CAPSTONE PROJECT

ON

Customer churn

SUBMITTED BY: EDIL MONICA S

(BATCH ~ PGPDSBA ONLINE JAN_A 2021)

26th December 2021

CONTENTS

1. Introduction of the Business Problem

Problem Statement

- ❖ The DTH company is facing an issue of customer churn and since there is lot of competition in the current market.
- ❖ The company is concerned where churn of one accounts is leading to loss of many customers.
- ❖ Hence, the company wants to develop a model through which they can do predict churn of the one accounts and provide segmented offers to the potential churners.
- ❖ Acquiring a new customer can cost five times more than retaining an existing customer.
- ❖ Increasing customer retention by 5% can increase profits from 25-95%.
- ❖ The success rate of selling to a customer you already have is 60-70%, while the success rate of selling to a new customer is 5-20%.
- ❖ If they find that we are giving a lot of free (or subsidized) stuff thereby making a loss to the company; they are not going to approve our recommendation.
- ❖ So, we should be very clear and unique while providing campaign recommendation.

Need of the study

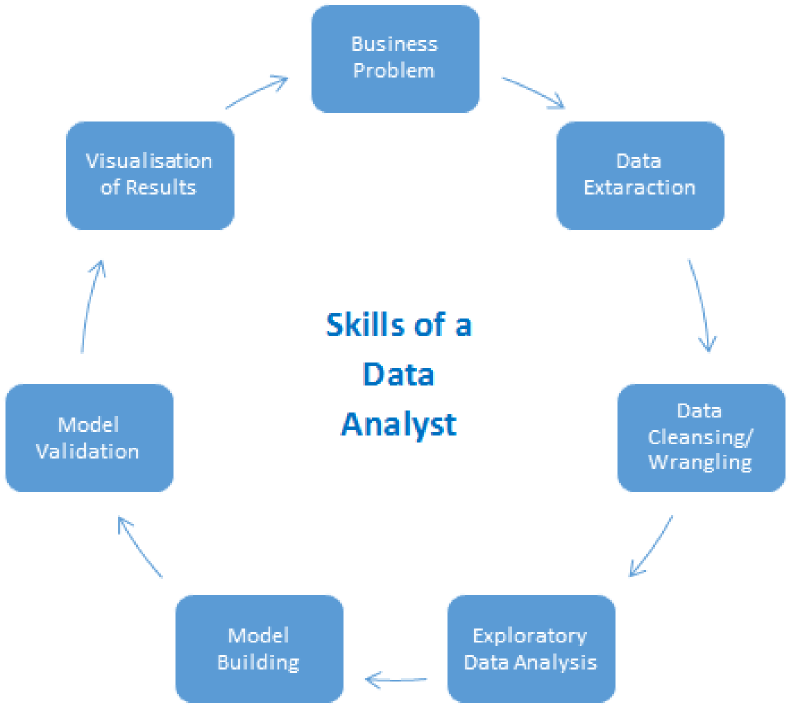
- ❖ Customer churn is an important measure for growth of the business. Churn rate will help in finding out why the customers are switching and fix the issue before it becomes the worse.

- ❖ To find the reason why the customers are getting churned and we to find recommendation to bring back or retain the customers.
- ❖ Non satisfactory customer service, bad product quality, delay in product delivery & services, better subscription rates & offers from the competitors etc. are some reasons behind customer churn.
- ❖ It results in negative impacts on profitability, future business, brand image, market share etc. It is critical issue for a company to ensure that its growth rate is higher than its churn rate otherwise, it will experience declining revenues and profits with the eventual scenario of closing down the business.
- ❖ The purpose of this project is to develop and design an effective and efficient model for customer churn prediction in the company.

Understanding business

- ❖ We have a data of DTH company it carries 19 variables to predict the customer churn rate.
- ❖ Main objective is to find the customer churn rate and provide recommendation to the company.
- ❖ Company wants to decrease the churn rate and build a good relationship with the customers to retain.
- ❖ Need to analyze the drawbacks, get feedback from the customers and rectify it.
- ❖ Provide some benefits to retain the customers and increase company revenue.
- ❖ Need to predict why customers are churned accordingly take action on it.

Our Approach



This is a supervised data and classification type problem.

Data Report

Read the Data

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

Information

```
<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
```

Datatypes

```
object     12
float64     5
int64       2
dtype: int64
```

Count of Target

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

Check duplicate

```
Number of duplicate rows = 0
(11260, 19)
```

Period, frequency and methodology of Data Collection:

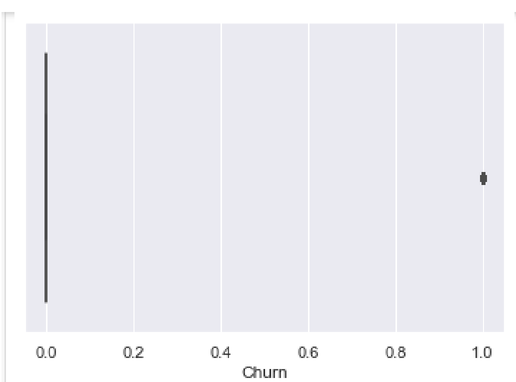
Yearly and monthly aggregation of Information of the DTH Company has been provided against 11260 Account ID’s across 18 different features. It seems the data is Primary Data and manually tabulated from their own database.

Visual inspection of data (rows, columns, descriptive details):
The data consists of churn details of 11260 accounts of DTH Company. Information is provided across 19 columns. Among those, ‘’Churn’’ is our target / dependent variable and the remaining 18 features are predictors / independent variables. We are to build a model, to predict which customer will churn on the basis of the given information. We shall make an end-to-end study on this dataset and prepare the best fitted model/s.

