FINAL REPORT

Customer Churn

- Prediction and Recommendation

Contents

1.Introduction

2.EDA and Business Implication

3.Data Cleaning and Pre-processing

4.Model Building

5.Model Validation

6.Final Interpretation and Recommendation

01. Introduction.

a) Defining the problem statement:

The DTH industry is one of the most competitive industries in the world with high acquisition costs and changing tariff plans. Industry also has to face severe competition from market through the attractive offers and discounts from competitors to retain & switch customers. The challenge in front of this service provider was the potential churn of subscribers. So, 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. A proactive retention campaign to retain high value customers from churning is the key to increase & protect revenue for any subscriber-based business. The business objective was to identify the subscribers who have a high propensity to churn and thereby focus the marketing campaigns on a select segments of subscribers and therefore optimize the marketing spend.

b) Need of the study:

Predicting churn is important only to the extent that effective action can be taken to retain the customer before it is too late. Once those customers at risk of churning have been identified, certain measures can be taken for each individual customer to maximize the chances that the customer will remain a customer. Since different customers exhibit different behaviours and preferences, and since different customers churn for different reasons, it is critical to understand the underlying reasons for the customer churn and act accordingly by giving recommendations to retain the customer.

c) Understanding business/social opportunity:

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. Besides the direct loss of revenue that results from a customer abandoning the business, the costs of initially acquiring that customer may not have already been covered by the customer's spending to date. (In other words, acquiring that customer may have actually been a losing investment.) Furthermore, it is always more difficult and expensive to acquire a new customer than it is to retain a current paying customer.

In order to succeed at retaining customers who would otherwise abandon the business, marketers and retention experts must be able to (a) predict in advance which customers are going to churn through churn analysis and (b) know which marketing actions will have the greatest retention impact on each particular customer. Armed with this knowledge, a large proportion of customer churn can be eliminated.

02. EDA and Business Implication.

a) Data Report:

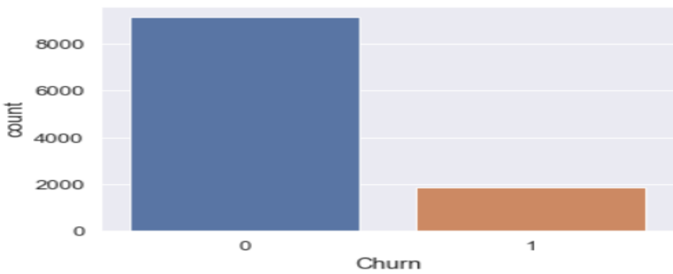
The given dataset is provided by the great learning for the Capstone project- Customer Churn. After performing different data cleaning techniques , we have got information about **11001** customers. This dataset has **one dependent variable 'Churn'** and **17 independent variables** excluding variable 'AccountID'. There is presence of unwanted values('@','#','$') in the dataset that needed to be treated before doing exploratory data analysis. This dataset has got all the required details about the customers in variables that give details about the tenure of the account , Gender of the primary customer of the account, Monthly average revenue generated by account in last 12 months etc.,

