196 1360]
 [ 380  946]]
              precision    recall  f1-score   support

         0.0       0.93      0.79      0.86      6556
         1.0       0.41      0.71      0.52      1326

    accuracy                           0.78      7882
   macro avg       0.67      0.75      0.69      7882
weighted avg       0.84      0.78      0.80      7882
```

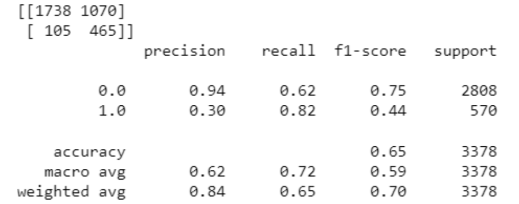


```
[[2226  582]
 [ 175  395]]
              precision    recall  f1-score   support

         0.0       0.93      0.79      0.85      2808
         1.0       0.40      0.69      0.51       570

    accuracy                           0.78      3378
   macro avg       0.67      0.74      0.68      3378
weighted avg       0.84      0.78      0.80      3378
```

**Naive Bayes(before smote)**



```
[[4070 2486]
 [1094 5462]]
              precision    recall  f1-score   support

         0.0       0.79      0.62      0.69      6556
         1.0       0.69      0.83      0.75      6556

    accuracy                           0.73     13112
   macro avg       0.74      0.73      0.72     13112
weighted avg       0.74      0.73      0.72     13112
```



```
[[1738 1070]
 [ 105  465]]
              precision    recall  f1-score   support

         0.0       0.94      0.62      0.75      2808
         1.0       0.30      0.82      0.44       570

    accuracy                           0.65      3378
   macro avg       0.62      0.72      0.59      3378
weighted avg       0.84      0.65      0.70      3378
```

After smote though the accuracy has dropped from 78% to 73% on training data and 78% to 65 % on test data, when it comes to recall we definitely see a improvement from 71% to 83% on training and 69% to 82% on test. Precision has improved on training data hoewver on test data it has decreased. F1 score has improved on training data and descresed on test data. AUC is **0.81** on both training and test

We have used coefficients to analyse Logistic Regression model. Then Top 5 features from Decision Treee model are Tenure, Day_Since_CC_connect, CC_Agent_Score, rev_per_month, Complain_ly, cashback.

c. Interpretation of the most optimum model and its implication on the business

- Our goal was to predict churn, to understand what is driving churn rate and how can we reduce churn to keep the business profitable. After our analysis, we have got 98% accuracy using XGBoost model in predicting churned customers.
- We have also identified the important features affecting churn from decision tree model as Tenure, Day_Since_CC_connect, CC_Agent_Score, rev_per_month, Complain_ly, cashback. As a first step the company can address Tenure of account, Monthly average cashback generated by account in last 12 months, Number of days since no customers in the account has contacted the customer care, Any complaints has been raised by account in last 12 months, Satisfaction score given by customers of the account on customer care service provided by company, revenue growth percentage of the account (last 12 months vs last 24 to 13 month) to ensure that the customer stays with the company.
- Once we are able to identify the customers who might churn, next step would be to retain such customers with more cashbacks, offering easy self-help options, resolving complains to customers satisfaction, deep dive into the revenue growth data for customers, improving customer service.
- Ensuring that the growth rate is more than churn rate, acquiring more customers than customers who have churned are some other ways to solve this business problem.

# THANK YOU !