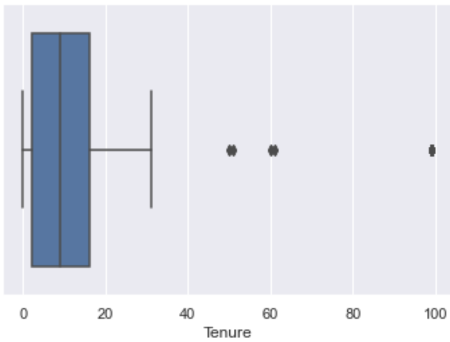
2. EDA and Business Implication

Univariate analysis for continuous variable

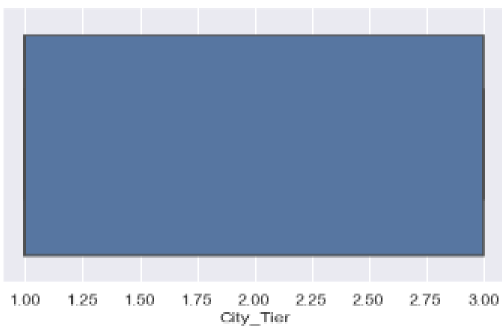
Churn



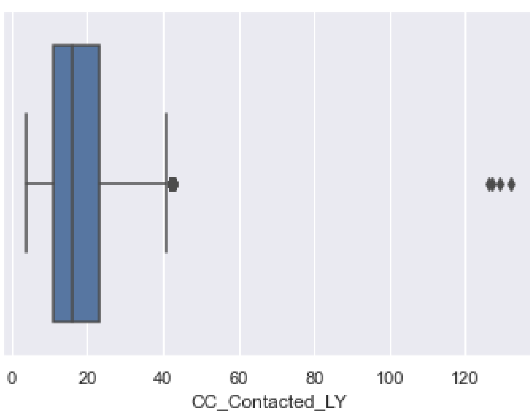
Tenure



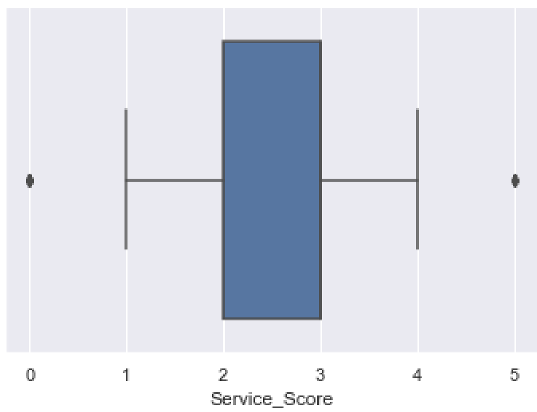
City tier



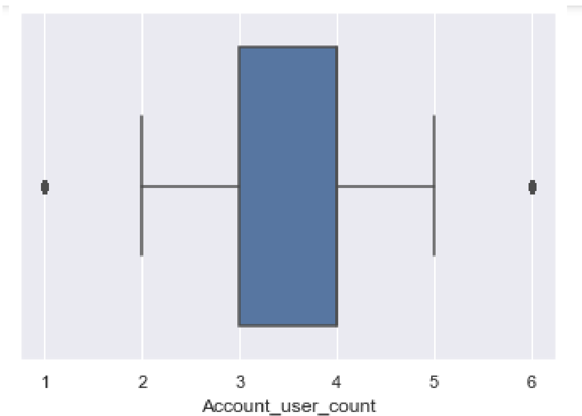
CC contacted Ly



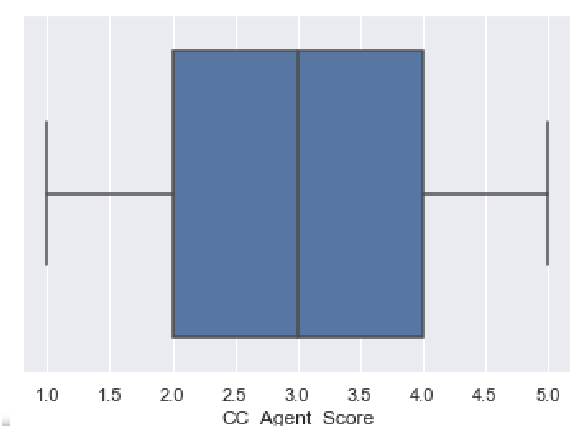
Service store



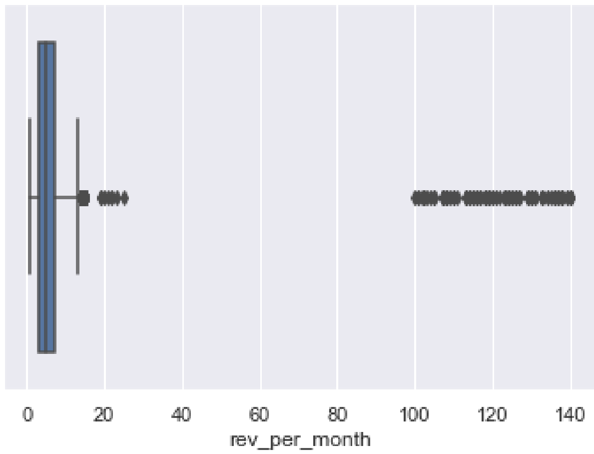
Account user count



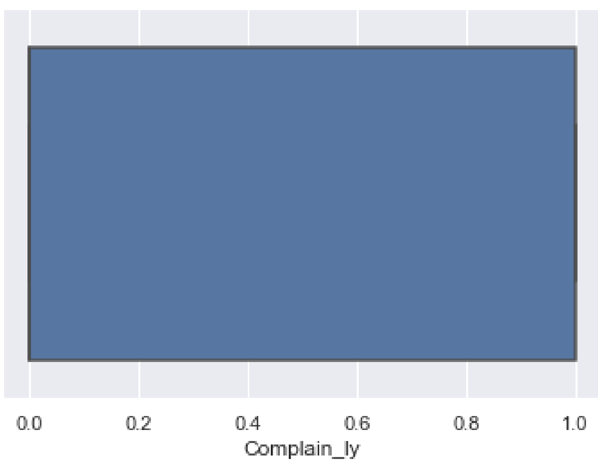
CC Agent score



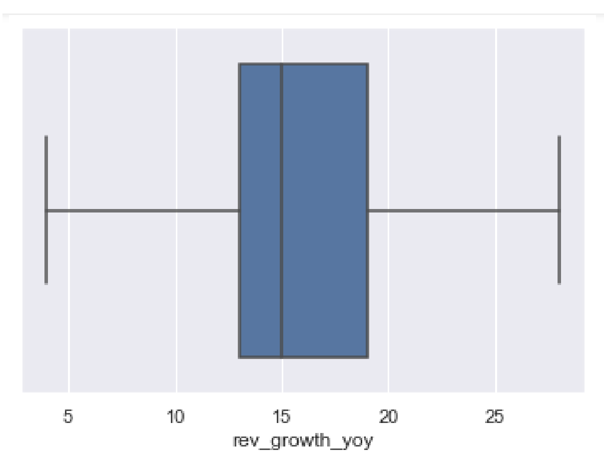
Revenue per month



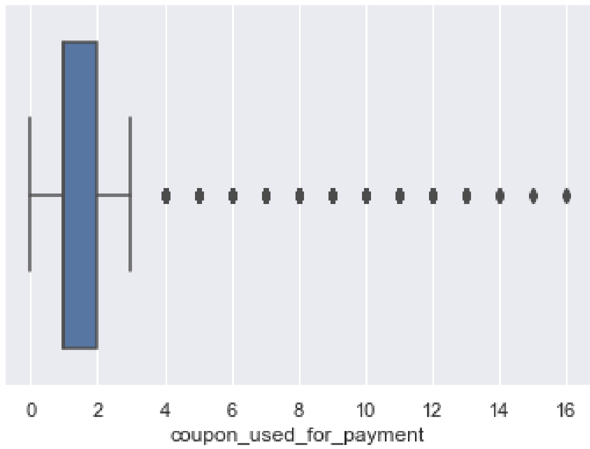
Complain Ly



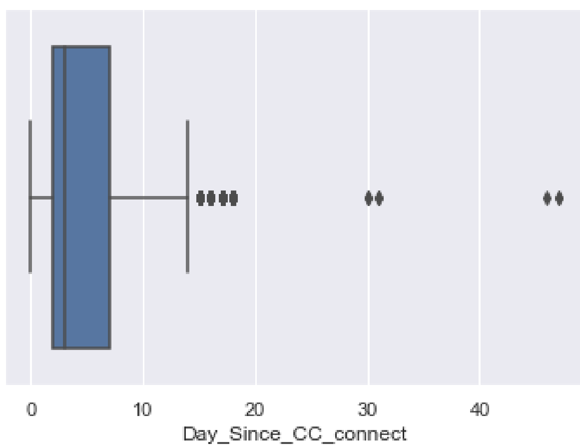
Revenue growth yoy



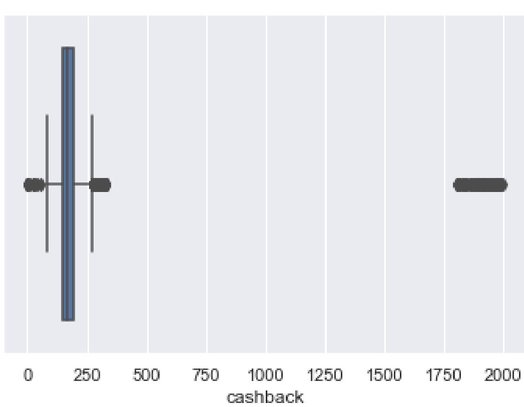
Coupon used for payment



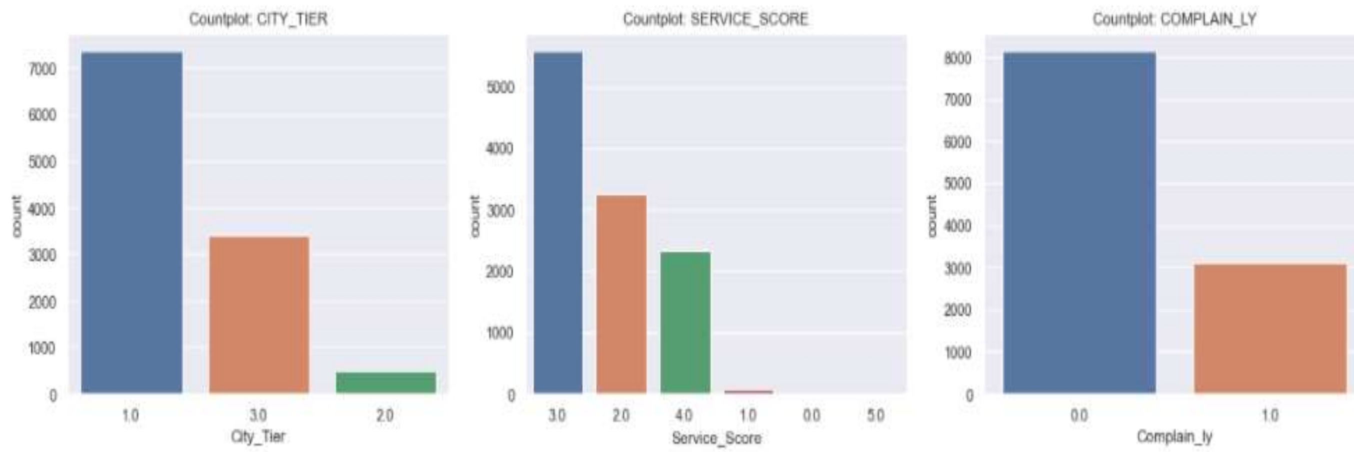
Day since CC connect

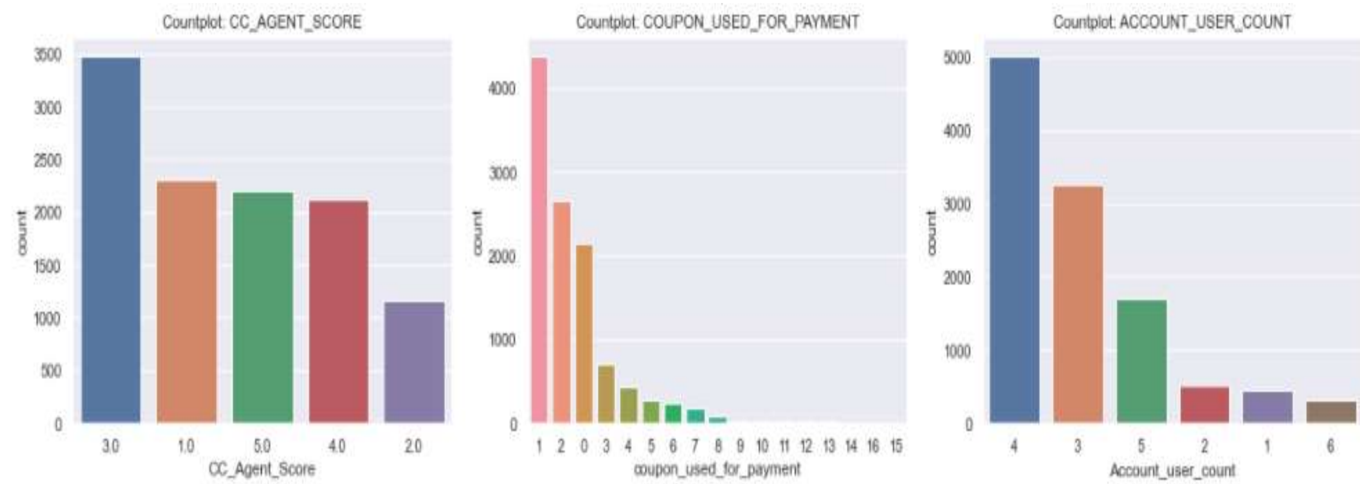


Cashback



Count plot for Numerical variables below

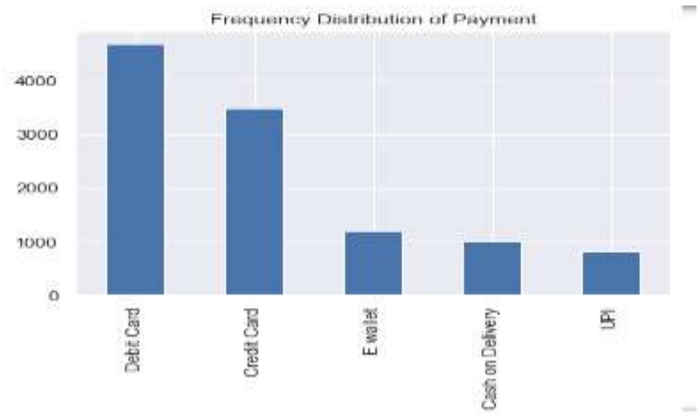




Univariate analysis for categorical variable

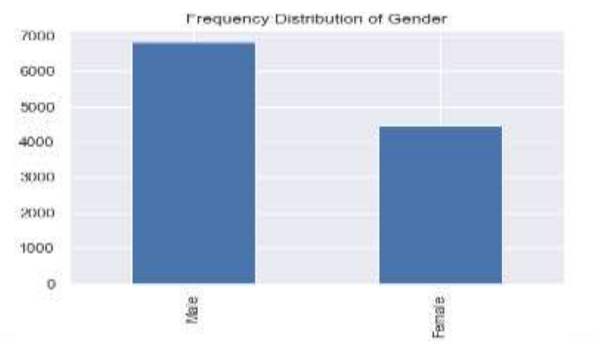
Payment

```
Details of Payment
----------------------------------------
Debit Card          4696
Credit Card         3511
E wallet            1217
Cash on Delivery    1014
UPI                  822
Name: Payment, dtype: int64
```



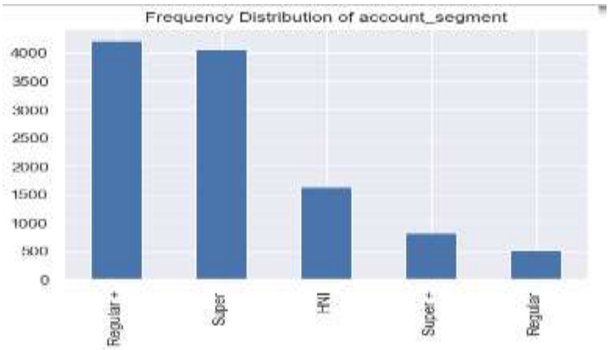
Gender

```
Details of Gender
------------------------------
Male        6812
Female      4448
Name: Gender, dtype: int64
```