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 11001 entries, 0 to 11000
Data columns (total 18 columns):
 #   Column                      Non-Null Count  Dtype 
---  ------                      --------------  ----- 
 0   Tenure                      11001 non-null  int64 
 1   City_Tier                   11001 non-null  object
 2   CC_Contacted_LY             11001 non-null  object
 3   Payment                     11001 non-null  object
 4   Gender                      11001 non-null  object
 5   Service_Score               11001 non-null  object
 6   Account_user_count          11001 non-null  object
 7   account_segment             11001 non-null  object
 8   CC_Agent_Score              11001 non-null  object
 9   Marital_Status              11001 non-null  object
 10  rev_per_month               11001 non-null  int64 
 11  Complain_ly                 11001 non-null  object
 12  rev_growth_yoy              11001 non-null  int64 
 13  coupon_used_for_payment     11001 non-null  object
 14  Day_Since_CC_connect        11001 non-null  object
 15  cashback                    11001 non-null  int64 
 16  Login_device                11001 non-null  object
 17  Churn                       11001 non-null  object
dtypes: int64(4), object(14)
memory usage: 1.5+ MB
```

There are '**four' variables** of datatype - **int** and **'fourteen'** variables of data type - **string**.

Tha data has got **259** number of **duplicate** rows. There is presence of null values in most of the columns and the dataset has total of **2676** null values. Variable **'cashback'** has the **highest percentage(4.18%)** of null values among all the available variables.

The given data is imbalanced based on variable **'Churn'** as it has lesser data points in True Churn category (**16.8%** of total dataset).

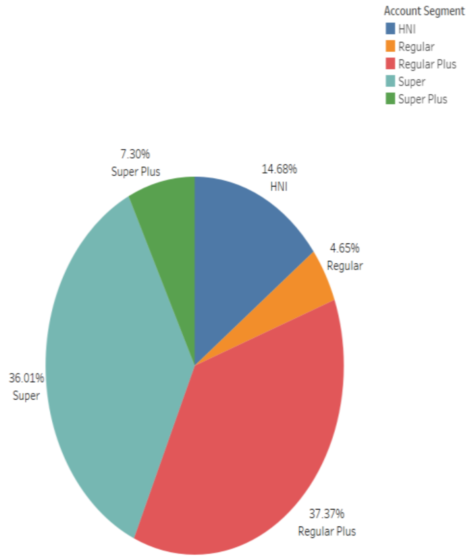


b) <u>Descriptive statistics of data:</u>

	count	unique	top	freq	mean	std	min	25%	50%	75%	max
Tenure	11001	NaN	NaN	NaN	10.2744	8.91056	0	2	9	16	37
City_Tier	11001	4	1.0	7097	NaN	NaN	NaN	NaN	NaN	NaN	NaN
CC_Contacted_LY	11001	45	14.0	663	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Payment	11001	5	Debit Card	4593	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Gender	11001	2	Male	6656	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Service_Score	11001	7	3.0	5360	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Account_user_count	11001	6	4	4898	NaN	NaN	NaN	NaN	NaN	NaN	NaN
account_segment	11001	5	Super	4058	NaN	NaN	NaN	NaN	NaN	NaN	NaN
CC_Agent_Score	11001	6	3.0	3270	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Marital_Status	11001	3	Married	5921	NaN	NaN	NaN	NaN	NaN	NaN	NaN
rev_per_month	11001	NaN	NaN	NaN	5.26879	2.88047	1	3	5	7	13
Complain_ly	11001	3	0.0	7602	NaN	NaN	NaN	NaN	NaN	NaN	NaN
rev_growth_yoy	11001	NaN	NaN	NaN	16.2068	3.75962	4	13	15	19	28
coupon_used_for_payment	11001	17	1	4261	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Day_Since_CC_connect	11001	23	3	2140	NaN	NaN	NaN	NaN	NaN	NaN	NaN
cashback	11001	NaN	NaN	NaN	178.396	45.5118	66	148	166	198	282
Login_device	11001	3	Mobile	7529	NaN	NaN	NaN	NaN	NaN	NaN	NaN

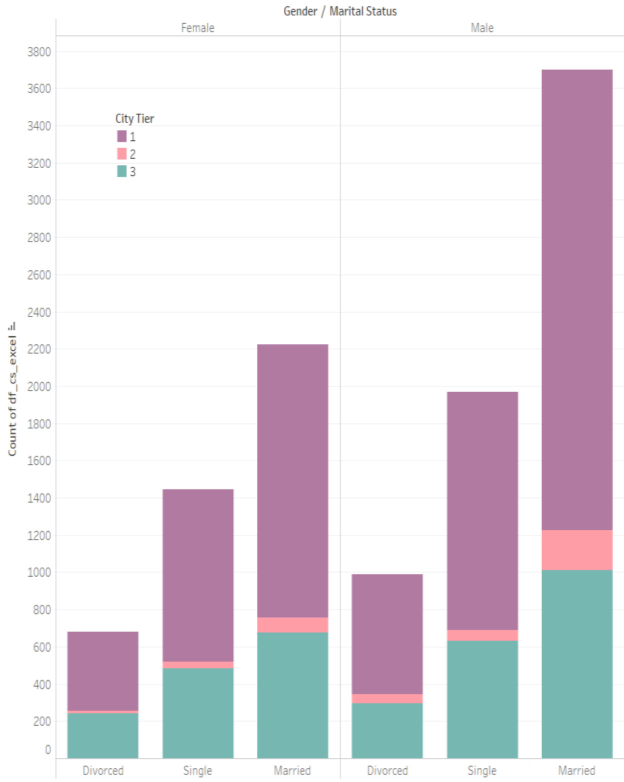
- Of the entire dataset, column **'cashback'** has **highest max** value of 282 and column **'Tenure'** has **least min** value of 0.
- Columns **'cashback'** and **'rev_per_month'** have highest mean value – 178.396 and least mean value – 5.268 respectively.
- Majority of the customers made **payment** through **'Debit Card'** and are coming under **account_segment –'Super'**
- Majority of the customers are **'Male'** and belongs to **City_Tier-01.**
- **Mobile** is the most preferred login device of the customers in the account and most of the customers gave **Satisfaction score** of **3** about the service provided by company
- Majority of the customers are **'Married'** and Most of accounts have **4 no of customers** tagged with the account
- Most of the customers have used coupons to do the payment in last 12 months for single time only

Account Segment



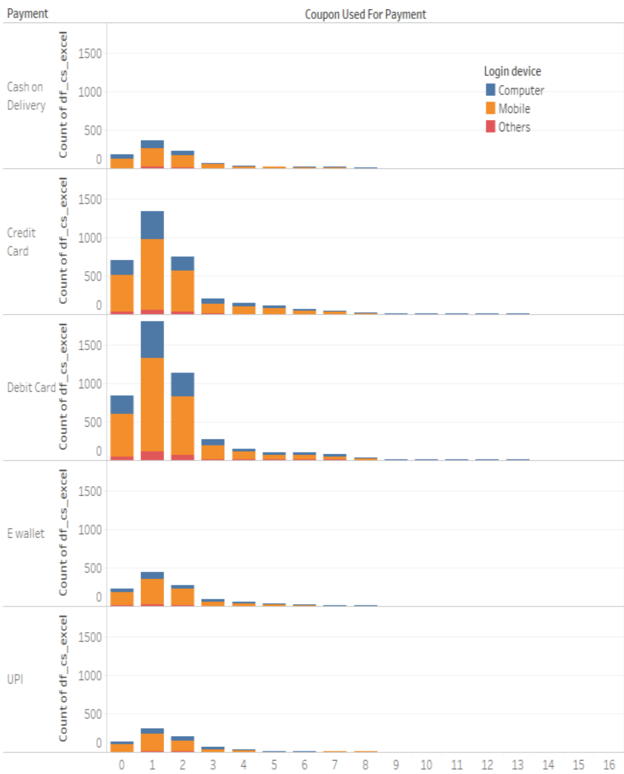
Majority of the customers belong to Account Segments - 'Regular Plus' and 'Super'

Gender-Marital Status-City Tier



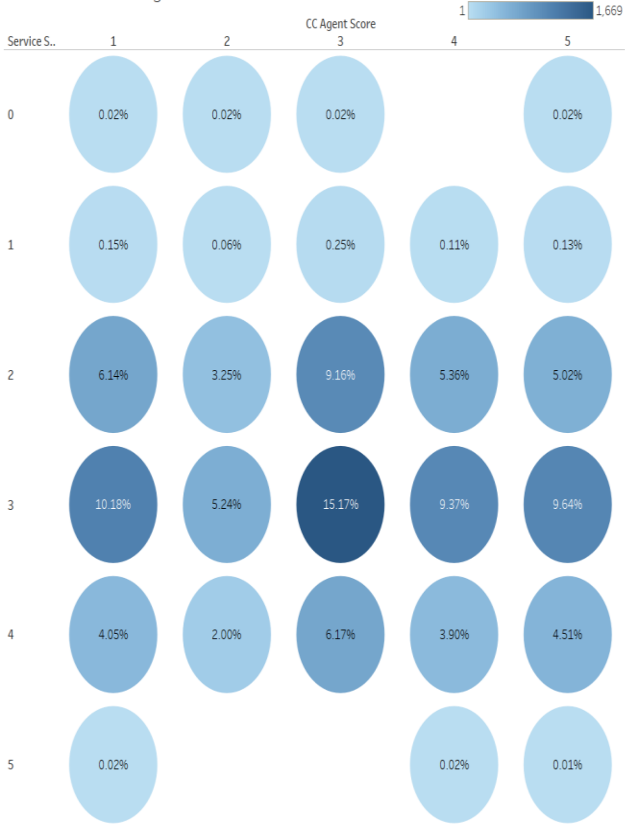
Majority of the customers belong to City Tier 1&3 and are Married

Login Device-Payment -Coupon used for Payment

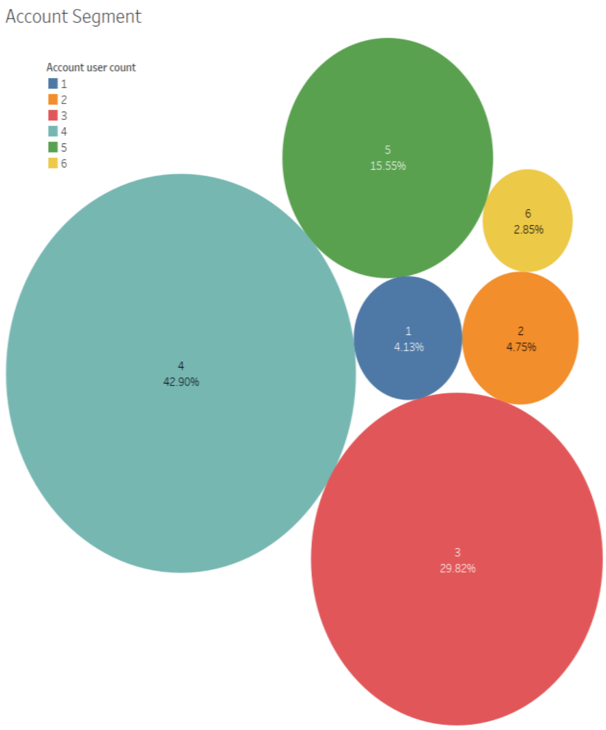


Majority of the customers are using Login Device as '**Mobile**' and are making payment through '**Credit Card' & 'Debit Card'**.