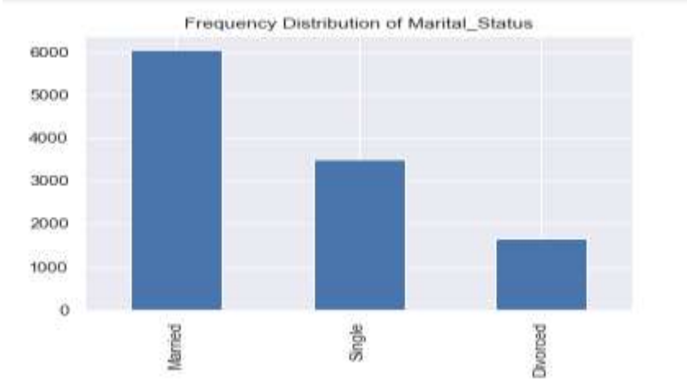
Account segment

```
Details of account_segment
----------------------------------------
Regular +     4221
Super         4062
HNI           1639
Super +        818
Regular        520
Name: account_segment, dtype: int64
```



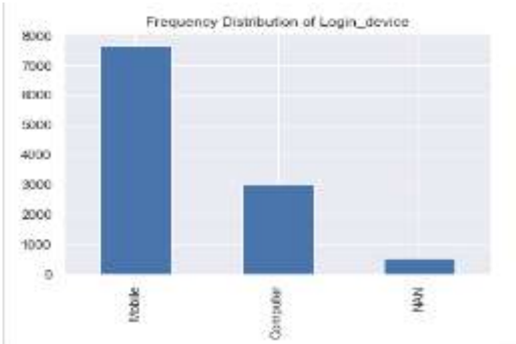
Marital Status

```
Details of Marital_Status
-------------------------------------
Married     6072
Single      3520
Divorced    1668
Name: Marital_Status, dtype: int64
```

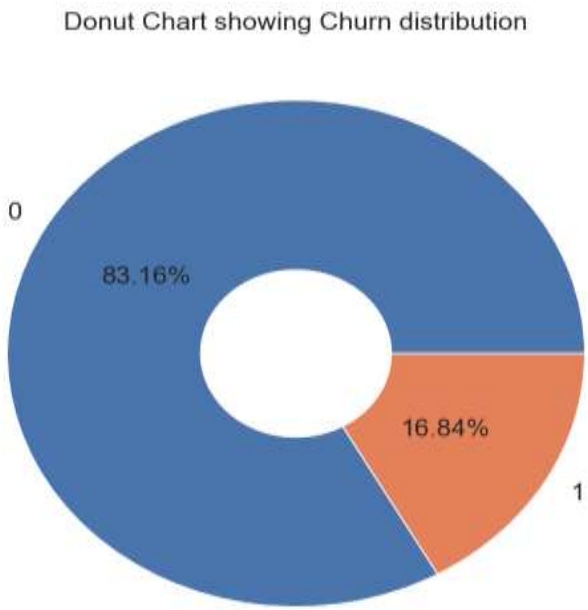


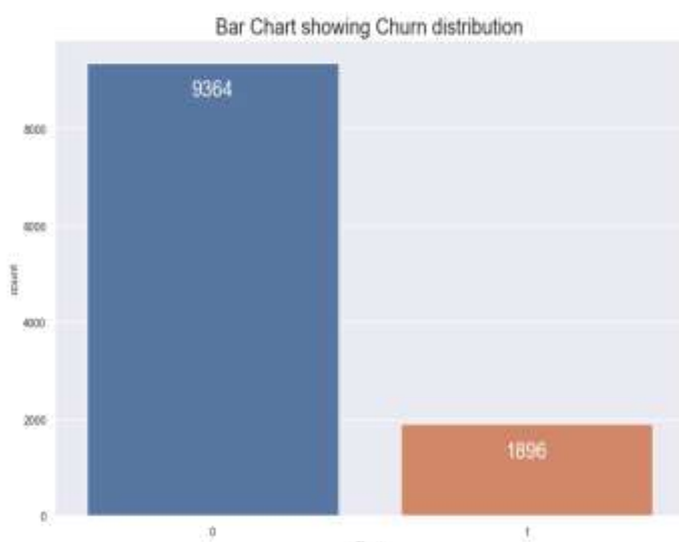
Login device

```
Details of Login_device
-------------------------------------
Mobile      7703
Computer    3018
NAN          539
Name: Login_device, dtype: int64
```



Target variable distribution

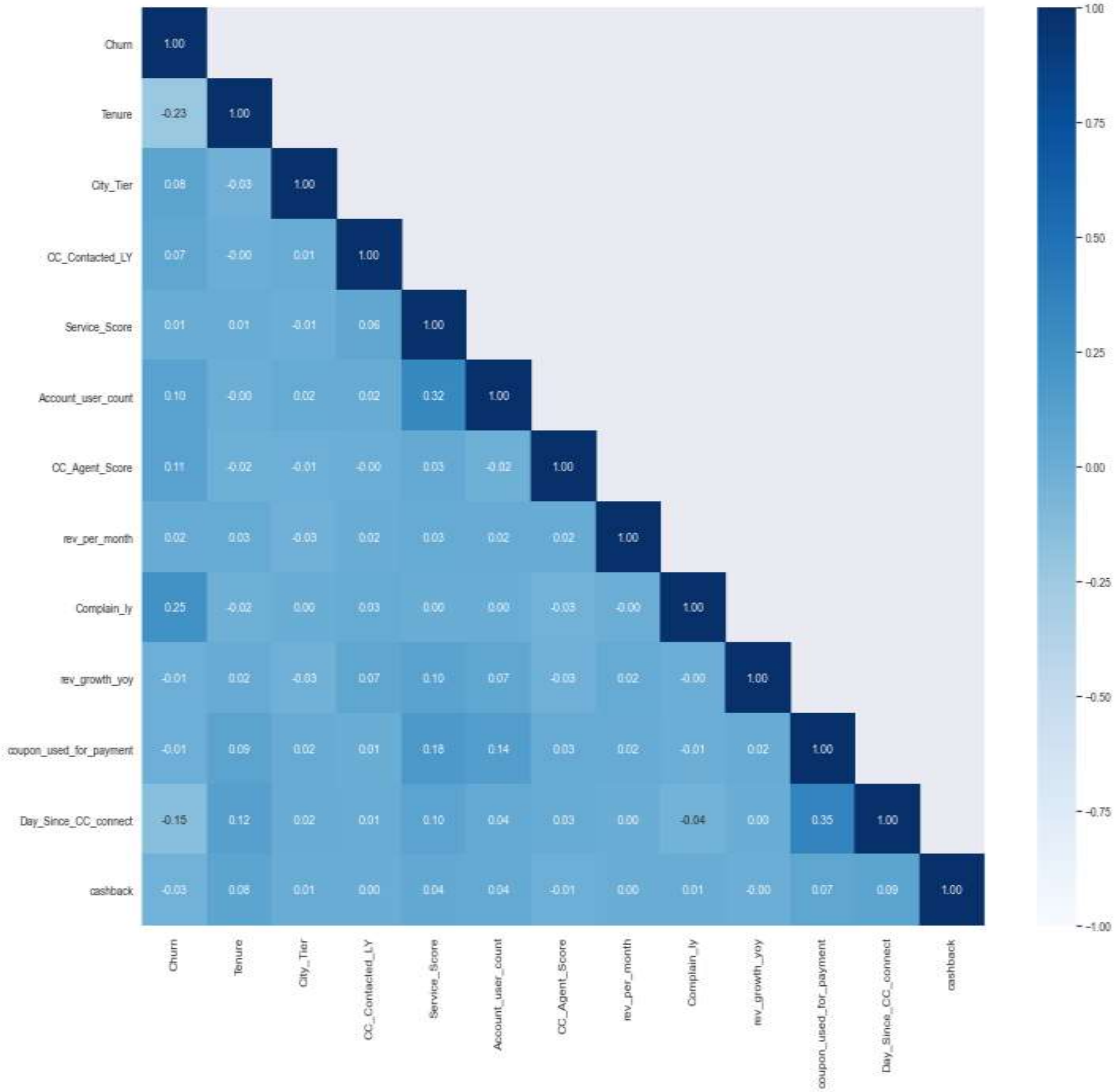




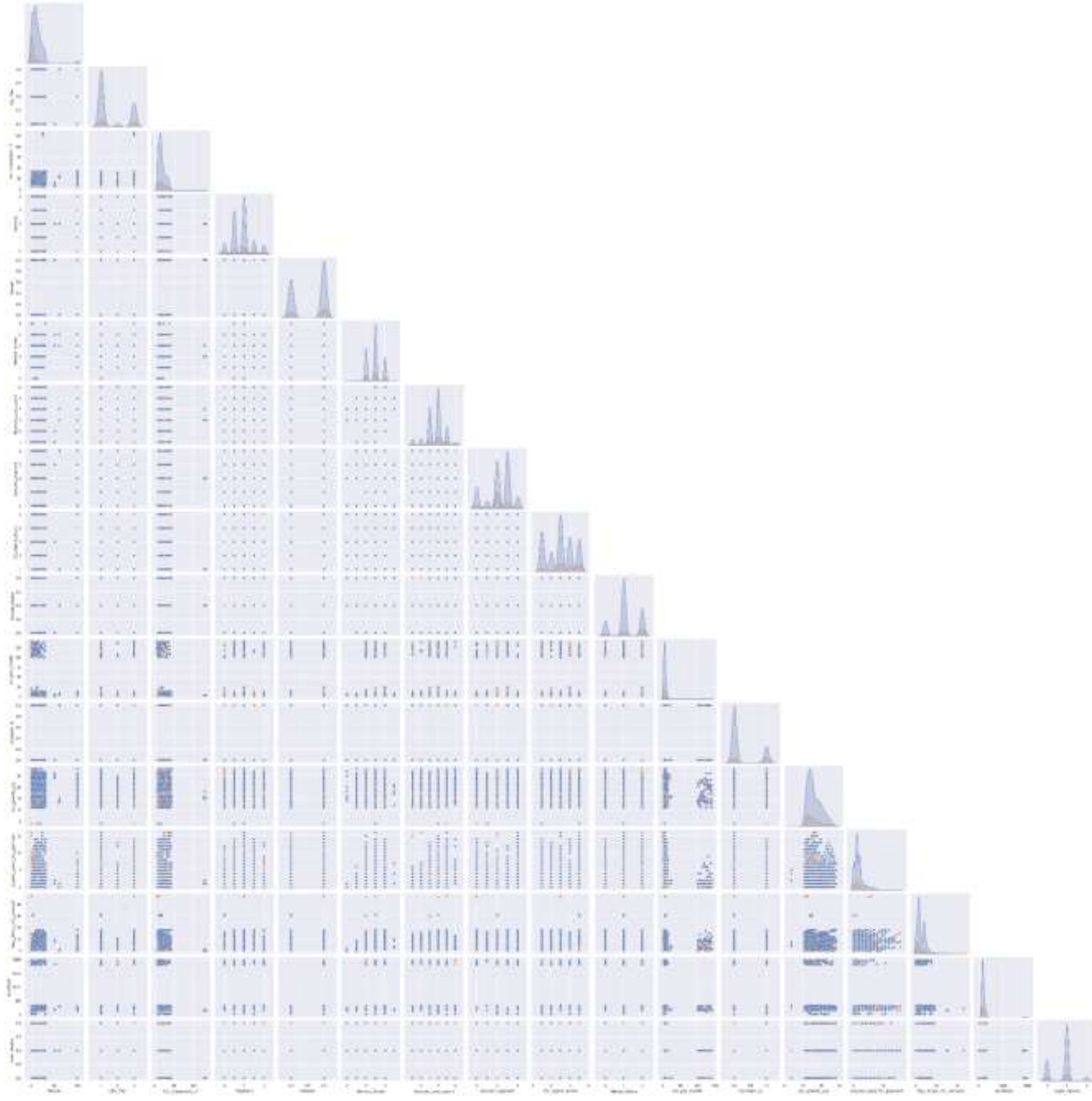
Insights from Univariate Analysis

- There are outliers in the data Churn, Tenure, CC_contacted_Ly, Service_score, Account_user_count, Rev_per_month, Coupon_used_for_payment, Day_since_cc_connect, Cashback.
- Most of the account_user_count are tagged with 4 customers.
- 1 ‘coupon_used_for_payment’ per year is used maximum number of times by the accounts.
- ‘City_Tier’ - 1 is the location with most accounts.
- ‘Service_Score’ awarded by most of the accounts 3
- ‘CC_Agent-Score’ awarded by most of the accounts 3.
- Maximum Cashback awarded to customers in 12 months is Rs.1997.
- Customers who logged in via mobile (7482) are more who have churned.
- Payment of 65 % of the churned account made by Debit & Credit Card.
- Highest customers are male (6704) who are churned compared to female.
- Married customers around (5860) are churned customers when compared to unmarried.
- 4124 customers prefer to regular plus account segment.
- 1896 out of 11260 account holders (16.84 %) churned last year.
- Data provided is very much imbalanced.

Bivariate analysis



Multivariate analysis



- ❖ From Pairplot and Heatmap, it is observed that there is no relationship among the predictors. No multicollinearity is present. All the predictors are independently affecting customer churn.

- The highest correlation is between day since cc connect and coupon used for payment 0.35, 2nd correlation is between account user count and service score 0.31, lowest correlation is between Tenure and churn -0.23.
- 'Churn' does not correlate with other predictors. Slight positive correlation with 'complain_l12m' of 0.25 indicates that churned customers have made more number of complains. In other side, slight negative correlation with 'Tenure' of -0.23 establishes the fact that long term customers are loyal.

3. Data Cleaning and Pre-processing

Value counts for categorical variables

- **Replace the gender variables**

```
Male      6704
Female    4448
Name: Gender, dtype: int64
```

The gender variables had F, M, Female, Male and I have replaced F and M to Male and Female.

- **Payments**

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

- **Replace the account segment variables**

```
Regular +    4124
Super        4062
HNI          1639
Super +       818
Regular       520
Name: account_segment, dtype: int64
```

The account segment variables had repeated variables and merged the Super plus to super+ and Regular plus to Regular +

- **Marital Status**

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

- **Login device**

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

Finding median value to replace with special characters

- Tenure

```
9.0
```

- Account user count

```
4.0
```

- Rev per month

```
5.0
```

- Rev growth yoy

```
15.0
```

- Cashback

```
165.25
```

- Day Since CC connect

```
3.0
```

❖ Coupon used for payment

```
1.0
```

Replaced the special characters

Arrays

❖ Tenure

```
array([ 4.,  0.,  2., 13., 11.,  9., 37., 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., 31.])
```

❖ City Tier

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

❖ CC Contacted LY

```
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.,  7., 36., 38., 37., 39., 40.])
```

❖ Service Score

```
array([3. , 2. , 1. , 0.5, 4. , 4.5])
```

❖ Account user count

```
array([3. , 4. , 5. , 2. , 1.5, 5.5])
```

❖ CC Agent Score

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

❖ Rev per month

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

❖ Complain Ly

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

❖ Day Since CC connect

```
array([ 5. ,  0. ,  3. ,  7. ,  2. ,  1. ,  8. ,  6. ,  4. , 14.5, 11. ,
       10. ,  9. , 13. , 12. , 14. ])
```

❖ Cashback

```
array([159., 120., 165., 134., 129., 139., 122., 126., 273., 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.,
        73., 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., 272., 226.,  81., 251.])
```

❖ Replaced all the special characters with the median values and above are arrays. Now there is no special characters.
❖ All ‘null’ values of categorical variables are imputed by respective column mode.
❖ All ‘null’ values of numerical variables are imputed by respective column median.

Dropped the unwanted column (Account ID)

	Churn	Tenure	City_Tier	CC_Contacted_LY	Payment	Gender	Service_Score	Account_user_count	account_segment	CC_Agent_Score	Marital_Status
0	1	4	3.0	6.0	Debit Card	Female	3.0	3	Super	2.0	Single
1	1	0	1.0	8.0	UPI	Male	3.0	4	Regular +	3.0	Single
2	1	0	1.0	30.0	Debit Card	Male	2.0	4	Regular +	3.0	Single
3	1	0	3.0	15.0	Debit Card	Male	2.0	4	Super	5.0	Single
4	1	0	1.0	12.0	Credit Card	Male	2.0	3	Regular +	5.0	Single

Number of rows and columns

```
(11260, 18)
```

Info

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 11260 entries, 0 to 11259
Data columns (total 18 columns):
 #   Column                   Non-Null Count  Dtype  
---  ------                   --------------  -----  
 0   Churn                    11260 non-null  int64  
 1   Tenure                   11158 non-null  object 
 2   City_Tier                11148 non-null  float64
 3   CC_Contacted_LY          11158 non-null  float64
 4   Payment                  11151 non-null  object 
 5   Gender                   11152 non-null  object 
 6   Service_Score            11162 non-null  float64
 7   Account_user_count       11148 non-null  object 
 8   account_segment          11163 non-null  object 
 9   CC_Agent_Score           11144 non-null  float64
 10  Marital_Status           11048 non-null  object 
 11  rev_per_month            11158 non-null  object 
 12  Complain_ly              10903 non-null  float64
 13  rev_growth_yoy           11260 non-null  object 
 14  coupon_used_for_payment  11260 non-null  object 
 15  Day_Since_CC_connect     10903 non-null  object 
 16  cashback                 10789 non-null  object 
 17  Login_device             11039 non-null  object 
dtypes: float64(5), int64(1), object(12)
memory usage: 1.5+ MB
```

❖ There are 11260 rows and 18 columns after dropping the variables and replacing the special characters.

Count of target variables after dropping Account ID

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

Count of target variables remains same even after dropping a variables.

Percentage of Target variable

```
(0    83.161634
 1    16.838366
 Name: Churn, dtype: float64,
 2)
```

❖ Only 17% ‘Yes’ and 83% ‘No’ so the data is unbalanced. We need to balance in further process of data.

Missing values

```
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
```

❖ There are missing values in the data.

Percentage of missing values

```
Churn                      0.00
Tenure                     0.91
City_Tier                  0.99
CC_Contacted_LY            0.91
Payment                    0.97
Gender                     0.96
Service_Score              0.87
Account_user_count         0.99
account_segment            0.86
CC_Agent_Score             1.03
Marital_Status             1.88
rev_per_month              0.91
Complain_ly                3.17
rev_growth_yoy             0.00
coupon_used_for_payment    0.00
Day_Since_CC_connect       3.17
cashback                   4.18
Login_device               1.96
dtype: float64
```

❖ Since the missing values are not more than 4 percentage we do not need to drop the missing values.

Imputing with median for numerical data

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

- ❖ Imputed the null values with median and now there is no null values.

Datatypes after converting datatypes

```
Churn                        int64
Tenure                       int32
City_Tier                  float64
CC_Contacted_LY            float64
Payment                     object
Gender                      object
Service_Score              float64
Account_user_count           int32
account_segment             object
CC_Agent_Score             float64
Marital_Status              object
rev_per_month                int32
Complain_ly                float64
rev_growth_yoy               int32
coupon_used_for_payment      int32
Day_Since_CC_connect         int32
cashback                     int32
Login_device                object
dtype: object
```

```
int32      7
float64    5
object     5
int64      1
dtype: int64
```

Description of the data

	count	unique	top	freq	mean	std	min	25%	50%	75%	max
Churn	11260	NaN	NaN	NaN	0.168384	0.374223	0	0	0	0	1
Tenure	11260	NaN	NaN	NaN	10.9859	12.7575	0	2	9	16	99
City_Tier	11260	NaN	NaN	NaN	1.64742	0.912763	1	1	1	3	3
CC_Contacted_LY	11260	NaN	NaN	NaN	17.8502	8.81485	4	11	16	23	132
Payment	11260	5	Debit Card	4696	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Gender	11260	2	Male	6812	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Service_Score	11260	NaN	NaN	NaN	2.90337	0.722476	0	2	3	3	5
Account_user_count	11260	NaN	NaN	NaN	3.70497	1.00438	1	3	4	4	6
account_segment	11260	5	Regular +	4221	NaN	NaN	NaN	NaN	NaN	NaN	NaN
CC_Agent_Score	11260	NaN	NaN	NaN	3.06581	1.37266	1	2	3	4	5
Marital_Status	11260	3	Married	6072	NaN	NaN	NaN	NaN	NaN	NaN	NaN
rev_per_month	11260	NaN	NaN	NaN	6.26687	11.489	1	3	5	7	140
Complain_ly	11260	NaN	NaN	NaN	0.276288	0.447181	0	0	0	1	1
rev_growth_yoy	11260	NaN	NaN	NaN	16.1931	3.75727	4	13	15	19	28
coupon_used_for_payment	11260	NaN	NaN	NaN	1.79041	1.96933	0	1	1	2	16
Day_Since_CC_connect	11260	NaN	NaN	NaN	4.58126	3.64964	0	2	3	7	47
cashback	11260	NaN	NaN	NaN	194.459	175.023	0	147	165	197	1997
Login_device	11260	3	Mobile	7703	NaN	NaN	NaN	NaN	NaN	NaN	NaN

Count of Top Frequency variables as per the descriptive data.

	Count	Unique	Top	Frequency
Payment	11260	5	Debit Card	4696
Gender	11260	2	Male	6812
Account Segment	11260	5	Regular Plus	4221
Marital Status	11260	3	Married	6072
Login device	11260	3	Mobile	7703

Check Outliers

Outliers in numbers

```
coupon_used_for_payment    1380
cashback                    965
Account_user_count          761
rev_per_month               185
Tenure                      139
Day_Since_CC_connect        130
CC_Contacted_LY              42
Service_Score                13
account_segment               0
Payment                       0
rev_growth_yoy                0
Login_device                  0
Gender                        0
Complain_ly                   0
City_Tier                     0
CC_Agent_Score                0
Marital_Status                0
dtype: int64
```

Outliers in percentage