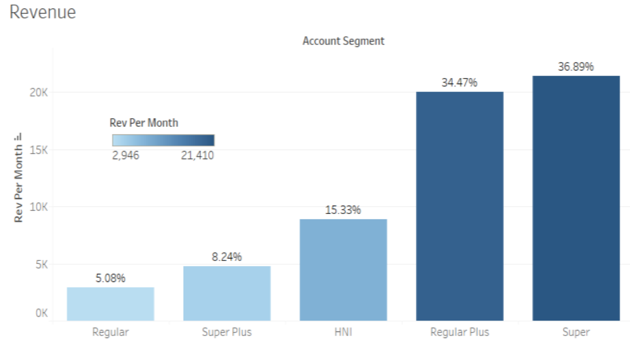
Service Score- CC Agent Score



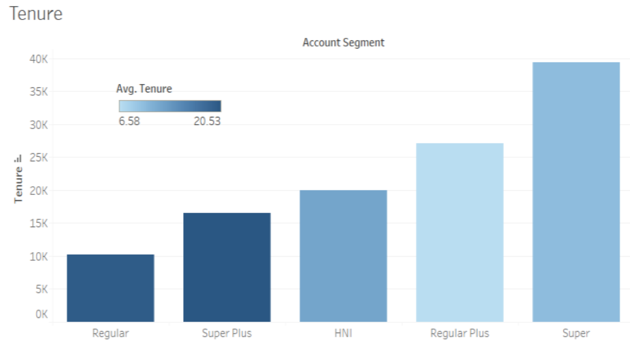
Majority of the customers gave an **average** Rating **'3'** on both Service Score and CC Agent Score.



Majority of the customers have an **account user count** of **'4'**



Account Segment **'Regular Plus' & 'Super'** have got highest Revenue values.

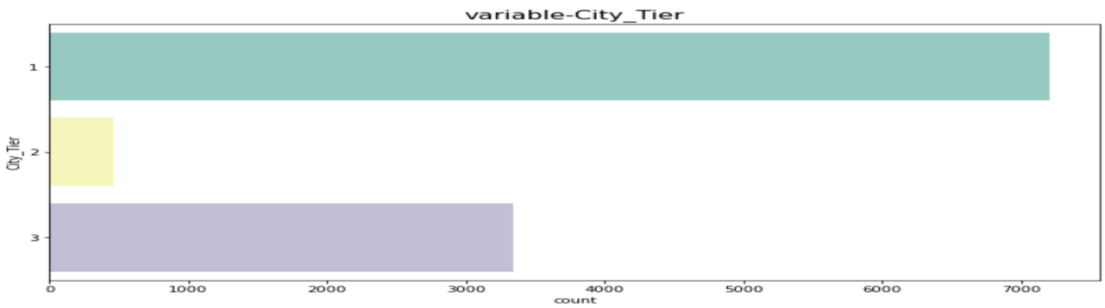


Account Segment **'Regular' & 'Super Plus'** have got highest Avg Tenure.

c) <u>Univariate analysis:</u>

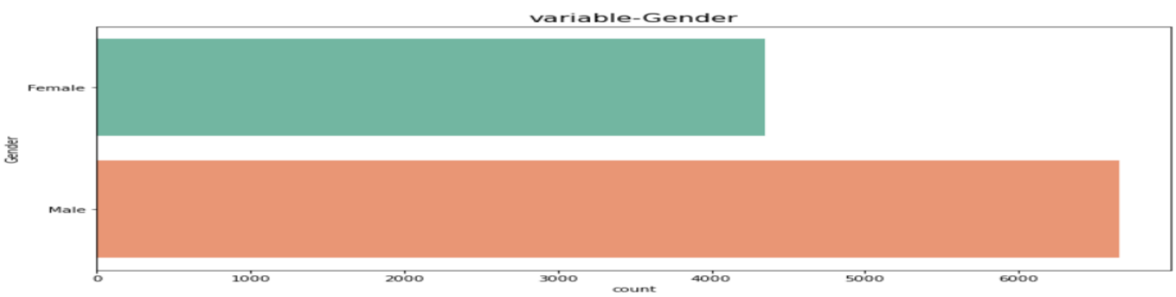
<u>Categorical variables:</u>

Variable – City_Tier



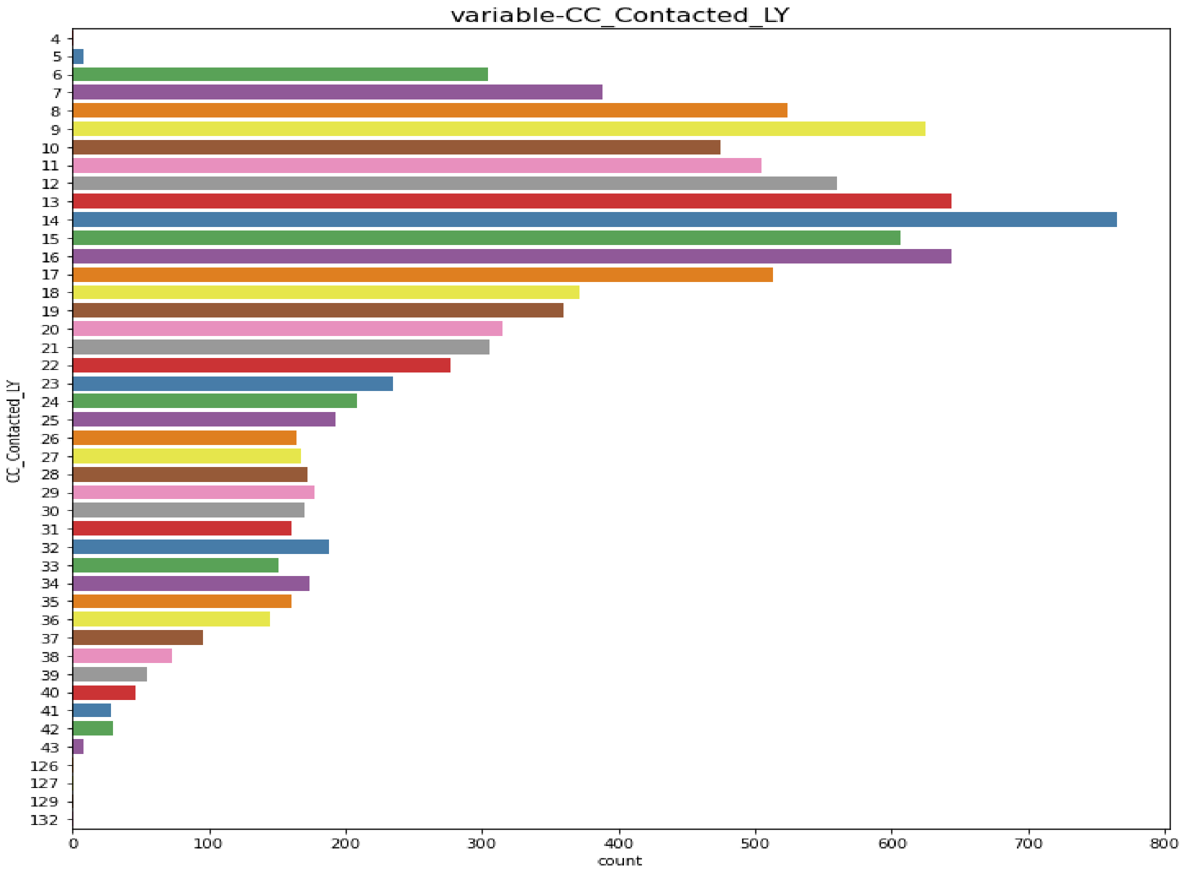
Most of the customers belong to city of 'Tier-01'.

Variable – Gender



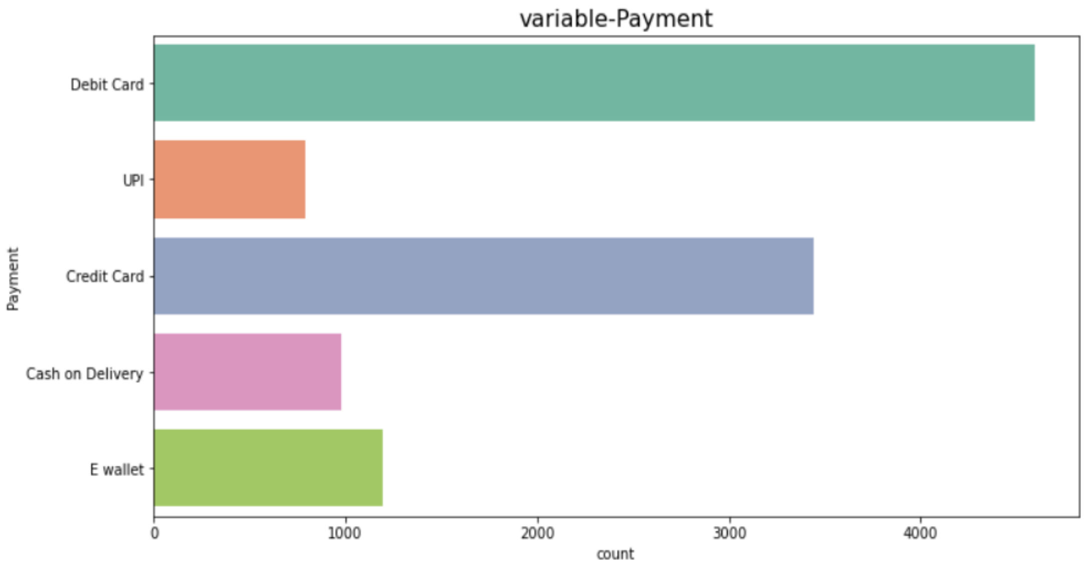
Most of the customers are Male.

Variable – CC_Contacted_LY



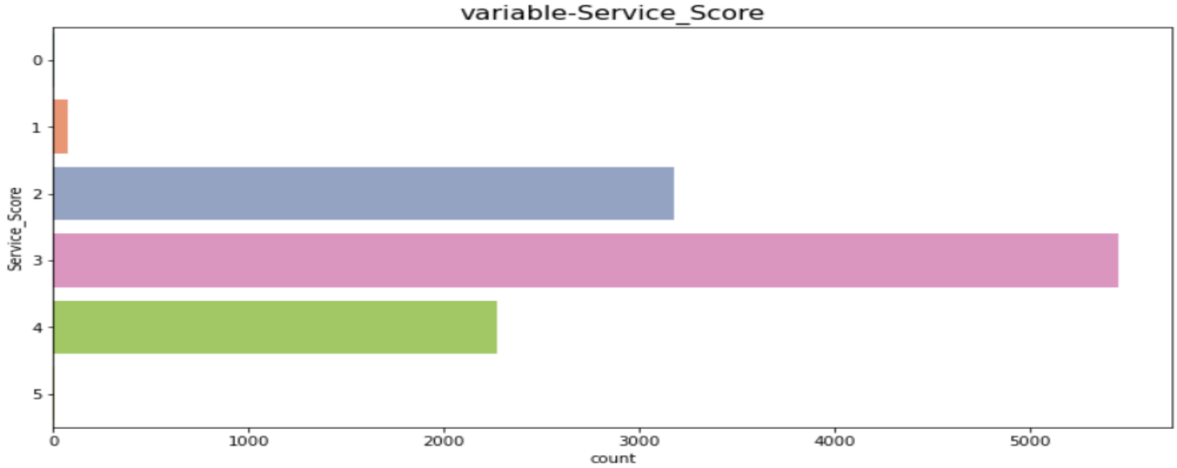
14 times is highest of all the customers of the account has contacted customer care in last 12months

Variable – Payment



Most of the customers made payment through 'Debit Card'.

Variable – Service_score

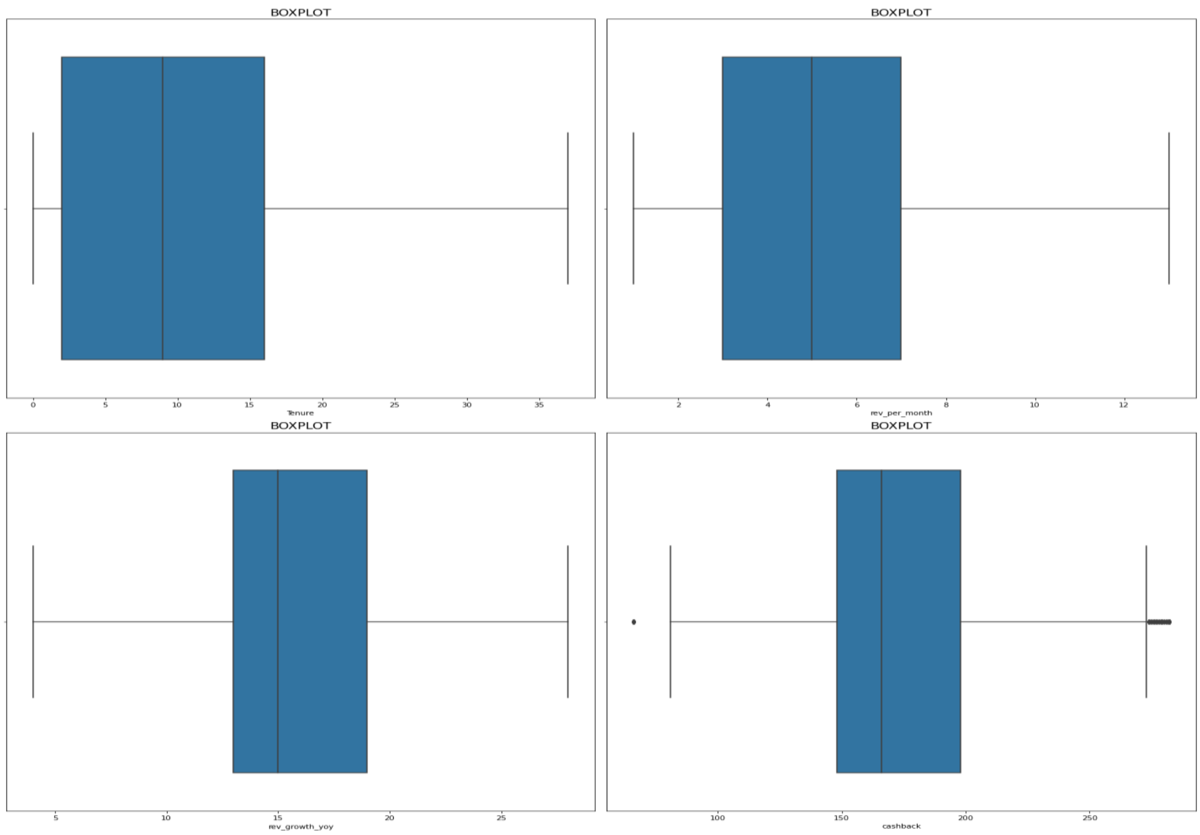


Most of the customers gave service score of 3.

<u>Continous variables:</u>

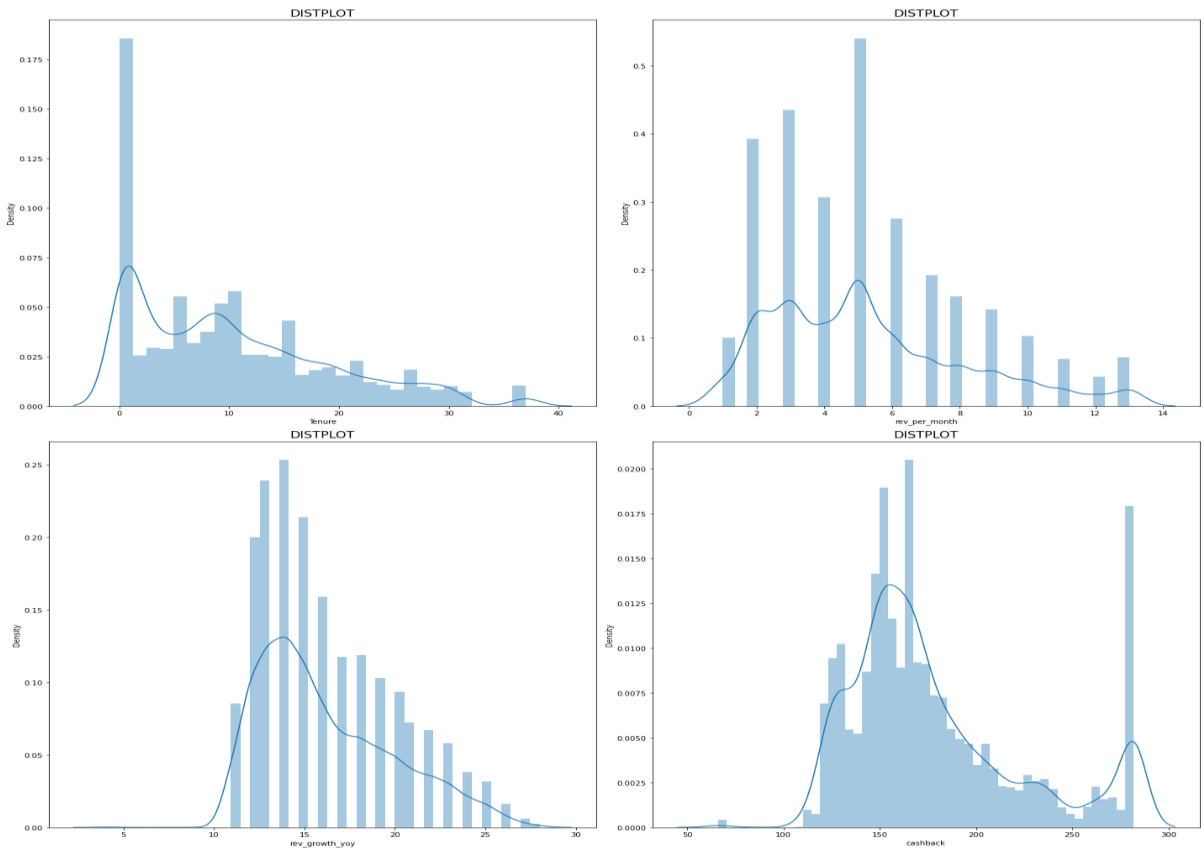
We have got four continuous variables in the data set - **'Tenure', 'rev_per_month', 'rev_growth_yoy', 'cashback'**

Boxplot :



We have got outliers in the variable ‘cashback’, which are to be treated in the data analysis.

Distplot :

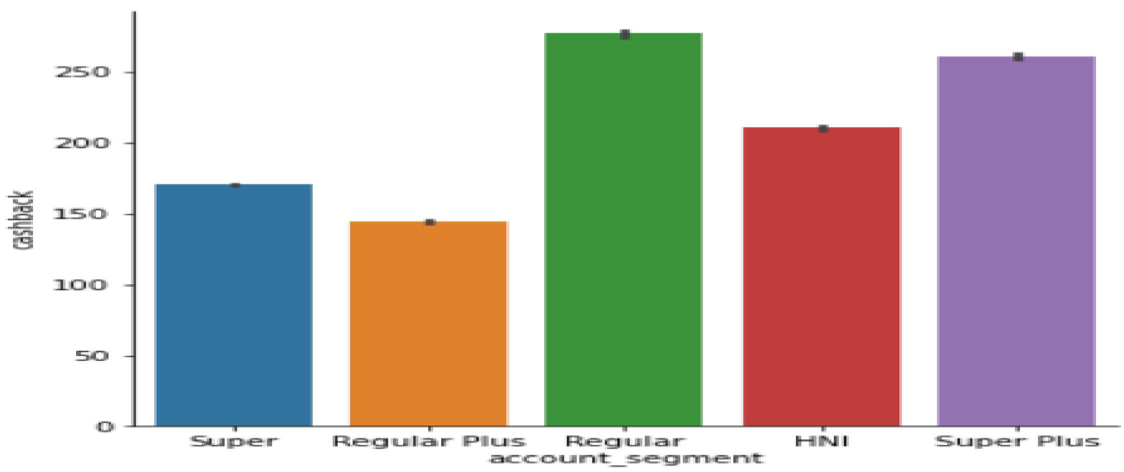


From above plot, we can see that the variables are not symmetric.

Variable ‘Tenure’ has skewness of value – 0.817 and variable- cashback has skewness of value -0.99

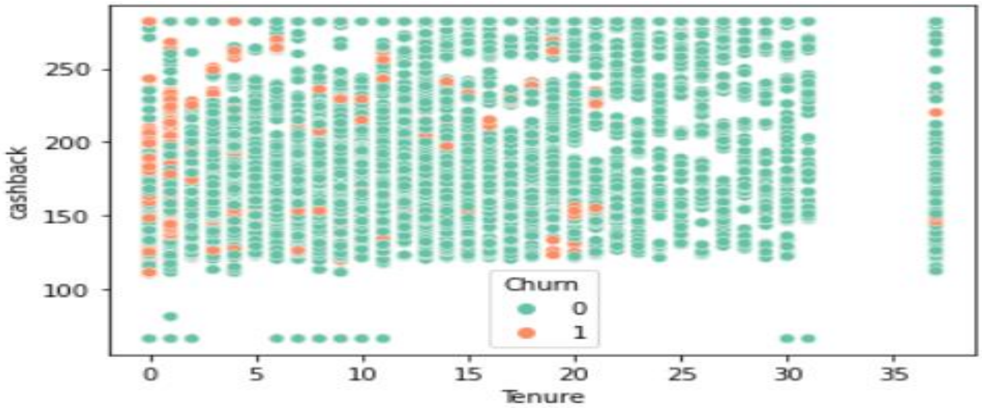
d) <u>Bivariate analysis:</u>

Cashback vs account segment:



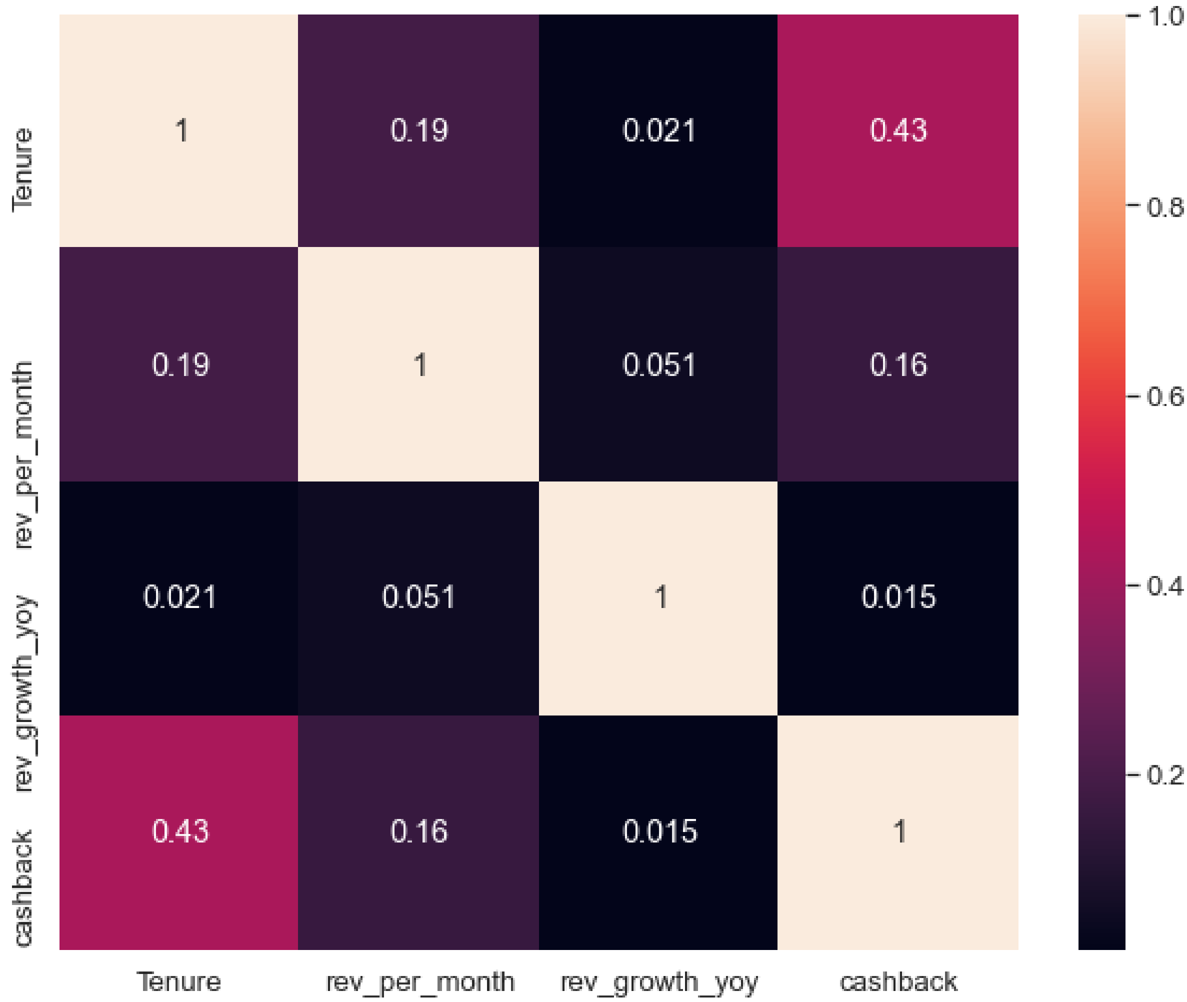
We can see from the above plot that account_segment ‘Regular’ has used higher cashback

Tenure vs cashback:



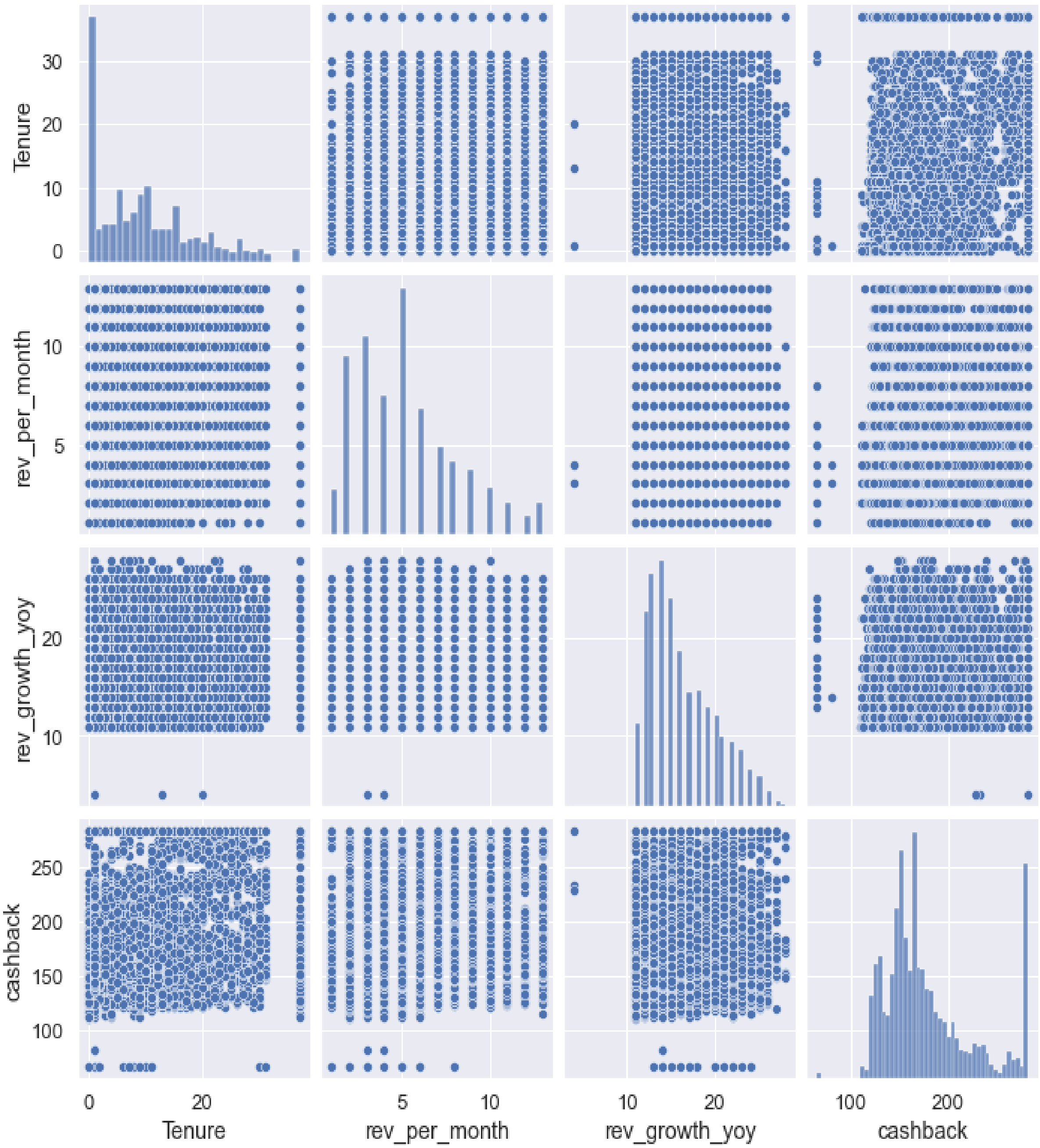
Most of the customers who churns out are having very low tenure and using cashback at high rate.

HeatMap:



From the above plot we can see that , there is **highest** correlation between ‘**cashback**’ and ‘**tenure**’ and **least** correlation between ‘**cashback**’ and ‘**rev_growth_yoy**.’

Pairplot:



From the above plot we can see that there is no linear relationship between any of the variables.

03 .Data Cleaning and Business Pre-processing.

a) Removal of unwanted variables:

- There is presence of invalid data values such as **@,#,$,&** in the variables '**Tenure**',' **Account_user_count**',' **rev_per_month**',' **rev_growth_yoy**',' **coupon_used_for_payment**'. We need to treat these values in the dataset.
- **Gender Female** is available twice in different cases **('F', 'Female')**. To avoid this being considered as 2 different, correct to single format.(**Female**) ,also **Gender Male** is

available twice in different cases (**'M', 'Male'**). To avoid this being considered as 2 different, correct to single format(**Male**)

- **account_segment Regular Plus** is available twice in different cases (**'Regular +', 'Regular Plus'**). To avoid this being considered as 2 different, correct to single format.(**Regular Plus)**,also account_segment **Super Plus** is available twice in different cases **('Super +', 'Super Plus'**). To avoid this being considered as 2 different, correct to single format(**'Super Plus)**
- **Login_device option &&&&** is replaced with '**others'**. To avoid unnecessary confusion.

b)Missing Value treatment :

There is presence of null values in most of the columns and the dataset has total of **2676** null values. Variable **'cashback'** has the **highest percentage(4.18%)** of null values among all the available variables.

We can replace the missing value with a measure of central tendency of the column it's present in. These measures are mean and median if the column variable type is numerical, and mode if the column variable type is categorical.

For numerical variables**, Mean imputation** works better if the distribution is **normally-distributed or has a Gaussian distribution**, while **median imputation** is preferable for **skewed distribution**(be it right or left). For the above dataset, most of the variables are skewed. So, we choose to use median imputation.

For Categorical variables, Mode imputation means replacing missing values by the mode, or the **most frequent- category value**.

Missing values in Variables