```
coupon_used_for_payment    12.255773
cashback                    8.570160
Account_user_count          6.758437
rev_per_month               1.642984
Tenure                      1.234458
Day_Since_CC_connect        1.154529
CC_Contacted_LY             0.373002
Service_Score               0.115453
account_segment             0.000000
Payment                     0.000000
rev_growth_yoy              0.000000
Login_device                0.000000
Gender                      0.000000
Complain_ly                 0.000000
City_Tier                   0.000000
CC_Agent_Score              0.000000
Marital_Status              0.000000
dtype: float64
```

- ❖ The maximum outliers present as 12.26 % in 'coupon_used_for_payment', followed by 8.76 % in 'cashback' and 6.76 in 'Account_user_count' which are genuine in nature for HNI customers. Moreover, all the features except 'coupon_used_for_payment' are with less than 10 % outliers.

- Hence, we will keep those entries without any further manipulation and use it as final dataset for model building. Again, classification models are not sensitive to outliers.

4. Model building

Descriptive Model for identifying important predictors:

We should build the model on the basis of important predictors only. To find those, we shall first check presence of multicollinearity in the dataset by calling Variance Inflation Factor (VIF).

	variables	VIF
10	rev_per_month	1.300422
11	Complain_ly	1.379147
0	Tenure	1.776579
13	coupon_used_for_payment	2.168521
15	cashback	2.266142
4	Gender	2.472687
14	Day_Since_CC_connect	2.962721
16	Login_device	3.171795
9	Marital_Status	3.919705
3	Payment	4.222503
1	City_Tier	4.432108
7	account_segment	4.519479
2	CC_Contacted_LY	4.988601
8	CC_Agent_Score	5.496712
6	Account_user_count	15.050329
12	rev_growth_yoy	15.651443
5	Service_Score	17.854014

Here we are finding VIF high in many variables so we are going to drop variables whose correlation is above 10.

With the remaining predictors, we have applied logit function to check the insignificant features. On first iteration, the summary is:

```
Optimization terminated successfully.
         Current function value: 0.341611
         Iterations 7
```

Logit Regression Results

Dep. Variable:	Churn	No. Observations:	11260
Model:	Logit	Df Residuals:	11246
Method:	MLE	Df Model:	13
Date:	Wed, 26 Jan 2022	Pseudo R-squ.:	0.2464
Time:	12:02:29	Log-Likelihood:	-3846.5
converged:	True	LL-Null:	-5104.3
Covariance Type:	nonrobust	LLR p-value:	0.000

	coef	std err	z	P>|z|	[0.025	0.975]
Intercept	-2.3621	0.163	-14.465	0.000	-2.682	-2.042
Tenure	-0.1307	0.005	-25.810	0.000	-0.141	-0.121
CC_Contacted_LY	0.0256	0.003	7.924	0.000	0.019	0.032
Payment	-0.0015	0.028	-0.054	0.957	-0.057	0.054
Gender	0.2626	0.061	4.341	0.000	0.144	0.381
account_segment	-0.2856	0.029	-9.977	0.000	-0.342	-0.230
CC_Agent_Score	0.2642	0.022	12.284	0.000	0.222	0.306
Marital_Status	0.5725	0.045	12.655	0.000	0.484	0.661
rev_per_month	0.0075	0.002	3.381	0.001	0.003	0.012
Complain_ly	1.4836	0.060	24.809	0.000	1.366	1.601
coupon_used_for_payment	0.1381	0.017	8.338	0.000	0.106	0.171
Day_Since_CC_connect	-0.1219	0.010	-11.628	0.000	-0.142	-0.101
cashback	2.583e-05	0.000	0.157	0.875	-0.000	0.000
Login_device	-0.2743	0.056	-4.912	0.000	-0.384	-0.165

P value is more than 0.05 for

i.e. ‘Payment’ 0.957 and ‘cashback_l12m’ 0.875 & hence removed.

On second iteration on removal of above mentioned two variables, the summary is:

```
Optimization terminated successfully.
         Current function value: 0.341613
         Iterations 7
```

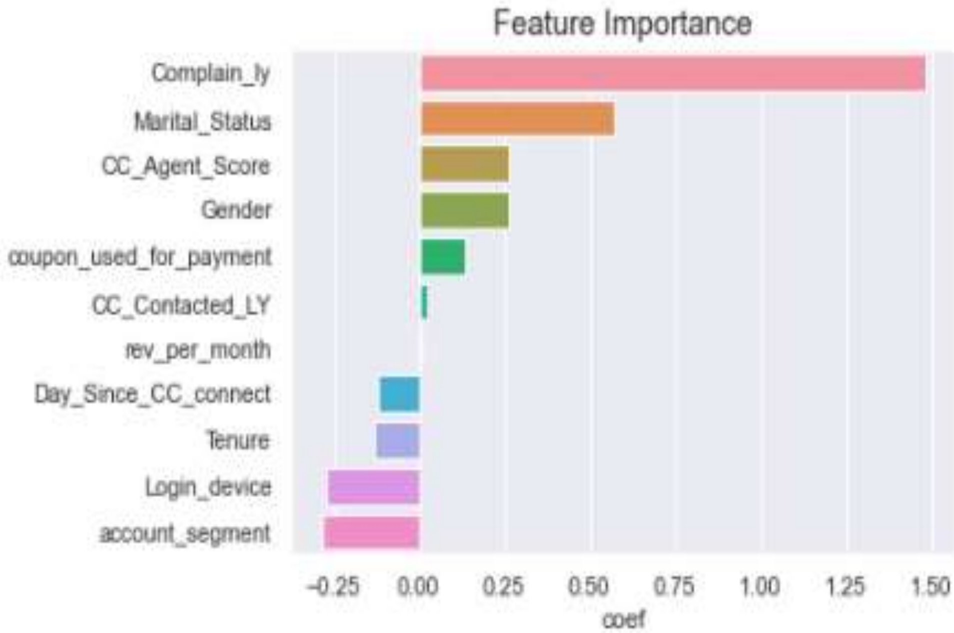
Logit Regression Results

Dep. Variable:	Churn	No. Observations:	11260
Model:	Logit	Df Residuals:	11248
Method:	MLE	Df Model:	11
Date:	Wed, 26 Jan 2022	Pseudo R-squ.:	0.2464
Time:	12:02:29	Log-Likelihood:	-3846.6
converged:	True	LL-Null:	-5104.3
Covariance Type:	nonrobust	LLR p-value:	0.000

	coef	std err	z	P>|z|	[0.025	0.975]
Intercept	-2.3606	0.155	-15.269	0.000	-2.664	-2.058
Tenure	-0.1307	0.005	-25.862	0.000	-0.141	-0.121
CC_Contacted_LY	0.0256	0.003	7.925	0.000	0.019	0.032
Gender	0.2633	0.060	4.365	0.000	0.145	0.382
account_segment	-0.2858	0.029	-9.993	0.000	-0.342	-0.230
CC_Agent_Score	0.2642	0.022	12.284	0.000	0.222	0.306
Marital_Status	0.5725	0.045	12.658	0.000	0.484	0.661
rev_per_month	0.0075	0.002	3.381	0.001	0.003	0.012
Complain_ly	1.4836	0.060	24.841	0.000	1.367	1.601
coupon_used_for_payment	0.1382	0.017	8.351	0.000	0.106	0.171
Day_Since_CC_connect	-0.1218	0.010	-11.629	0.000	-0.142	-0.101
Login_device	-0.2743	0.056	-4.911	0.000	-0.384	-0.165

These are the significant predictors, on which we shall build our models.

Feature Importance



- ❖ Removed unnecessary features for model building to arrive the final dataset.

Model building and interpretation

However, the accuracy measure is not sufficient for imbalanced data. Basically, for predicting positive (As 1) class, recall is the most efficient measure. But for business perspective, it is also important for the company to predict the customers who will not churn (negative class as 0) for strategic decisions. So, precision also has its own impact. Hence, we shall consider F1 score (which balances both recall and precision) as our optimum measure. We will also examine AUC measure.

First, we have defined two objects X and y as independent and dependent variables respectively. In X, we have taken all independent variables (predictors) other than the target variable ‘Churn’. The object y consists of only the values of the column

‘Churn’, our target variable. So, X is a matrix of order 11260 X 12 and y is a column matrix / vector matrix of order 11260.

Then we split the data frame into testing & training data by calling ‘train_test_split’ function from sklearn.model_selection module. We have chosen the ratio of training & testing dataset as 70:30 with ‘stratify’ function to keep proportion of target feature at a same level for both train and test dataset. It returns four subsets containing training independent, testing independent, training dependent & testing dependent variables respectively. We have got the following data frames under allotted object name & sizes:

Name	**Size**	**Description**
X_train	(7882, 12)	Training data frame of independent variables
X_test	(3378, 12)	Testing data frame of independent variables
train_labels	(7882,)	Training data frame of dependent variables
test_labels	(3378,)	Testing data frame of dependent variables

The following classification models will be built and validated on these training and testing data frames. On the basis of their performances, we compare and choose the best model.

1. Decision Tree
2. Random Forest
3. Artificial Neural Network
4. Logistic Regression
5. KNN
6. Ensemble – Bagging

7. Ensemble – Ada Boosting

8. Ensemble – Gradient Boosting

Data's in X matrix

	Tenure	City_Tier	CC_Contacted_LY	Gender	account_segment	CC_Agent_Score	Marital_Status	rev_per_month	Complain_ly	coupon_used_for_payment
0	4	3.0	6.0	0	3	2.0	2	9	1.0	1
1	0	1.0	8.0	1	2	3.0	2	7	1.0	0
2	0	1.0	30.0	1	2	3.0	2	6	1.0	0
3	0	3.0	15.0	1	3	5.0	2	8	0.0	0
4	0	1.0	12.0	1	2	5.0	2	3	0.0	1

1. Decision Tree

DT Model Performance Evaluation on Training data

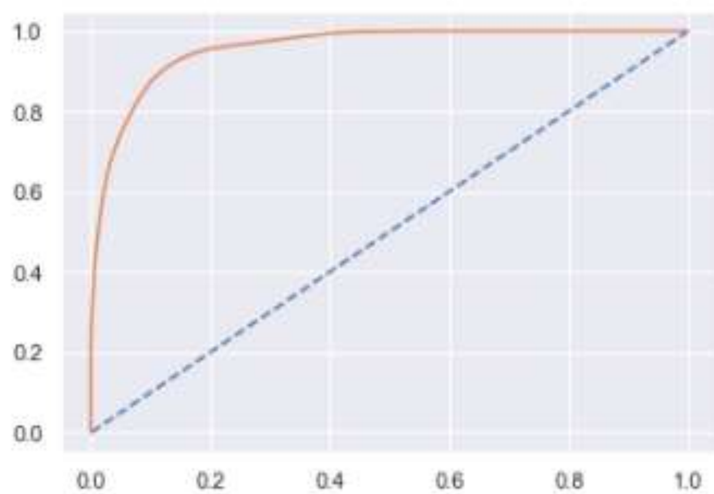
Accuracy for Training data

```
0.9180411063181934
```

AUC

```
AUC: 0.957
```

```
[<matplotlib.lines.Line2D at 0x187d4a4f490>]
```



Classification report

```
              precision    recall  f1-score   support

           0       0.94      0.97      0.95      6556
           1       0.80      0.68      0.74      1326

    accuracy                           0.92      7882
   macro avg       0.87      0.82      0.84      7882
weighted avg       0.91      0.92      0.92      7882
```

Confusion Matrix