```
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
```

Original

```
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
```

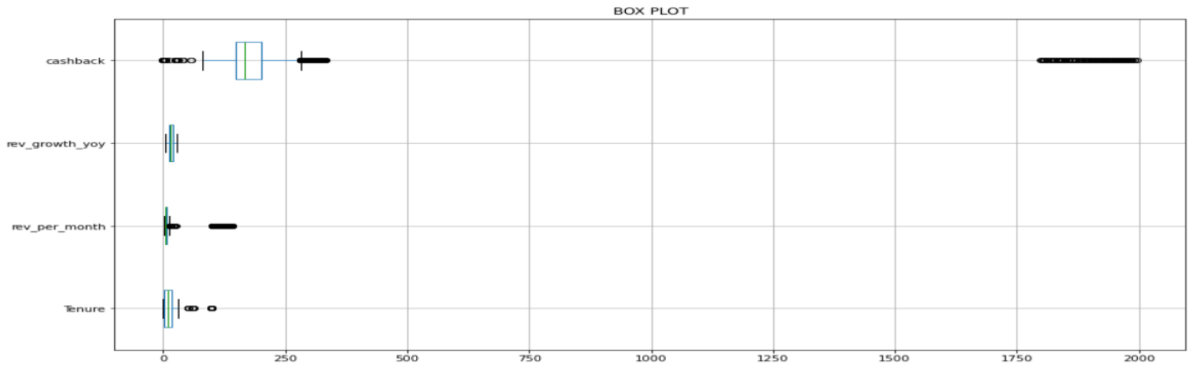
Updated

We have imputed '**Categorical**' variables with **Mode** and **Continuous** variables with **Median** of the respective columns.

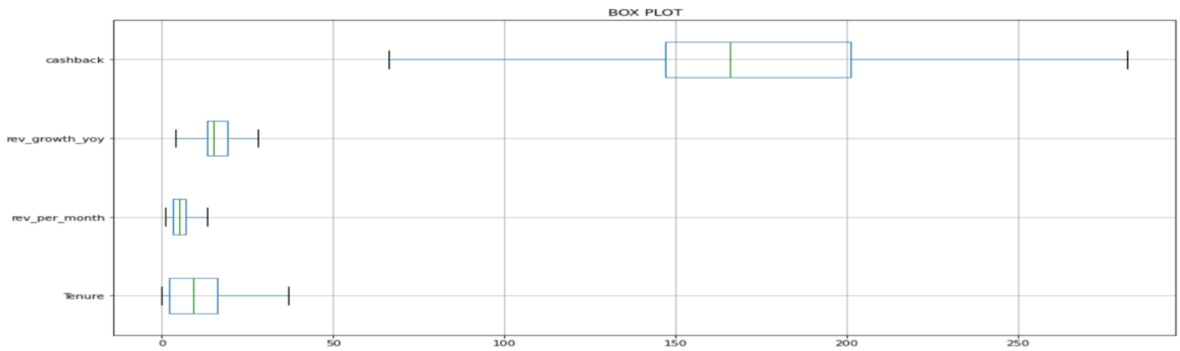
c) Outlier treatment :

Outlier detection is an important task in data mining activities and involves identifying a set of observations whose values deviate from the expected range. These extreme values can unduly influence the results of the analysis and lead to incorrect conclusions. It is extremely important to treat these outliers present in the variables.

There is presence of outliers in the variables **'Tenure','rev_per_month','cashback'.**



We are treating the outliers in the variables by capping with Max and min value.



d) Variable transformation :

We have to change the data type of certain variables from the original data type based on the data associated with the respective variable.

Data types of the Variables

```
 #   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
```

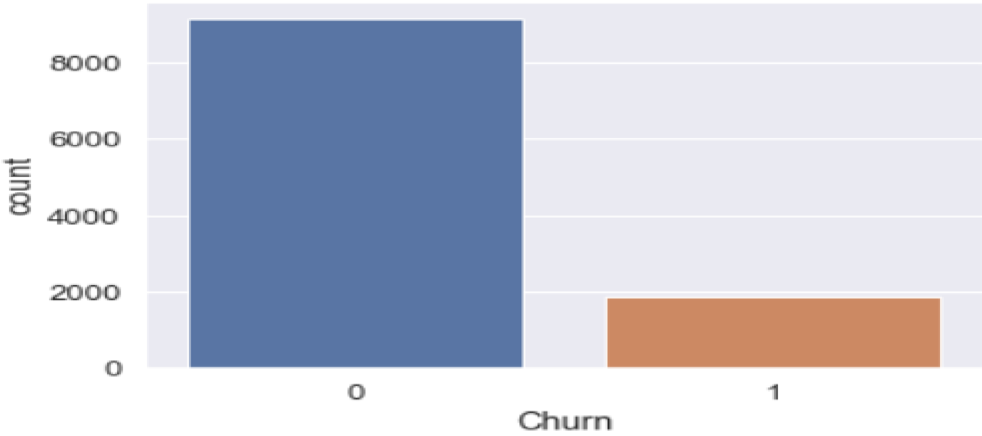
Original

```
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   Tenure                  11001 non-null  float64
 1   City_Tier               11001 non-null  object
 2   CC_Contacted_LY         11001 non-null  object
 3   Payment                 11001 non-null  object
 4   Gender                  11001 non-null  object
 5   Service_Score           11001 non-null  object
 6   Account_user_count      11001 non-null  object
 7   account_segment         11001 non-null  object
 8   CC_Agent_Score          11001 non-null  object
 9   Marital_Status          11001 non-null  object
 10  rev_per_month           11001 non-null  float64
 11  Complain_ly             11001 non-null  object
 12  rev_growth_yoy          11001 non-null  float64
 13  coupon_used_for_payment 11001 non-null  object
 14  Day_Since_CC_connect    11001 non-null  object
 15  cashback                11001 non-null  float64
 16  Login_device            11001 non-null  object
dtypes: float64(4), object(13)
memory usage: 1.5+ MB
```

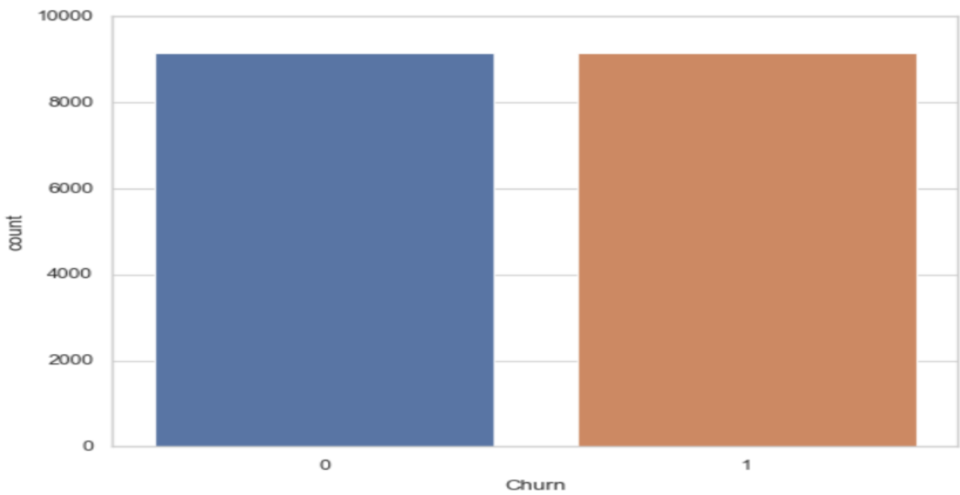
Updated

04 .Model Building.

The given data is Imbalanced based on variable **'Churn'** as it has Lesser datapoints in True Churn category (**16.8%** of total dataset).



There are several techniques to handle the imbalance in a dataset. We can techniques such as Re-sampling Technique,in which we focus on balancing the classes in the training data (data preprocessing) before providing the data as input to the machine learning algorithm. The main objective of balancing classes is to either increase the frequency of the minority class or decrease the frequency of the majority class. This is done in order to obtain approximately the same number of instances for both the classes.



In the above case we try to increase the frequency of the customers of churn class or decrease the frequency of the customers of non-churn class to obtain approximately the same number of instances for both the classes.

The quality of the data affects the quality of the generated model. Therefore it is very important that significant effort should be spent in the data pre-processing phase to achieve the highest model quality. We have prepared the dataset for model building by doing certain pre-processing on the data.

Dataset:

	Tenure	City_Tier	CC_Contacted_LY	Payment	Gender	Service_Score	Account_user_count	account_segment	CC_Agent_Score	Marital_Status
0	4	3	6	Debit Card	Female	3	3	Super	2	Single
1	0	1	8	UPI	Male	3	4	Regular Plus	3	Single
2	0	1	30	Debit Card	Male	2	4	Regular Plus	3	Single
3	0	3	15	Debit Card	Male	2	4	Super	5	Single
4	0	1	12	Credit Card	Male	2	3	Regular Plus	5	Single

The given dataset has got four object data type variables.

```
df['Payment'].value_counts()

Debit Card          4593
Credit Card         3441
E wallet            1195
Cash on Delivery     977
UPI                  795
```

```
df['Gender'].value_counts()

Male      6656
Female    4345
```

```
df['account_segment'].value_counts()

Regular Plus    4111
Super           3961
HNI             1615
Super Plus       803
Regular          511
```

```
df['Marital_Status'].value_counts()

Married     5921
Single      3412
Divorced    1668
```

We need to encode the data for modelling. We can do label encoding for the ordinal variable 'account_segment' and one hot coding for the remaining variables.

After encoding the object type data , we have got the following dataset,

	Tenure	City_Tier	CC_Contacted_LY	Service_Score	Account_user_count	account_segment	CC_Agent_Score	rev_per_month	Complain_ly	rev_growth_yoy
0	4	3	6	3	3	3	2	9	1	11
1	0	1	8	3	4	2	3	7	1	15
2	0	1	30	2	4	2	3	6	1	14
3	0	3	15	2	4	3	5	8	0	23
4	0	1	12	2	3	2	5	3	0	11
5	0	1	22	3	4	2	5	2	1	22
6	2	3	11	2	3	3	2	4	0	14
7	0	1	6	3	3	2	2	3	1	16
8	13	3	9	2	4	2	3	2	1	14
9	0	1	31	2	5	2	3	2	0	12

Scaling of variables does not affect the accuracy of the model. We can do scaling as per the model requirement.

Train-Test Split:

The trained model need to be perform well on the new,unseen data. In order to simulate unseen data,we have to split the available data into 2 parts(train set and test set).So, We are splitting the data into training set(70% of the original data) and testing set(30% of the original data).

This training set is used to build a predictive model and such trained model is then applied on the testing set to make predictions. Selection of the best model is made on basis of model's performance on the testing set.

As per our project, we have to predict whether a customer is going to churn or not. We have target variable of qualitative data type. So, we are going to build classification models for prediction.

Splitting data into train and test (70:30)

```
x_train.head()
```

	Tenure	City_Tier	CC_Contacted_LY	Service_Score	Account_user_count	account_segment	CC_Agent_Score	rev_per_month	Complain_ly
7345	1	2	16	3	4	2	1	2	0
4178	27	3	13	3	4	3	3	4	1
1616	28	1	26	2	3	3	1	4	1
2775	19	3	13	3	5	5	2	8	0
10273	37	1	22	4	4	2	1	9	0

```
x_test.head()
```

	Tenure	City_Tier	CC_Contacted_LY	Service_Score	Account_user_count	account_segment	CC_Agent_Score	rev_per_month	Complain_ly
3380	3	1	17	4	4	2	4	2	0
10114	8	3	22	4	4	3	5	10	0
8577	1	1	17	3	3	2	5	5	0
7617	21	1	14	3	3	2	4	4	0
9229	1	1	11	3	4	2	1	5	1

```
train_labels.head()

7345     0
4178     0
1616     0
2775     0
10273    0
Name: Churn, dtype: int64
```

```
test_labels.head()

3380     0
10114    0
8577     0
7617     0
9229     0
Name: Churn, dtype: int64
```

Predicting churn is important only to the extent that effective action can be taken to retain the customer before it is too late. Once those customers at risk of churning have been identified, certain measures can be taken for each individual customer to maximize the chances that the customer will remain a customer.

As per our project, we have to predict whether a customer is going to **churn or not**. We have target variable of qualitative data type. So, we are going to build **classification models** for prediction.

We have used variety of models such as Logistic Regression, KNN Classifier, Decision Tree etc., for prediction on test data and summarized overall results.

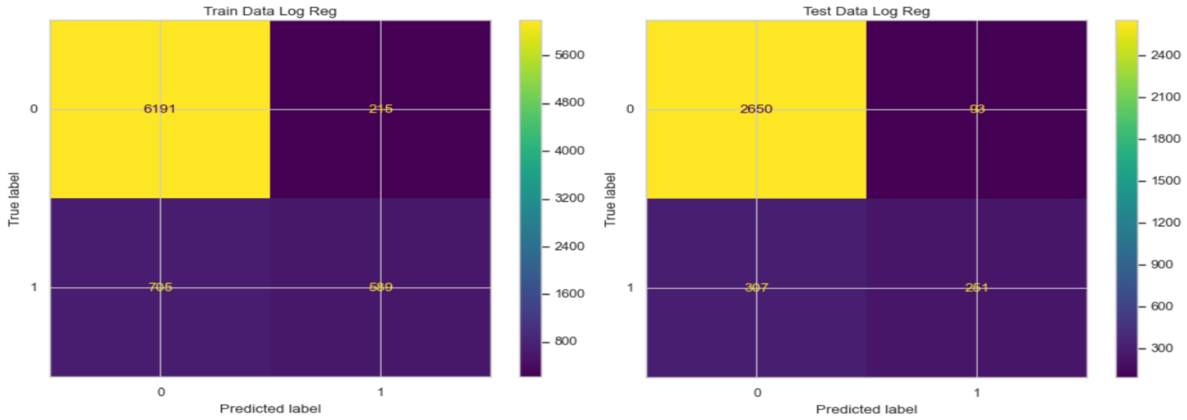
<u>Logistic Regression:</u>

This is a predictive analysis algorithm based on the concept of probability.

```
LogisticRegression(max_iter=10000, n_jobs=2, penalty='none', solver='newton-cg',
                   verbose=True)
```

	Train data	Test data
Model Score	0.88	0.88
Recall	0.46	0.45
F1 Score	0.56	0.56
AUC	0.873	0.859

Confusion Matrix:



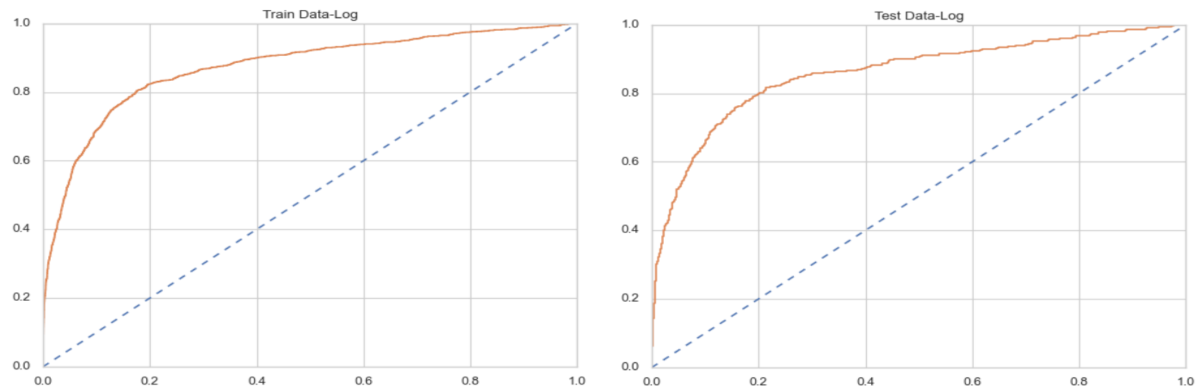
Classification Report:

```
Classification Report of the training data:

              precision    recall  f1-score   support

           0       0.90      0.97      0.93      6406
           1       0.73      0.46      0.56      1294

    accuracy                           0.88      7700
   macro avg       0.82      0.71      0.75      7700
weighted avg       0.87      0.88      0.87      7700

Classification Report of the test data:

              precision    recall  f1-score   support

           0       0.90      0.97      0.93      2743
           1       0.73      0.45      0.56       558

    accuracy                           0.88      3301
   macro avg       0.81      0.71      0.74      3301
weighted avg       0.87      0.88      0.87      3301
```

ROC Curve:

```
AUC: 0.873
Text(0.5, 1.0, 'Train Data-Log')
```

```
AUC: 0.859
Text(0.5, 1.0, 'Test Data-Log')
```



Training and Test set results are almost similar, and with the overall measures high, the model is a good model.

Feature Importance :

The coefficient for Tenure is -0.1448
The coefficient for City_Tier is 0.1158
The coefficient for CC_Contacted_LY is 0.017
The coefficient for rev_per_month is 0.094
The coefficient for Complain_ly is 1.154
The coefficient for coupon_used_for_payment is 0.096
The coefficient for Day_Since_CC_connect is -0.078
The coefficient for Payment_Credit Card is -2.278
The coefficient for Payment_Debit Card is -2.113
The coefficient for Payment_E wallet is -1.589
The coefficient for Payment_UPI is -2.529
The coefficient for Gender_Female is -0.513
The coefficient for Marital_Status_Married is -1.589
The coefficient for Marital_Status_Single is -0.512
The coefficient for Login_device_Others is -1.268

From above results , The most **important feature** is **'Payment_UPI'.**

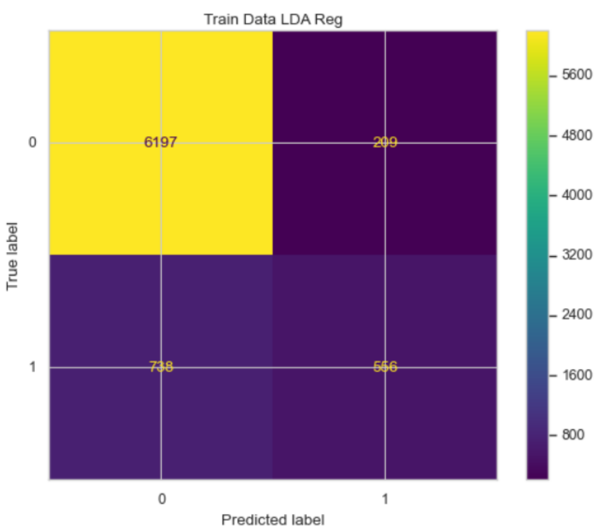
Linear Discriminant analysis:

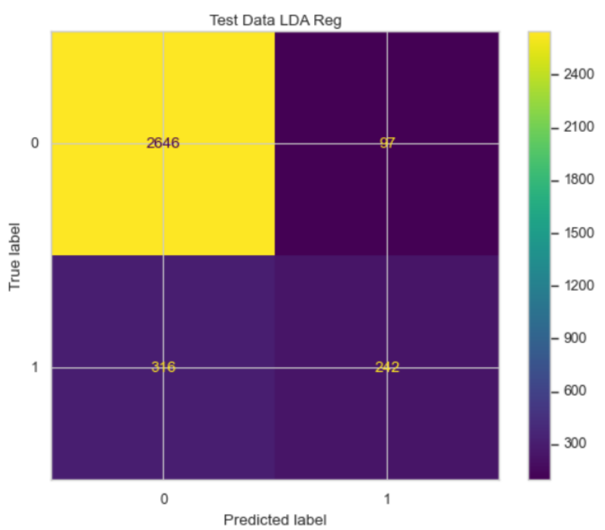
This is a predictive analysis algorithm used for the classification of target variables.

```
LinearDiscriminantAnalysis()
```

	Train data	Test data
Model Score	0.88	0.87
Recall	0.43	0.43
F1 Score	0.54	0.54
AUC	0.863	0.85

Confusion Matrix:





Classification Report:

```
Classification Report of the training data LDA:

              precision    recall  f1-score   support

           0       0.89      0.97      0.93      6406
           1       0.73      0.43      0.54      1294

    accuracy                           0.88      7700
   macro avg       0.81      0.70      0.73      7700
weighted avg       0.87      0.88      0.86      7700

Classification Report of the test data LDA:

              precision    recall  f1-score   support

           0       0.89      0.96      0.93      2743
           1       0.71      0.43      0.54       558

    accuracy                           0.87      3301
   macro avg       0.80      0.70      0.73      3301
weighted avg       0.86      0.87      0.86      3301
```

Training and Test set results are almost similar, and with the overall measures high, the model is a good model.

KNN Classifier:

KNN is used for classification, the output can be calculated as the class with the highest frequency from the K-most similar instances. Each instance in essence votes for their class and the class with the most votes is taken as the prediction.

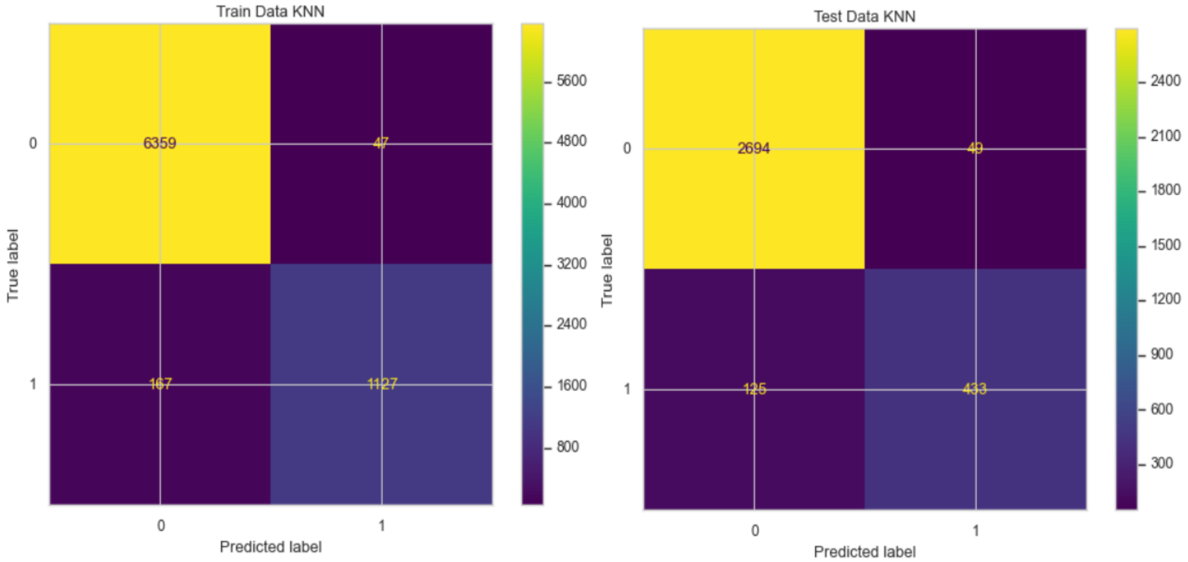
```
knn=KNeighborsClassifier()
```

We need to scale the data for using in this model , as the model works based on distance calculations.

	Tenure	City_Tier	CC_Contacted_LY	Service_Score	Account_user_count	account_segment	CC_Agent_Score	rev_per_month	Complain_ly
0	-0.701579	1.479979	-1.341549	0.135956	-0.683005	0.090912	-0.770012	1.254313	1.61752
1	-1.148284	-0.709243	-1.115216	0.135956	0.299311	-0.818288	-0.041708	0.580795	1.61752
2	-1.148284	-0.709243	1.374447	-1.247623	0.299311	-0.818288	-0.041708	0.244036	1.61752
3	-1.148284	1.479979	-0.323051	-1.247623	0.299311	0.090912	1.414899	0.917554	-0.61823
4	-1.148284	-0.709243	-0.662550	-1.247623	-0.683005	-0.818288	1.414899	-0.766240	-0.61823

	Train data	Test data
Model Score	0.97	0.95
Recall	0.87	0.78
F1 Score	0.91	0.83
AUC	0.994	0.974

Confusion Matrix:



Classification Report:

```
Classification Report of the training data KNN:

              precision    recall  f1-score   support

           0       0.97      0.99      0.98      6406
           1       0.96      0.87      0.91      1294

    accuracy                           0.97      7700
   macro avg       0.97      0.93      0.95      7700
weighted avg       0.97      0.97      0.97      7700


Classification Report of the test data kNN:

              precision    recall  f1-score   support

           0       0.96      0.98      0.97      2743
           1       0.90      0.78      0.83       558

    accuracy                           0.95      3301
   macro avg       0.93      0.88      0.90      3301
weighted avg       0.95      0.95      0.95      3301
```

Training and Test set results are almost similar, and with the very high overall measures , the model is a very good model.

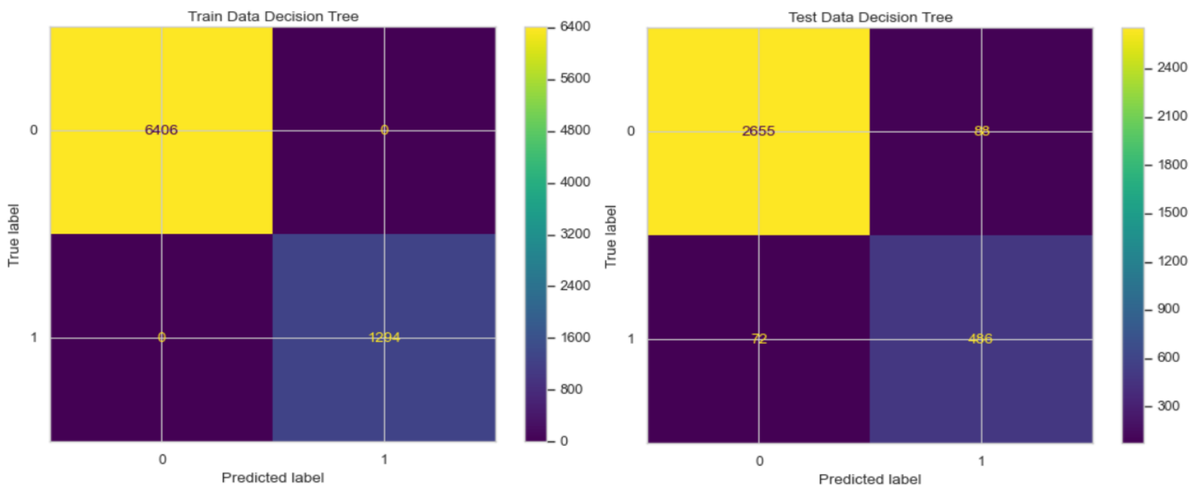
Decision Tree Classifier:

This supervised learning method is useful for classification and regression. The generated model will help in predicting the target variable through learning simple decision rules.

```
DecisionTreeClassifier()
```

	Train data	Test data
Model Score	1	0.95
Recall	1	0.87
F1 Score	1	0.86
AUC	1	0.919

Confusion Matrix:



Classification Report:

```
Classification Report of the training data Decision Tree:

              precision    recall  f1-score   support

           0       1.00      1.00      1.00      6406
           1       1.00      1.00      1.00      1294

    accuracy                           1.00      7700
   macro avg       1.00      1.00      1.00      7700
weighted avg       1.00      1.00      1.00      7700


Classification Report of the test data Decision Tree:

              precision    recall  f1-score   support

           0       0.97      0.97      0.97      2743
           1       0.85      0.87      0.86       558

    accuracy                           0.95      3301
   macro avg       0.91      0.92      0.91      3301
weighted avg       0.95      0.95      0.95      3301
```

Training and Test set results are almost similar, the model is performing exceptionally well on train data set. It is a overfit model.

Feature importance:

	Imp
Tenure	0.295797
CC_Agent_Score	0.084935
Day_Since_CC_connect	0.079184
rev_growth_yoy	0.066625
CC_Contacted_LY	0.058782
Complain_ly	0.056617
rev_per_month	0.050099
account_segment	0.034768
cashback	0.029659
Login_device_Computer	0.027811
Marital_Status_Single	0.027423
Account_user_count	0.026910
City_Tier	0.022665
Payment_Debit Card	0.020985
Payment_E wallet	0.016097
Login_device_Mobile	0.014811
coupon_used_for_payment	0.013264
Payment_Cash on Delivery	0.012192
Payment_Credit Card	0.011774
Gender_Male	0.010619
Service_Score	0.010270
Gender_Female	0.009793
Marital_Status_Married	0.007830
Payment_UPI	0.007543
Login_device_Others	0.002126
Marital_Status_Divorced	0.001421

The highest importance feature denoted by this method is **Tenure (29.57% importance)**

MLP Classifier(Artificial Neural Network):

It is a supervised learning algorithm that learn a non-linear function approximator for either classification or regression. It is different from logistic regression, in that between the input and the output layer, there can be one or more non-linear layers, called hidden layers.

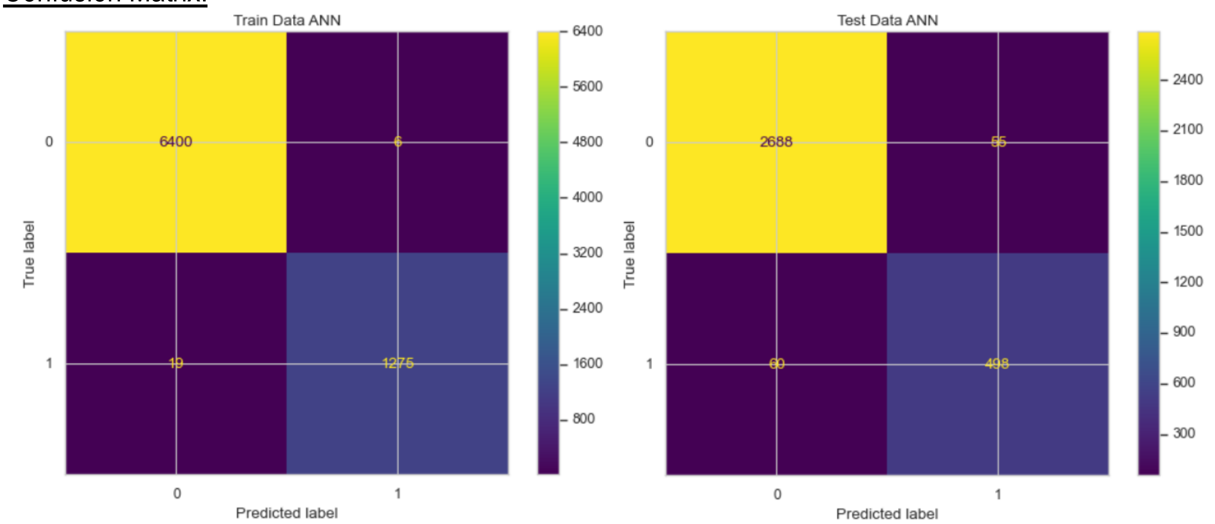
```
ann = MLPClassifier()
```

We need to scale the data for using in this model.

	Tenure	City_Tier	CC_Contacted_LY	Service_Score	Account_user_count	account_segment	CC_Agent_Score	rev_per_month	Complain_ly
0	-0.701579	1.479979	-1.341549	0.135956	-0.683005	0.090912	-0.770012	1.254313	1.61752
1	-1.148284	-0.709243	-1.115216	0.135956	0.299311	-0.818288	-0.041708	0.580795	1.61752
2	-1.148284	-0.709243	1.374447	-1.247623	0.299311	-0.818288	-0.041708	0.244036	1.61752
3	-1.148284	1.479979	-0.323051	-1.247623	0.299311	0.090912	1.414899	0.917554	-0.61823
4	-1.148284	-0.709243	-0.662550	-1.247623	-0.683005	-0.818288	1.414899	-0.766240	-0.61823

	Train data	Test data
Model Score	1	0.97
Recall	0.99	0.89
F1 Score	0.99	0.9
AUC	1	0.985

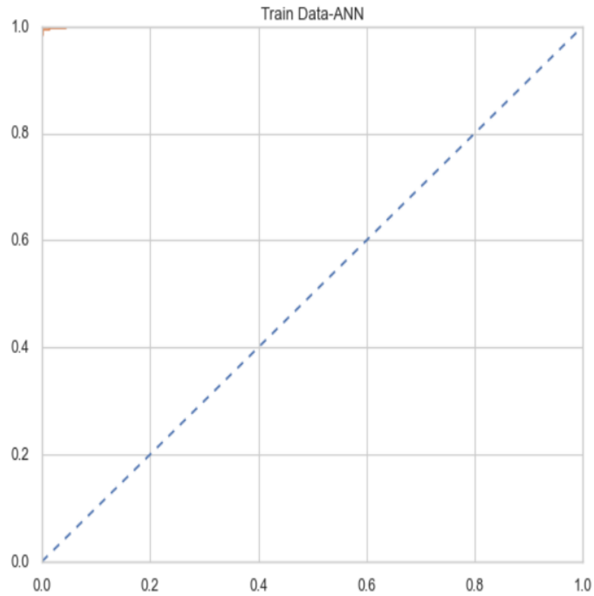
Confusion Matrix:



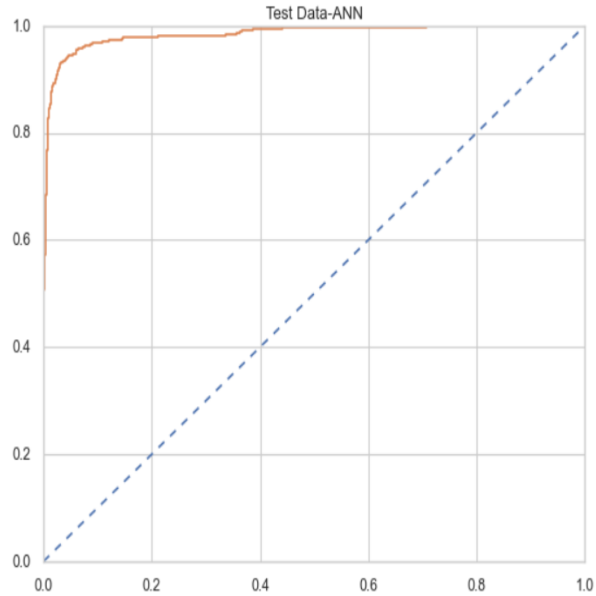
Classification Report:

```
Classification Report of the training data ANN:

              precision    recall  f1-score   support

           0       1.00      1.00      1.00      6406
           1       1.00      0.99      0.99      1294

    accuracy                           1.00      7700
   macro avg       1.00      0.99      0.99      7700
weighted avg       1.00      1.00      1.00      7700


Classification Report of the test data ANN:

              precision    recall  f1-score   support

           0       0.98      0.98      0.98      2743
           1       0.90      0.89      0.90       558

    accuracy                           0.97      3301
   macro avg       0.94      0.94      0.94      3301
weighted avg       0.97      0.97      0.97      3301
```

```
AUC: 1.000

Text(0.5, 1.0, 'Train Data-ANN')
```




```
AUC: 0.985

Text(0.5, 1.0, 'Test Data-ANN')
```



Training and Test set results are almost similar, the model is performing exceptionally well on train data set. It is a over fit model.

Regularization of model:

Model Tuning helps in maximizing a model's performance without over fitting or creating too high of a v ariance . This can be accomplished by selecting appropriate hyper parameters. GridSearch CV is one o f the hyper parameter tuning method to select best parameters.

Applying Gridsearch CV on different models :

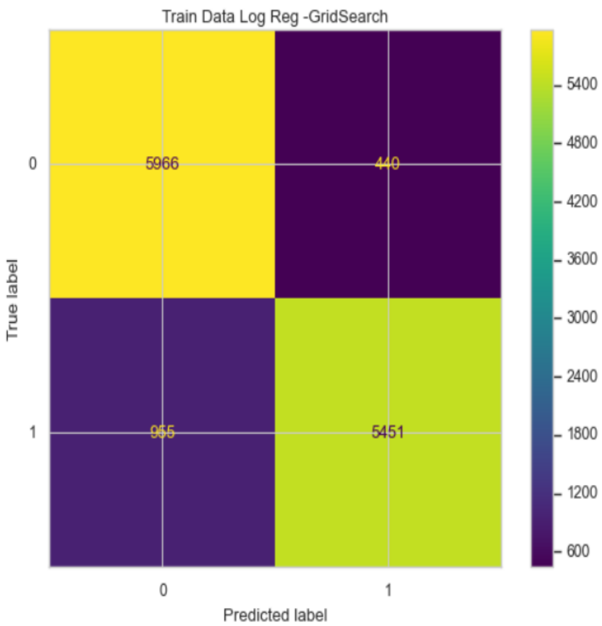
Logistic Regression:

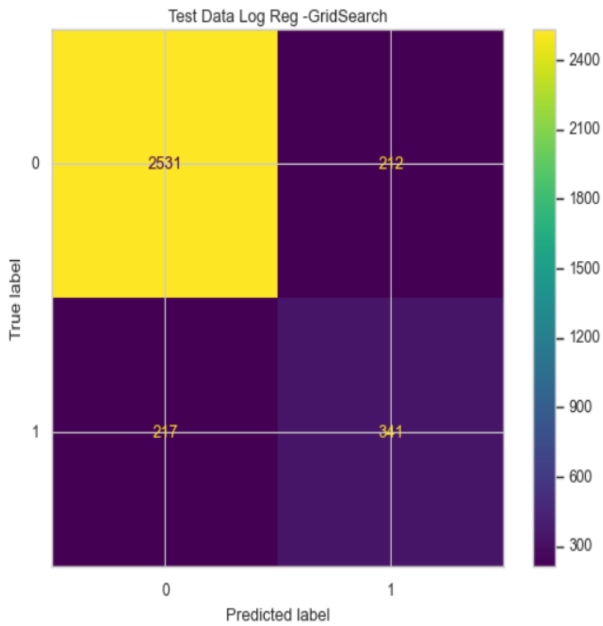
Best parameters :

```
{'penalty': 'none', 'solver': 'lbfgs', 'tol': 0.0001}

LogisticRegression(max_iter=10000, n_jobs=2, penalty='none')
```

	Train data	Test data
Model Score	0.89	0.87
Recall	0.85	0.61
F1 Score	0.89	0.61
AUC	0.953	0.857

Confusion matrix:





Classification Report:

```
Classification Report of the training data GS-Log:

              precision    recall  f1-score   support

           0       0.86      0.93      0.90      6406
           1       0.93      0.85      0.89      6406

    accuracy                           0.89     12812
   macro avg       0.89      0.89      0.89     12812
weighted avg       0.89      0.89      0.89     12812
```

```
Classification Report of the test data GS-Log:

              precision    recall  f1-score   support

           0       0.92      0.92      0.92      2743
           1       0.62      0.61      0.61       558

    accuracy                           0.87      3301
   macro avg       0.77      0.77      0.77      3301
weighted avg       0.87      0.87      0.87      3301
```

The model is performing well on Training set but is not performing well on test set. It is a overfit model.

Feature Importance:

```
The coefficient for Tenure is -0.18122868225864117
The coefficient for City_Tier is 0.13097066760468926
The coefficient for CC_Contacted_LY is 0.022337917905443355
The coefficient for Service_Score is -0.18453380984244705
The coefficient for Account_user_count is 0.29939399554147533
The coefficient for account_segment is -0.08629596691561105
The coefficient for CC_Agent_Score is 0.20850700675670972
The coefficient for rev_per_month is 0.116114213357057 8
The coefficient for Complain_ly is 1.4795057835870102
The coefficient for rev_growth_yoy is -0.03950827089135866 6
The coefficient for coupon_used_for_payment is 0.11122551679580876
The coefficient for Day_Since_CC_connect is -0.05598044707942025
The coefficient for cashback is 0.00050637880720528 15
The coefficient for Payment_Cash on Delivery is -11.385039234120862
The coefficient for Payment_Credit Card is -12.099757612205762
The coefficient for Payment_Debit Card is -11.78281862470355
The coefficient for Payment_E wallet is -11.331985196441687
The coefficient for Payment_UPI is -12.035982341658569
The coefficient for Gender_Female is -11.481095441401798
The coefficient for Gender_Male is -11.237855244538006
The coefficient for Marital_Status_Divorced is -11.843447790169861
The coefficient for Marital_Status_Married is -11.892270902250262
The coefficient for Marital_Status_Single is -10.865354071329962
The coefficient for Login_device_Computer is -10.0474207688897
The coefficient for Login_device_Mobile is -10.497131248959416
The coefficient for Login_device_Others is -11.071715126878265
```

From above results , The most **important feature** is **'Payment_Credit Card'.**

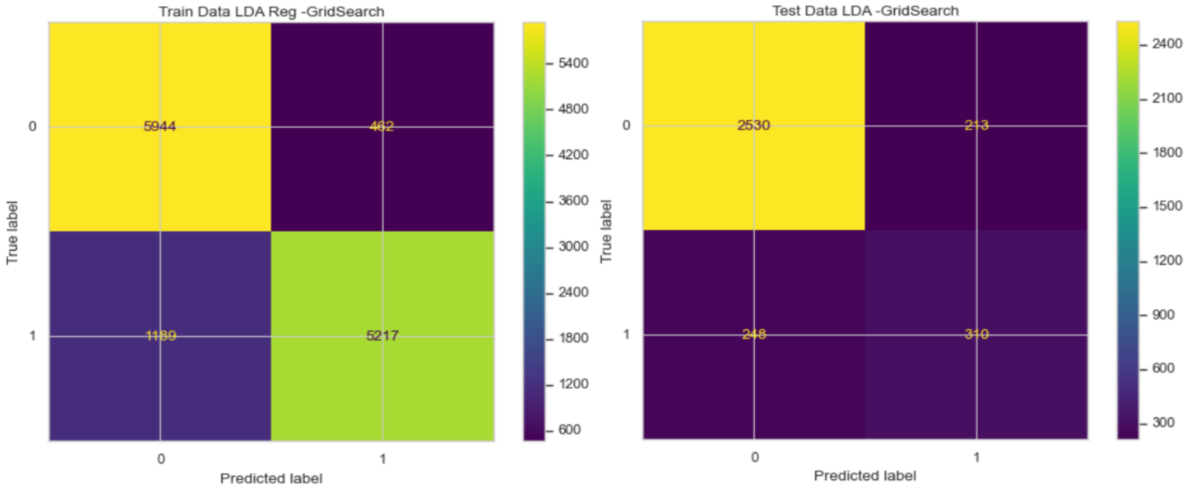
Linear Discriminant Analysis:

Best parameters :

```
{'shrinkage': 'auto', 'solver': 'lsqr', 'tol': 0.0001}

LinearDiscriminantAnalysis(shrinkage='auto', solver='lsqr')
```

	Train data	Test data
Model Score	0.87	0.86
Recall	0.81	0.56
F1 Score	0.86	0.57
AUC	0.941	0.847

Confusion matrix:



Classification Report:

```
Classification Report of the training data GS-LDA:

              precision    recall  f1-score   support

           0       0.83      0.93      0.88      6406
           1       0.92      0.81      0.86      6406

    accuracy                           0.87     12812
   macro avg       0.88      0.87      0.87     12812
weighted avg       0.88      0.87      0.87     12812

Classification Report of the test data GS-LDA:

              precision    recall  f1-score   support

           0       0.91      0.92      0.92      2743
           1       0.59      0.56      0.57       558

    accuracy                           0.86      3301
   macro avg       0.75      0.74      0.75      3301
weighted avg       0.86      0.86      0.86      3301
```

The model is performing well on Training set but is not performing well on test set.

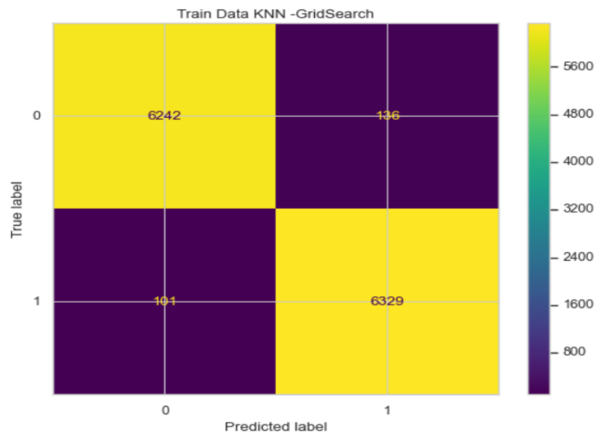
KNN Classifier:

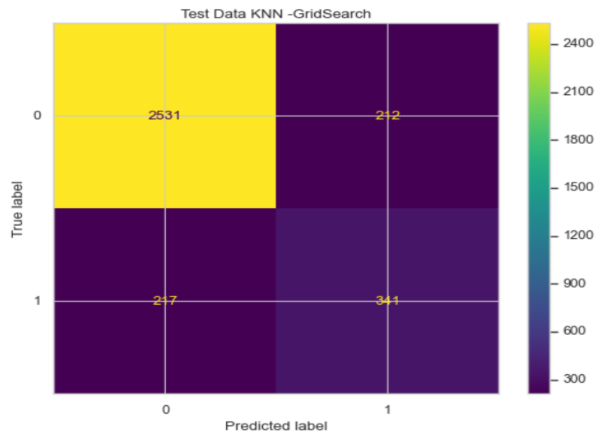
Best Parameters:

```
{'n_neighbors': 5}

KNeighborsClassifier()
```

	Train data	Test data
Model Score	0.98	0.97
Recall	0.98	0.97
F1 Score	0.98	0.97
AUC	0.999	0.995

Confusion Matrix:





Classification Report:

```
Classification Report of the training data KNN-GS:

              precision    recall  f1-score   support

           0       0.98      0.98      0.98      6378
           1       0.98      0.98      0.98      6430

    accuracy                           0.98     12808
   macro avg       0.98      0.98      0.98     12808
weighted avg       0.98      0.98      0.98     12808