```
array([[6338,  218],
       [ 428,  898]], dtype=int64)
```

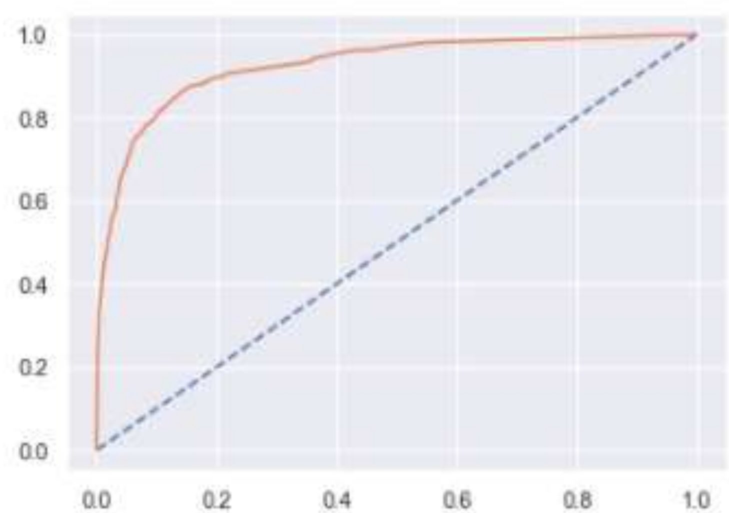
DT Model Performance Evaluation on Test data

Accuracy for Test data

```
0.9076376554174067
```

AUC

```
AUC: 0.926

[<matplotlib.lines.Line2D at 0x187d4a9a730>]
```



Classification report

```
              precision    recall  f1-score   support

           0       0.93      0.96      0.95      2808
           1       0.78      0.64      0.70       570

    accuracy                           0.91      3378
   macro avg       0.85      0.80      0.82      3378
weighted avg       0.90      0.91      0.90      3378
```

Confusion Matrix

```
array([[2703,  105],
       [ 207,  363]], dtype=int64)
```

2. Random Forest Classifier

Grid search

```
GridSearchCV(cv=10, estimator=RandomForestClassifier(),
             param_grid={'max_depth': [20, 40, 60], 'max_features': [10, 20],
                         'min_samples_leaf': [5, 10],
                         'min_samples_split': [50, 100],
                         'n_estimators': [500, 700, 1000]})
```

Best parameter

```
{'max_depth': 60,
 'max_features': 10,
 'min_samples_leaf': 10,
 'min_samples_split': 50,
 'n_estimators': 1000}
```

Best grid

```
RandomForestClassifier(max_depth=60, max_features=10, min_samples_leaf=10,
                       min_samples_split=50, n_estimators=1000)
```

RF Model Performance Evaluation on Training data

Accuracy for Training data

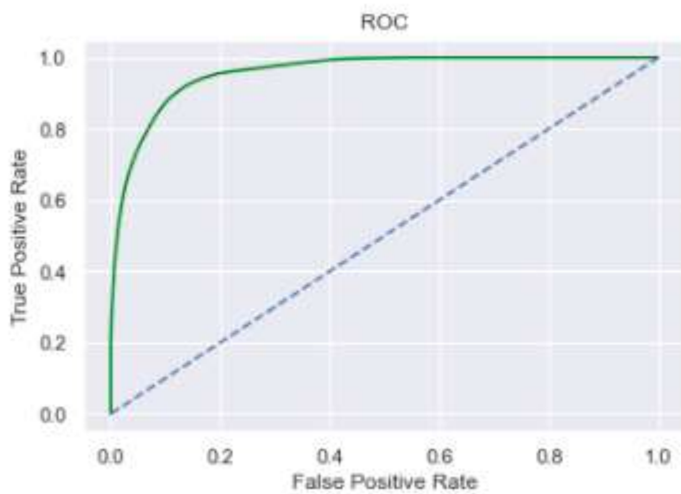
```
0.9200710479573713
```

Classification report

```
              precision    recall  f1-score   support

           0       0.93      0.97      0.95      6556
           1       0.83      0.66      0.74      1326

    accuracy                           0.92      7882
   macro avg       0.88      0.82      0.84      7882
weighted avg       0.92      0.92      0.92      7882
```

AUC

```
Area under Curve is 0.9574704230497756
```



Confusion Matrix

```
array([[6372,  184],
       [ 446,  880]], dtype=int64)
```

RF Model Performance Evaluation on Test data

Accuracy for Test data

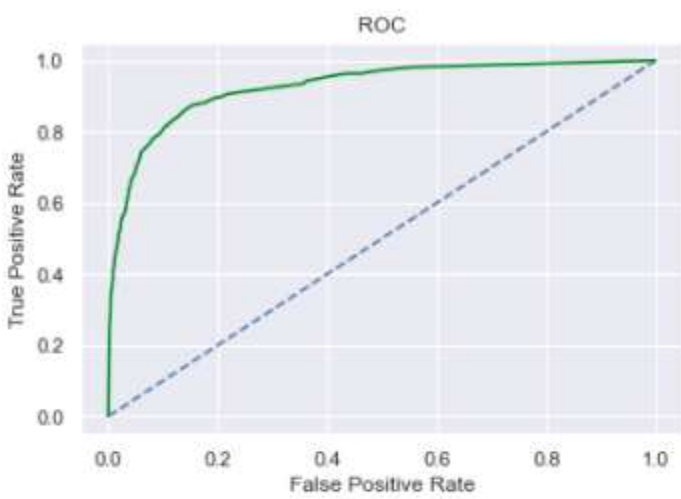
```
0.9126702190645353
```

Classification report

```
              precision    recall  f1-score   support

           0       0.93      0.97      0.95      2808
           1       0.81      0.62      0.71       570

    accuracy                           0.91      3378
   macro avg       0.87      0.80      0.83      3378
weighted avg       0.91      0.91      0.91      3378
```

AUC

```
Area under Curve is 0.9261642800019994
```



Confusion Matrix

```
array([[2727,   81],
       [ 214,  356]], dtype=int64)
```

3. Artificial Neural Network

Best grid

```
MLPClassifier(hidden_layer_sizes=1000, max_iter=500, random_state=1, tol=0.001)
```

NN Model Performance Evaluation on Training data

Accuracy for Training data

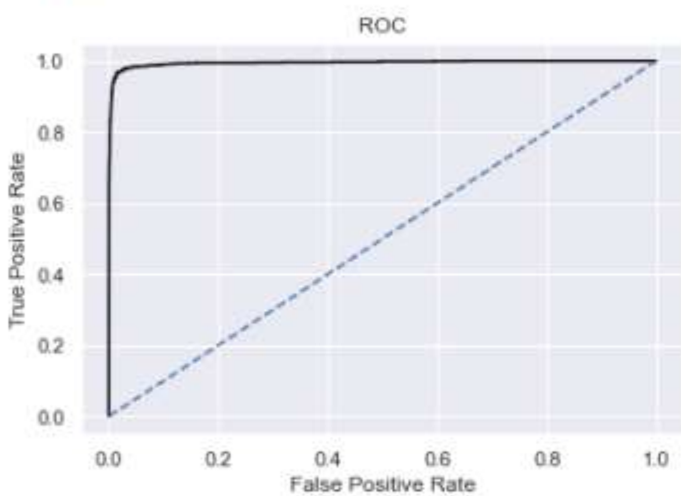
```
0.9743719868053794
```

Classification report

```
              precision    recall  f1-score   support

           0       0.97      1.00      0.98      6556
           1       0.98      0.87      0.92      1326

    accuracy                           0.97      7882
   macro avg       0.98      0.93      0.95      7882
weighted avg       0.97      0.97      0.97      7882
```

AUC

```
Area under Curve is 0.9949019676862156
```



Confusion matrix

```
array([[6531,   25],
       [ 177, 1149]], dtype=int64)
```

NN Model Performance Evaluation on Test data

Accuracy for Test data

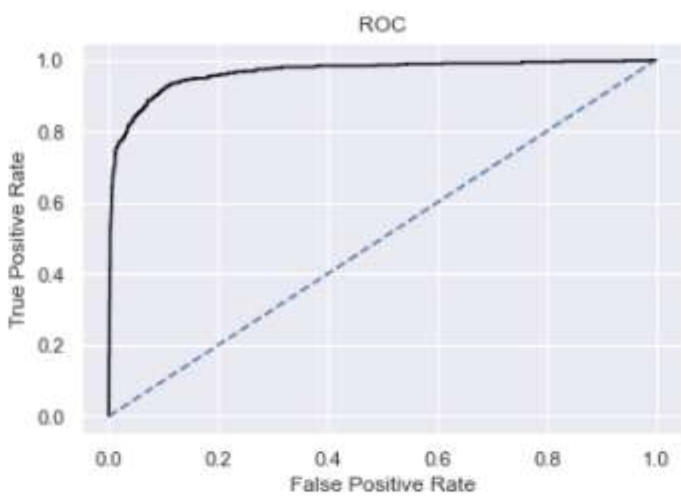
```
0.9458259325044405
```

Classification report

```
              precision    recall  f1-score   support

           0       0.95      0.99      0.97      2808
           1       0.92      0.74      0.82       570

    accuracy                           0.95      3378
   macro avg       0.94      0.86      0.90      3378
weighted avg       0.94      0.95      0.94      3378
```

AUC

```
Area under Curve is 0.9261642800019994
```



Confusion Matrix

```
array([[2772,   36],
       [ 147,  423]], dtype=int64)
```

4. Logistic Regression

Y test predict probability

	0	1
0	0.361945	0.638055
1	0.417063	0.582937
2	0.929845	0.070155
3	0.711698	0.288302
4	0.958776	0.041224

LR model Performance Evaluation on Train data

Accuracy for Training data

```
0.8742704897234205
```

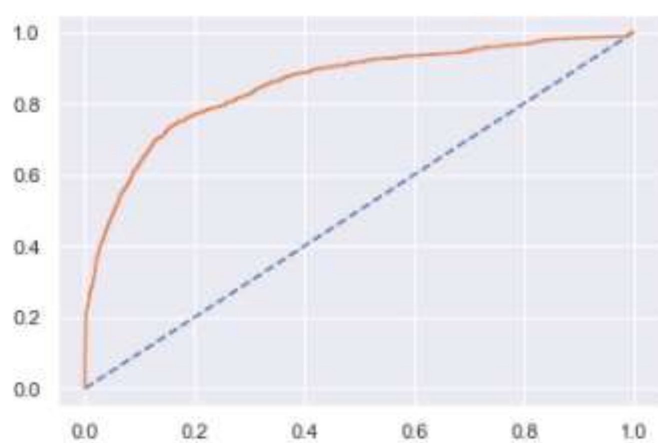
Classification report

	precision	recall	f1-score	support
0	0.94	0.85	0.89	6556
1	0.49	0.72	0.59	1326
accuracy			0.83	7882
macro avg	0.72	0.79	0.74	7882
weighted avg	0.86	0.83	0.84	7882

AUC

```
AUC: 0.853
```

```
[<matplotlib.lines.Line2D at 0x187c4f13fd0>]
```



Confusion Matrix

```
array([[5567,  989],
       [ 365,  961]], dtype=int64)
```

LR model Performance Evaluation on Test data

Accuracy for Test data

```
0.8747779751332149
```

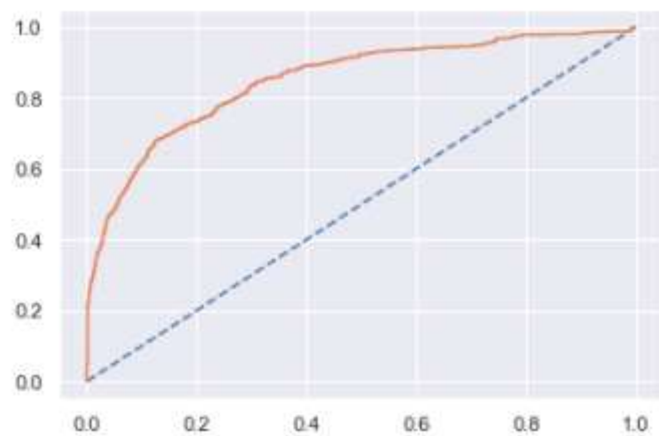
Classification report

	precision	recall	f1-score	support
0	0.93	0.85	0.89	2808
1	0.48	0.70	0.57	570
accuracy			0.82	3378
macro avg	0.71	0.77	0.73	3378
weighted avg	0.86	0.82	0.83	3378

AUC

```
AUC: 0.851
```

```
[<matplotlib.lines.Line2D at 0x187c8030910>]
```



Confusion Matrix

```
array([[2378,  430],
       [ 172,  398]], dtype=int64)
```

5. KNN Model

KNN model score

```
0.9341537680791677
```

Performance Matrix on train data

Accuracy for Training data

```
0.9645
```

Classification report

```
              precision    recall  f1-score   support

           0       0.95      0.98      0.96      6556
           1       0.86      0.72      0.79      1326

    accuracy                           0.93      7882
   macro avg       0.90      0.85      0.87      7882
weighted avg       0.93      0.93      0.93      7882
```

Confusion Matrix

```
[[6403  153]
 [ 366  960]]
```

Performance Matrix on test data set

Accuracy for Test data

```
0.9361
```

Classification report

```
              precision    recall  f1-score   support

           0       0.92      0.96      0.94      2808
           1       0.73      0.59      0.65       570

    accuracy                           0.89      3378
   macro avg       0.82      0.77      0.79      3378
weighted avg       0.89      0.89      0.89      3378
```

Confusion Matrix

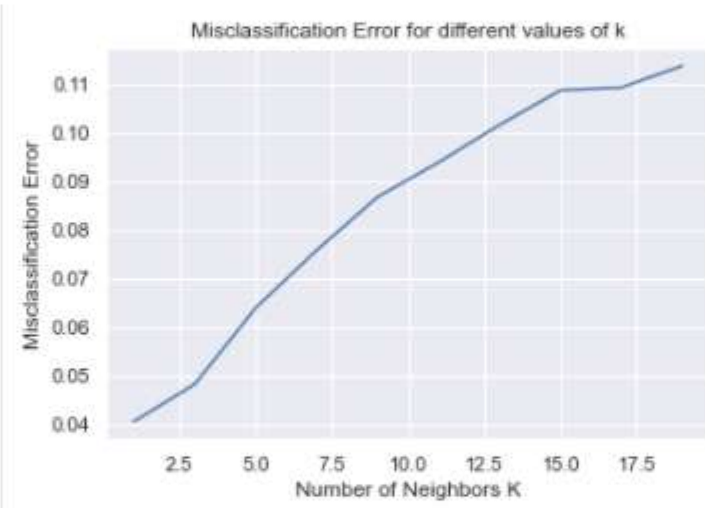
```
[[2683  125]
 [ 234  336]]
```

Let us find optimum value of k

Ac scores

```
[0.04055654233274131,
 0.048253404381290665,
 0.06394316163410307,
 0.07578448786264058,
 0.0867377146240379,
 0.09384251036116042,
 0.101539372409709,
 0.10864416814683242,
 0.10923623445825936,
 0.11367673179396087]
```

Misclassification Error



In K=3 it is giving the best accuracy now let’s check on train and test data.

Accuracy for Training data

```
0.9787
```

Classification report

```
              precision    recall  f1-score   support

           0       0.99      0.99      0.99      6556
           1       0.95      0.93      0.94      1326

    accuracy                           0.98      7882
   macro avg       0.97      0.96      0.96      7882
weighted avg       0.98      0.98      0.98      7882
```

Confusion Matrix

```
[[6486   70]
 [  98 1228]]
```

Accuracy for Test data

```
0.9517
```

Classification report

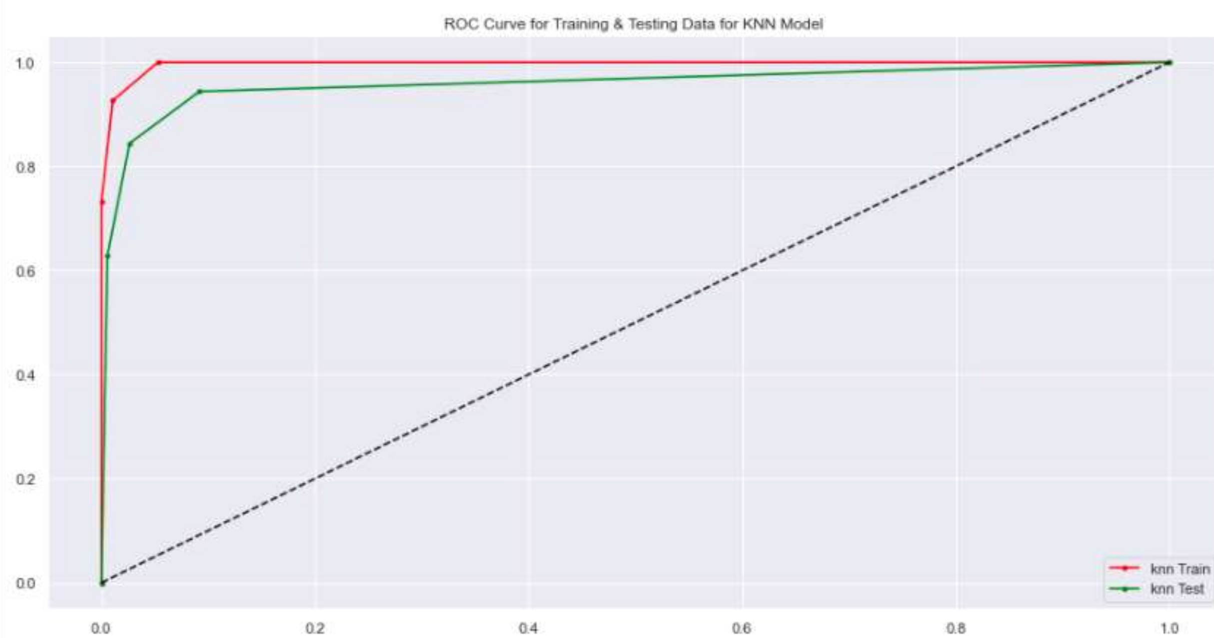
```
              precision    recall  f1-score   support

           0       0.97      0.97      0.97      2808
           1       0.87      0.84      0.86       570

    accuracy                           0.95      3378
   macro avg       0.92      0.91      0.91      3378
weighted avg       0.95      0.95      0.95      3378
```

Confusion Matrix

```
[[2734   74]
 [  89  481]]
```

AUC

```
AUC for the knn Train Data: 0.997
AUC for the knn Test Data: 0.958
```



6. Bagging

Performance of matrix on train data

Accuracy for Training data

0.9999

Classification report

	precision	recall	f1-score	support
0	1.00	1.00	1.00	6556
1	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

Confusion Matrix

```
[[6556    0]
 [   1 1325]]
```

Performance of matrix on test data

Accuracy for Test data

0.9609

Classification report

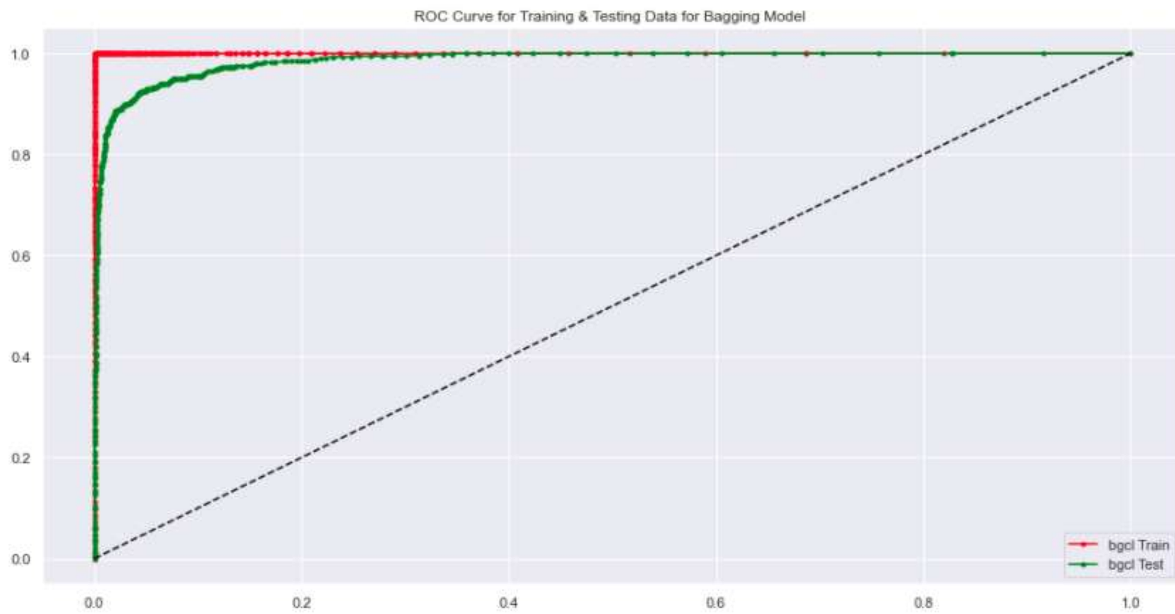
	precision	recall	f1-score	support
0	0.96	0.99	0.98	2808
1	0.94	0.82	0.88	570
accuracy			0.96	3378
macro avg	0.95	0.91	0.93	3378
weighted avg	0.96	0.96	0.96	3378

Confusion Matrix

```
[[2778   30]
 [ 102  468]]
```

AUC

```
AUC for the bgcl Train Data: 1.000
AUC for the bgcl Test Data: 0.986
```



7. Ada Boosting

Performance of matrix on train data

Accuracy for Training data

```
0.8961
```

Classification report

```
              precision    recall  f1-score   support

           0       0.92      0.96      0.94      6556
           1       0.74      0.59      0.66      1326

    accuracy                           0.90      7882
   macro avg       0.83      0.77      0.80      7882
weighted avg       0.89      0.90      0.89      7882
```

Confusion Matrix

```
[[6285  271]
 [ 548  778]]
```

Performance of matrix on test data

Accuracy for Test data

```
0.8931
```

Classification report

	precision	recall	f1-score	support
0	0.96	0.99	0.98	2808
1	0.94	0.82	0.88	570
accuracy			0.96	3378
macro avg	0.95	0.91	0.93	3378
weighted avg	0.96	0.96	0.96	3378

Confusion Matrix

```
[[2778   30]
 [ 102  468]]
```

AUC

```
AUC for the abcl Train Data: 0.917
AUC for the abcl Test Data: 0.902
```

8. Gradient Boosting

Performance of matrix on train data

Accuracy for Training data

0.9779

Classification report

	precision	recall	f1-score	support
0	0.97	1.00	0.99	6556
1	1.00	0.87	0.93	1326
accuracy			0.98	7882
macro avg	0.99	0.94	0.96	7882
weighted avg	0.98	0.98	0.98	7882

Confusion Matrix