Classification Report of the test data KNN-GS:

              precision    recall  f1-score   support

           0       0.97      0.97      0.97      2771
           1       0.97      0.97      0.97      2719

    accuracy                           0.97      5490
   macro avg       0.97      0.97      0.97      5490
weighted avg       0.97      0.97      0.97      5490
```

The model is performing very well on both Training set and testing set.

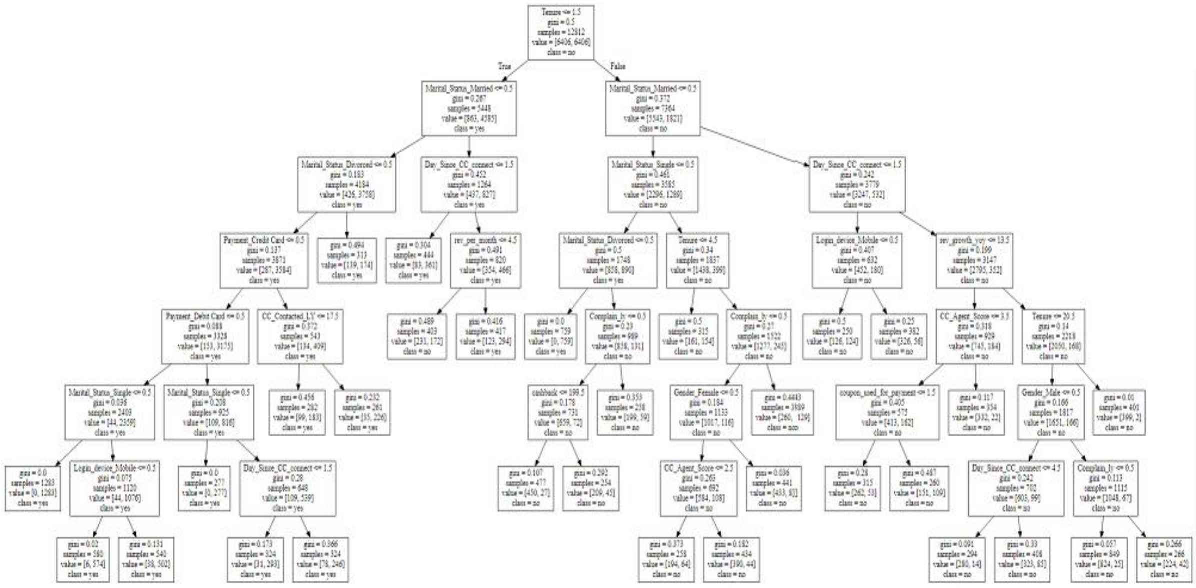
Decison Tree:

Best Parameters:

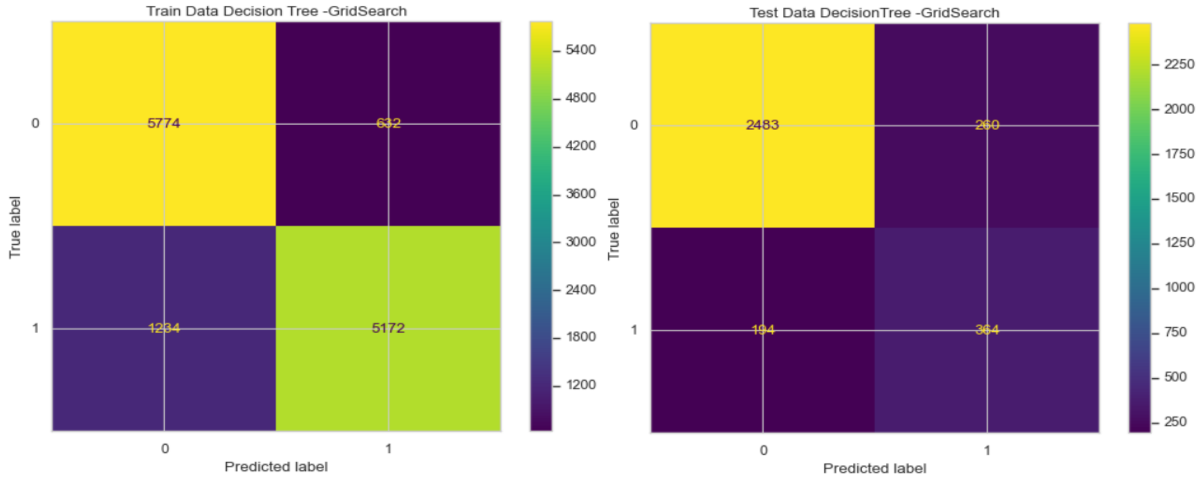
```
{'max_depth': 7, 'min_samples_leaf': 250, 'min_samples_split': 390}

DecisionTreeClassifier(max_depth=7, min_samples_leaf=250, min_samples_split=390)
```

	Train data	Test data
Model Score	0.85	0.86
Recall	0.81	0.65
F1 Score	0.85	0.62
AUC	0.934	0.856

Decision Tree:



Confusion Matrix:



Classification Report:

```
Classification Report of the training data Decision Tree:

              precision    recall  f1-score   support

           0       0.82      0.90      0.86      6406
           1       0.89      0.81      0.85      6406

    accuracy                           0.85     12812
   macro avg       0.86      0.85      0.85     12812
weighted avg       0.86      0.85      0.85     12812

Classification Report of the test data Decision Tree:

              precision    recall  f1-score   support

           0       0.93      0.91      0.92      2743
           1       0.58      0.65      0.62       558

    accuracy                           0.86      3301
   macro avg       0.76      0.78      0.77      3301
weighted avg       0.87      0.86      0.87      3301
```

The model is performing well on Training set but is not performing well on test set.

Feature Importance:

	Imp
Tenure	0.597902
Marital_Status_Divorced	0.190882
Marital_Status_Married	0.076706
Marital_Status_Single	0.043538
Day_Since_CC_connect	0.021476
Complain_ly	0.011464
Login_device_Mobile	0.010235
Payment_Credit Card	0.009903
rev_per_month	0.008347
CC_Agent_Score	0.007390
rev_growth_yoy	0.005154
coupon_used_for_payment	0.004720
Payment_Debit Card	0.003481
CC_Contacted_LY	0.003357
Gender_Female	0.002696
Gender_Male	0.001485
cashback	0.001267
Login_device_Computer	0.000000
Payment_Cash on Delivery	0.000000
Payment_UPI	0.000000
Payment_E wallet	0.000000
City_Tier	0.000000
account_segment	0.000000
Account_user_count	0.000000
Service_Score	0.000000
Login_device_Others	0.000000

From above results , The most **important feature** is **'Tenure'.**

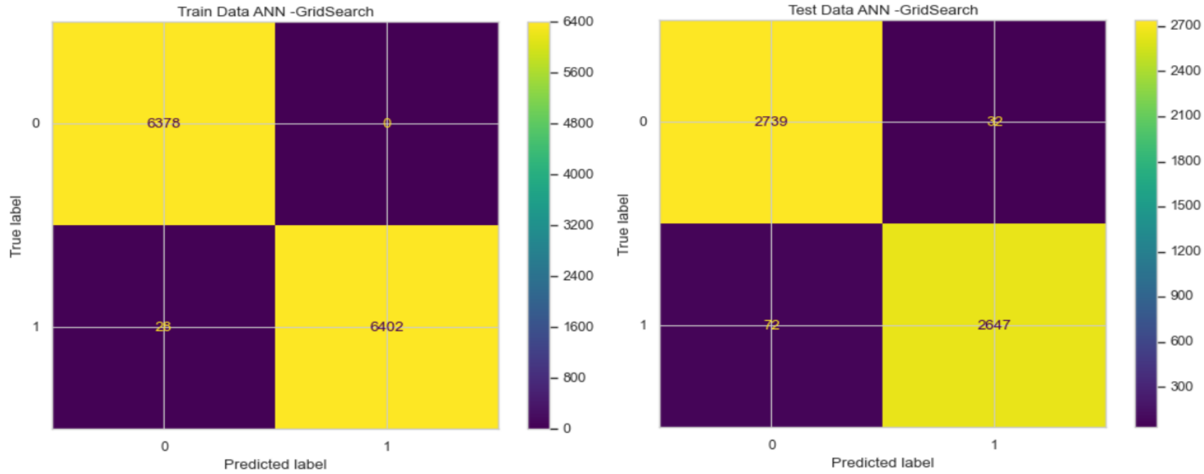
Artificial Neural Network:

Best Parameters:

```
{'activation': 'relu', 'hidden_layer_sizes': (100, 100, 100), 'max_iter': 10000, 'solver': 'adam', 'tol': 0.01}

MLPClassifier(hidden_layer_sizes=(100, 100, 100), max_iter=10000, tol=0.01)
```

	Train data	Test data
Model Score	1	0.59
Recall	1	0.94
F1 Score	1	0.69
AUC	1	0.997

Confusion Matrix:



Classification Report:

```
Classification Report of the training data ANN-GS:

              precision    recall  f1-score   support

           0       1.00      1.00      1.00      6378
           1       1.00      1.00      1.00      6430

    accuracy                           1.00     12808
   macro avg       1.00      1.00      1.00     12808
weighted avg       1.00      1.00      1.00     12808

Classification Report of the test data ANN-GS:

              precision    recall  f1-score   support

           0       0.81      0.23      0.36      2771
           1       0.55      0.94      0.69      2719

    accuracy                           0.59      5490
   macro avg       0.68      0.59      0.53      5490
weighted avg       0.68      0.59      0.53      5490
```

The model is performing well on Training set but is not performing well on test set. It is a overfit model.

Applying Gridsearch CV on Random Forest Model:

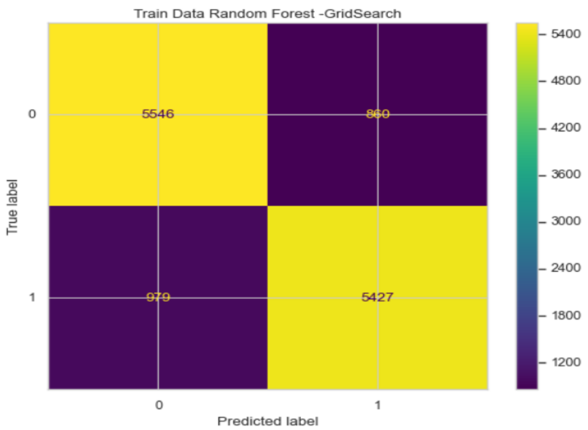
Best Parameters:

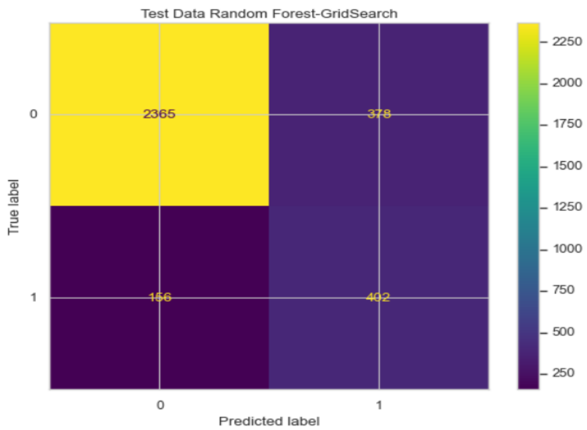
```
{'max_depth': 7, 'max_features': 12, 'min_samples_leaf': 150, 'min_samples_split': 450, 'n_estimators': 101}
```

```
RandomForestClassifier(max_depth=7, max_features=12, min_samples_leaf=150,
                       min_samples_split=450, n_estimators=101)
```

	Train data	Test data
Model Score	0.86	0.84
Recall	0.85	0.72
F1 Score	0.86	0.6
AUC	0.943	0.885

Confusion Matrix:





Classification Report:

```
Classification Report of the training data RF-GS:
              precision    recall  f1-score   support

           0       0.85      0.87      0.86      6406
           1       0.86      0.85      0.86      6406

    accuracy                           0.86     12812
   macro avg       0.86      0.86      0.86     12812
weighted avg       0.86      0.86      0.86     12812

Classification Report of the test data RF-GS:
              precision    recall  f1-score   support

           0       0.94      0.86      0.90      2743
           1       0.52      0.72      0.60       558

    accuracy                           0.84      3301
   macro avg       0.73      0.79      0.75      3301
weighted avg       0.87      0.84      0.85      3301
```

The model is performing exceptionally well on Training set but is not performing well on test set.

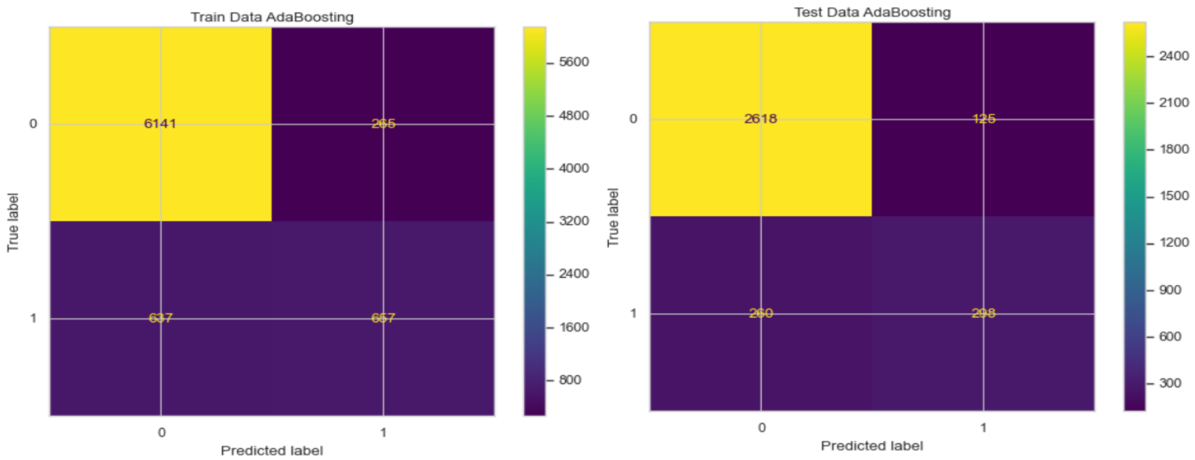
Ensemble- Ada Boosting

In Adaptive Boosting the weights are re-assigned to each instance, with higher weights to incorrectly classified instances. Boosting is used to reduce bias as well as the variance for supervised learning. It works on the principle where learners are grown sequentially. Except for the first, each subsequent learner is grown from previously grown learners

```
AdaBoostClassifier(n_estimators=10, random_state=1)
```

	Train data	Test data
Model Score	0.88	0.88
Recall	0.51	0.53
F1 Score	0.59	0.61
AUC	0.891	0.881

Confusion Matrix:



Classification Report:

```
Classification Report of the training data AdaBoosting:

              precision    recall  f1-score   support

           0       0.91      0.96      0.93      6406
           1       0.71      0.51      0.59      1294

    accuracy                           0.88      7700
   macro avg       0.81      0.73      0.76      7700
weighted avg       0.87      0.88      0.87      7700

Classification Report of the test data AdaBoosting:

              precision    recall  f1-score   support

           0       0.91      0.95      0.93      2743
           1       0.70      0.53      0.61       558

    accuracy                           0.88      3301
   macro avg       0.81      0.74      0.77      3301
weighted avg       0.87      0.88      0.88      3301
```

The model is performing similar on both Training set and testing set. But Overall recall is not high.

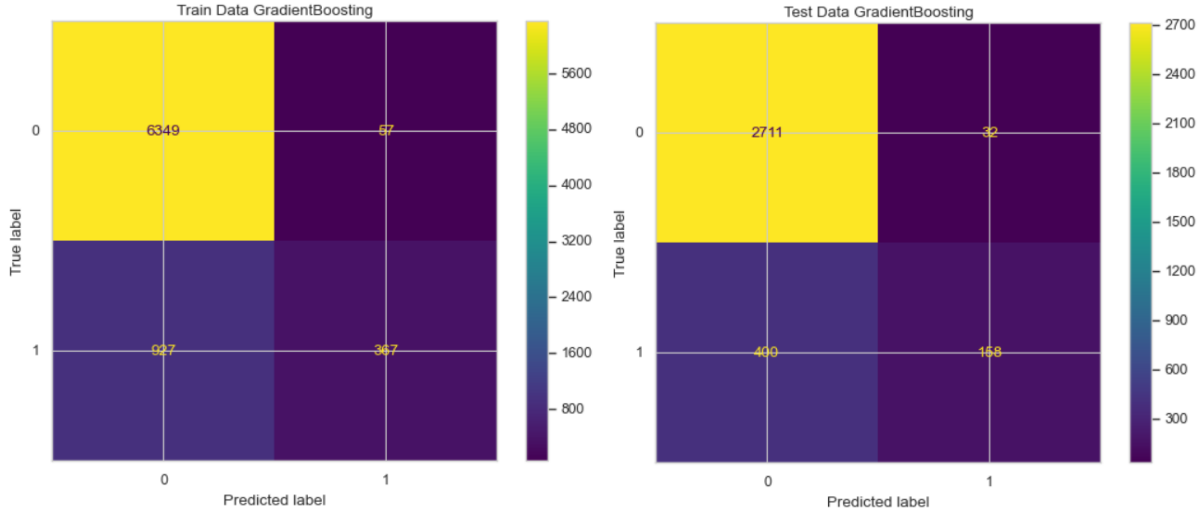
Ensemble- Gradient Boosting

Gradient Boosting is used for regression as well as classification tasks. This approach trains learners based upon minimising the loss function of a learner (i.e., training on the residuals of the model).