```
[[6553    3]
 [ 171 1155]]
```

Performance of matrix on test data

Accuracy for Test data

0.9364

Classification report

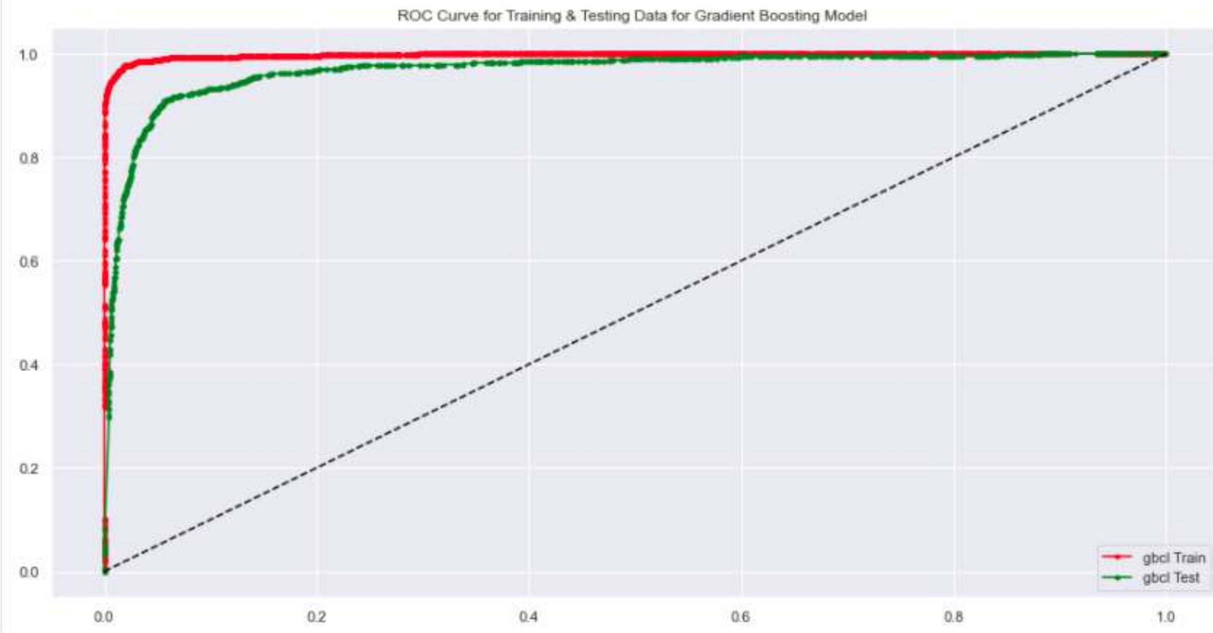
	precision	recall	f1-score	support
0	0.94	0.98	0.96	2808
1	0.90	0.70	0.79	570
accuracy			0.94	3378
macro avg	0.92	0.84	0.88	3378
weighted avg	0.93	0.94	0.93	3378

Confusion Matrix

```
[[2762   46]
 [ 169  401]]
```

AUC

```
AUC for the gbcl Train Data: 0.997
AUC for the gbcl Test Data: 0.968
```



Smote data

Performance of matrix on train data

Y train predict kNN

```
0.9524100061012812
```

Classification report

```
              precision    recall  f1-score   support

           0       1.00      0.91      0.95      6556
           1       0.92      1.00      0.95      6556

    accuracy                           0.95     13112
   macro avg       0.96      0.95      0.95     13112
weighted avg       0.96      0.95      0.95     13112
```

Confusion Matrix

```
[[5952  604]
 [  20 6536]]
```

Performance Matrix on test data set

Accuracy for Test data

```
0.8596802841918295
```

Classification report

```
              precision    recall  f1-score   support

           0       0.97      0.86      0.91      2808
           1       0.55      0.86      0.67       570

    accuracy                           0.86      3378
   macro avg       0.76      0.86      0.79      3378
weighted avg       0.90      0.86      0.87      3378
```

Confusion Matrix

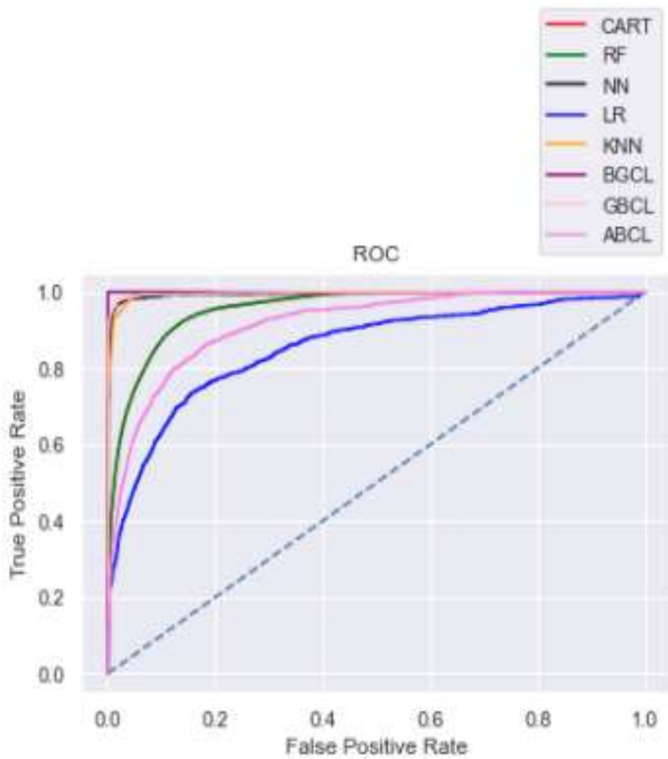
```
[[2414  394]
 [  80  490]]
```

Comparison of the performance metrics from the models

	CART Train	CART Test	Random Forest Train	Random Forest Test	Logistic Regression Train	Logistic Regression Test	Neural Network Train	Neural Network Test	KNN Train	KNN Test	BGCL Train	BGCL Test	GBCL Train	GBCL Test	ABCL Train	ABCL Test
Accuracy	0.92	0.91	0.92	0.91	0.87	0.87	0.97	0.95	0.98	0.95	1.0	0.96	1.00	0.94	0.90	0.89
AUC	0.96	0.93	0.96	0.93	0.85	0.85	0.99	0.93	1.00	0.96	1.0	0.99	1.00	0.97	0.92	0.90
Recall	0.68	0.64	0.66	0.62	0.72	0.70	0.87	0.74	0.93	0.84	1.0	0.94	1.00	0.90	0.74	0.94
Precision	0.80	0.78	0.83	0.81	0.49	0.48	0.98	0.92	0.95	0.87	1.0	0.82	0.87	0.70	0.59	0.82
F1 Score	0.74	0.70	0.74	0.71	0.59	0.57	0.92	0.82	0.94	0.86	1.0	0.88	0.93	0.79	0.66	0.88

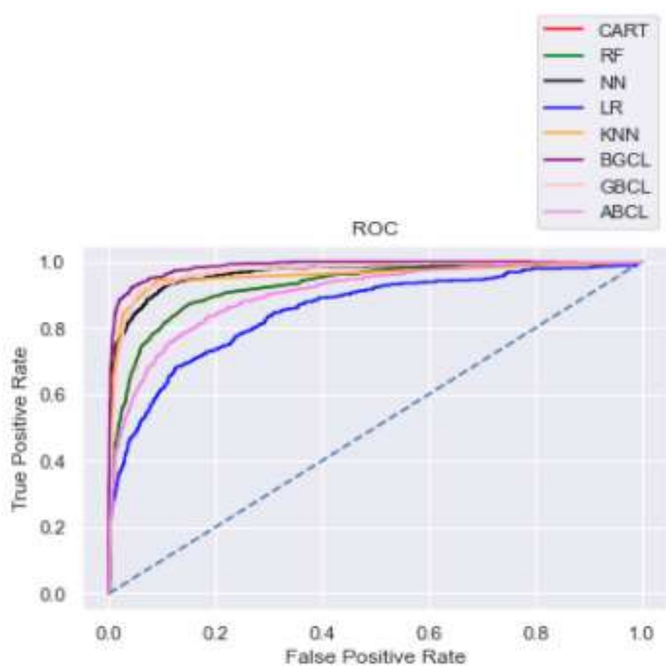
ROC Curve for all the models on the Training data

<matplotlib.legend.Legend at 0x187c6014280>



ROC Curve for all the models on the Test data

<matplotlib.legend.Legend at 0x187c5127d60>



Model Chosen

I would prefer to Artificial Neural network model. Why?

- ❖ I prefer the model which is highest score in Recall, precision, F1 score and Accuracy. Compared to all the models I would suggest the best model is “Artificial neural network”.
- ❖ Since the data is imbalanced used Smote technique to increase the minority class and executed on the same.
- ❖ Hence ‘Bagging and Gradient Boosting’ is overfitting and cannot rely totally on 100% prediction so, I would prefer to go to ANN.
- ❖ Since the data is imbalanced I would suggest the first priority to go head as per high in Recall, second Precision, third F1 score and last Accuracy.

5. Model validation

- ❖ Artificial neural network is also giving the best result in reference to Accuracy in train data is 97%, Recall 87%, Precision 98% and F1 Score 92%. In Test data Accuracy is 95%, Recall is 74%, Precision is 92% and F1 score is 82%.
- ❖ From the graph we can predict that “Artificial neural network” predicts the data best as it is the curve which is above all the curve in training and test data when compared to all the models. The more it shifts to the left side i.e. true positive rate the better the model is.

6. Final interpretation

- ❖ This data is taken for 12 months (1 year).
- ❖ 1896 account holder have churned out of 11260 last year.
- ❖ The given dataset is imbalanced. We will use suitable method to balance the data preferably by oversampling using SMOTE (Synthetic Minority Over-Sampling Technique).
- ❖ No multicollinearity is present. All the predictors are independently affecting customer churn.
- ❖ Churn of accounts has increased due to complaints lodged.
- ❖ High negative impact of ‘Tenure’ on ‘Churn’ establishes the fact that long term customers are loyal.
- ❖ Customers care services have been contacting the each account holders.
- ❖ The dataset has outliers and missing values in which imputed missing values with median and not treated outliers.
- ❖ Maximum Cashback awarded to customers in 12 months is Rs.1997.
- ❖ Customers who logged in via mobile (7482) are more who have churned.
- ❖ Payment of 65 % of the churned account made by Debit & Credit Card.
- ❖ 4124 customers prefer to regular plus account segment.

Recommendations

Excellent product and efficient customer service are the steering factors behind any stable business. Both of them are not in pole positions in the given problem.so, the churn rate of the company is higher than the industry level.

❖ Service charge has to be reduced.

❖ Deep attention account holders who are using Regular plus & Super plan, logging in with mobile and making payments by debit & credit cards.

❖ Some extra benefits may be offered for ‘Regular plus’ and ‘Super’ plan.

❖ Ask for feedback often.

❖ Need to provide different better packages and more benefits to the loyal customers who are there for a long period.

❖ Taking advantage of opportunities to reach out to clients you have not talked to in years.

❖ Focus on Customer Support.

❖ Minimize Seller Mistakes.

THE END