```
GradientBoostingClassifier(n_estimators=10, random_state=1)
```

	Train data	Test data
Model Score	0.87	0.87
Recall	0.28	0.28
F1 Score	0.43	0.42
AUC	0.884	0.873

Confusion Matrix:



Classification Report:

```
Classification Report of the training data GradientBoosting:

              precision    recall  f1-score   support

           0       0.87      0.99      0.93      6406
           1       0.87      0.28      0.43      1294

    accuracy                           0.87      7700
   macro avg       0.87      0.64      0.68      7700
weighted avg       0.87      0.87      0.84      7700

Classification Report of the test data GradientBoosting:

              precision    recall  f1-score   support

           0       0.87      0.99      0.93      2743
           1       0.83      0.28      0.42       558

    accuracy                           0.87      3301
   macro avg       0.85      0.64      0.67      3301
weighted avg       0.86      0.87      0.84      3301
```

The model is performing similar on both Training set and testing set. But Overall recall is not high.

05.Model Validation.

Model Selection:

- Logistic Regression:

Basic Model -Without Hyperparameter Tuning		
	Train data	Test data
Model Score	0.88	0.88
Recall	0.46	0.45
F1 Score	0.56	0.56
AUC	0.873	0.859

Regularized Model -With Hyperparameter Tuning		
	Train data	Test data
Model Score	0.89	0.87
Recall	0.85	0.61
F1 Score	0.89	0.61
AUC	0.953	0.857

Regularized model has overall higher performance measures.

- Linear Disriminant Analysis:

Basic Model -Without Hyperparameter Tuning		
	Train data	Test data
Model Score	0.88	0.87
Recall	0.43	0.43
F1 Score	0.54	0.54
AUC	0.863	0.85

Regularized Model -With Hyperparameter Tuning		
	Train data	Test data
Model Score	0.87	0.86
Recall	0.81	0.56
F1 Score	0.86	0.57
AUC	0.941	0.847

Regularized model has overall higher performance measures

- KNN Classifier:

Basic Model -Without Hyperparameter Tuning		
	Train data	Test data
Model Score	0.97	0.95
Recall	0.87	0.78
F1 Score	0.91	0.83
AUC	0.994	0.974

Regularized Model -With Hyperparameter Tuning		
	Train data	Test data
Model Score	0.98	0.97
Recall	0.98	0.97
F1 Score	0.98	0.97
AUC	0.999	0.995

Regularized model has overall higher performance measures.

- Decision Tree Classifier:

Basic Model -Without Hyperparameter Tuning		
	Train data	Test data
Model Score	1	0.95
Recall	1	0.87
F1 Score	1	0.86
AUC	1	0.919

Regularized Model -With Hyperparameter Tuning		
	Train data	Test data
Model Score	0.85	0.86
Recall	0.81	0.65
F1 Score	0.85	0.62
AUC	0.934	0.856

Basic model is overfitting and Regularized model has low recall and f1 score.

- Artificial Neural Network:

Basic Model -Without Hyperparameter Tuning		
	Train data	Test data
Model Score	1	0.97
Recall	0.99	0.89
F1 Score	0.99	0.9
AUC	1	0.985

Regularized Model -With Hyperparameter Tuning		
	Train data	Test data
Model Score	1	0.59
Recall	1	0.94
F1 Score	1	0.69
AUC	1	0.997

Both Basic model and and Regularized model are overfitting.

- Random Forest:

Basic Model -Without Hyperparameter Tuning		
	Train data	Test data
Model Score	1	0.97
Recall	1	0.85
F1 Score	1	0.91
AUC	1	0.994

Regularized Model -With Hyperparameter Tuning		
	Train data	Test data
Model Score	0.86	0.84
Recall	0.85	0.72
F1 Score	0.86	0.6
AUC	0.943	0.885

Basic model is overfitting and Regularized model has low recall and f1 score.

- AdaBoosting:

Basic Model -Without Hyperparameter Tuning		
	Train data	Test data
Model Score	0.88	0.88
Recall	0.51	0.53
F1 Score	0.59	0.61
AUC	0.891	0.881

Regularized Model -With Hyperparameter Tuning		
	Train data	Test data
Model Score	0.85	0.84
Recall	0.82	0.72
F1 Score	0.85	0.61
AUC	0.917	0.853

Regularized model has overall higher performance measures.

- Gradient Boosting:

Basic Model -Without Hyperparameter Tuning		
	Train data	Test data
Model Score	0.87	0.87
Recall	0.28	0.28
F1 Score	0.43	0.42
AUC	0.884	0.873

Regularized Model -With Hyperparameter Tuning		
	Train data	Test data
Model Score	0.85	0.84
Recall	0.82	0.72
F1 Score	0.85	0.61
AUC	0.919	0.861

Regularized model has overall higher performance measures.

Summarizing overall model results:

Test Data Performnace metrices	Logistic Regression	Linear Disriminant Analysis	KNN Classifier	Random Forest	AdaBoosting	Gradient Boosting
Model Score	0.87	0.86	0.97	0.84	0.84	0.84
Recall	0.61	0.56	0.97	0.72	0.72	0.72
F1 Score	0.61	0.57	0.97	0.6	0.61	0.61
AUC	0.857	0.847	0.995	0.885	0.853	0.861

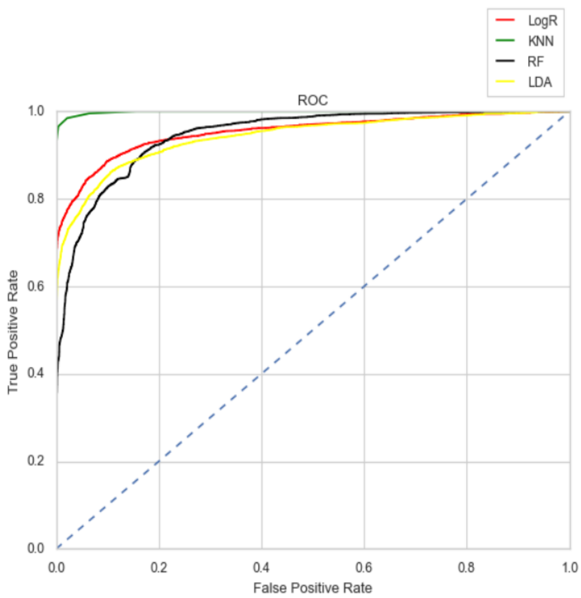
Looking at the details got from **test data** from these models ,

Accuracy : KNN Classifier model has highest value of 0.97

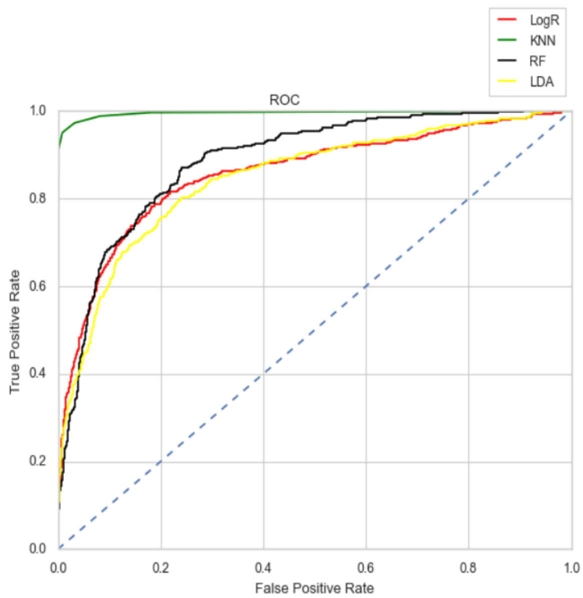
AUC : KNN Classifier model has highest value of 0.995 and LDA model has least value of 0.847

Recall : KNN Classifier model has highest value of 0.97 and LDA model has least value of 0.56

F1 Score : KNN Classifier model has highest value of 0.97 and LDA model has least value of 0.57



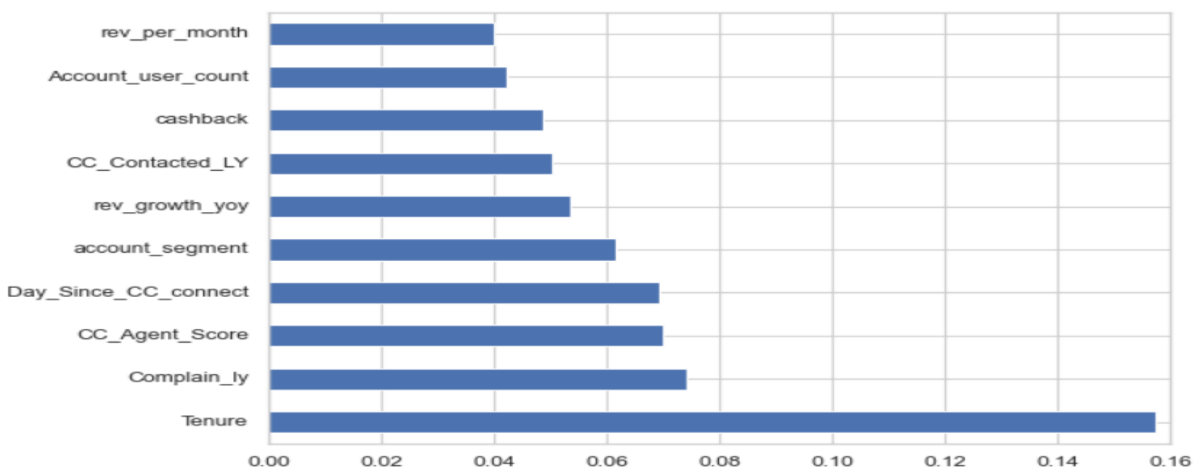
Train Data



Test Data

The overall measures are high in KNN Classifier model. Therefore, **KNN Classifier model has best performance among all the models with 'Recall'and 'F1 Score'.**

From the overall results of these models, the variable **'Tenure'** is found to be the most useful feature amongst all other features for predicting the churn status.



KNN Model is reasonably stable enough to be used for making any future predictions. Also this Model is simple and easy to implement. There is no requirement of making many assumptions while model building.

KNN gives a probability of a particular customer churning. The threshold is usually set to .5 by default. This means that anyone with a probability of more than .5 is predicted to churn. If you reduce the probability threshold, more people will be predicted to churn, this gives you a higher number of “at risk customers” to target. However, this increases the likelihood that customers who are not at risk will pass the threshold and be predicted to churn.

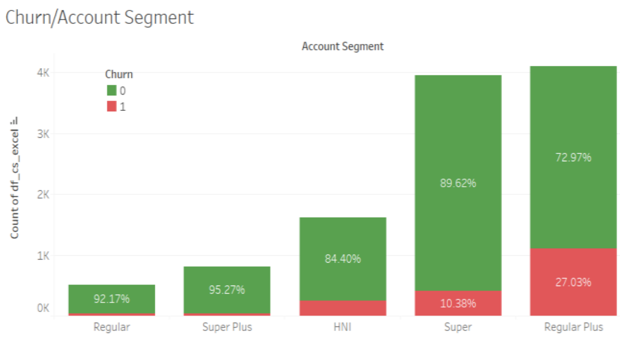
The choice of the probability threshold can be set based on the business requirement, if the company wants to target a large amount of customers then a low threshold will be set. However, if the company wants to be more efficient in spending a higher threshold will be set, at the cost of a smaller number of customers to target. This can be checked by looking at the **Recall score (TP/TP+FN**) and **F1 score((2*Precision*Recall)/(Precision+Recall)).**

Due to the importance of understanding and managing the risks in volatile business domains, it is required to find an effective aid in making decisions. The results from models show that KNN Classifier algorithm is a promising opportunity in predicting customer churn status for the given data set.

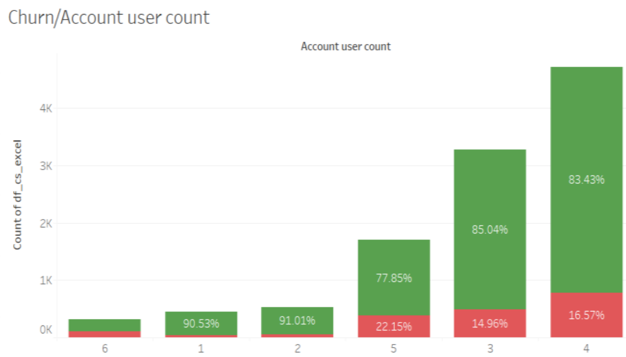
With this model, we can predict about the customers who are at risk of churning.

06.Final Interpretation and Recommendation.

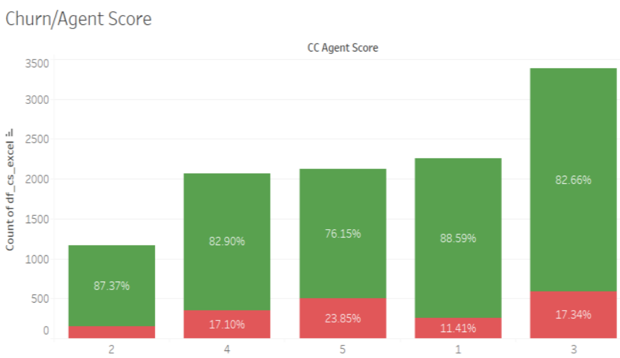
On observing different independent variables with respect to output variable ‘Churn’. We have got the following info :



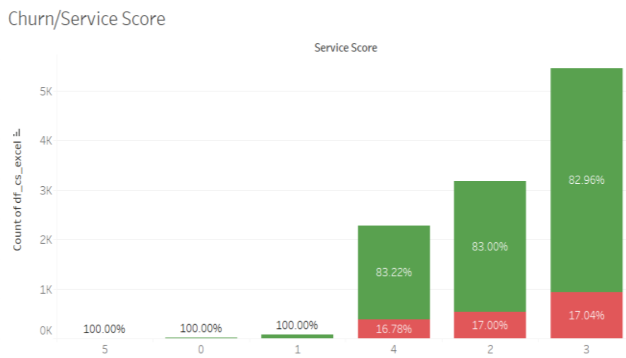
There is High Churn Rate in **account segment** of **'Regular Plus'**



There is High Churn Rate in **account with user count** of **'5'**

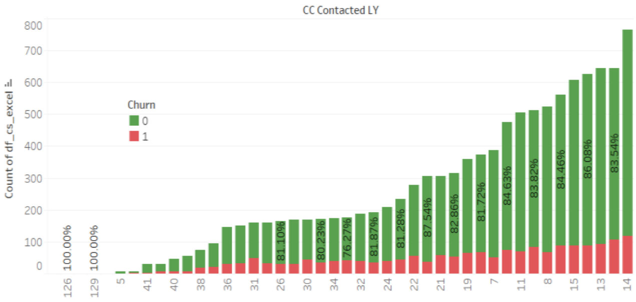


There is High Churn Rate in **CC Agent Score** of **'5'**



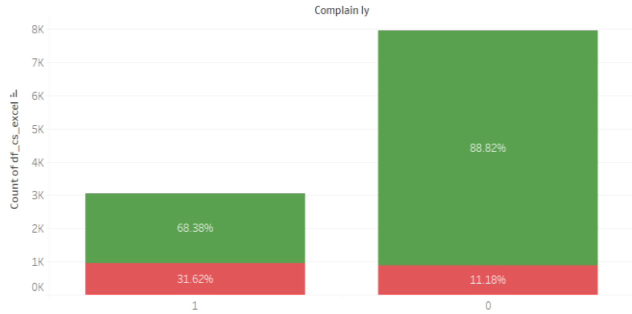
There is High Churn Rate in **Service Score** of **'3'**

Churn/CC Contacted LY



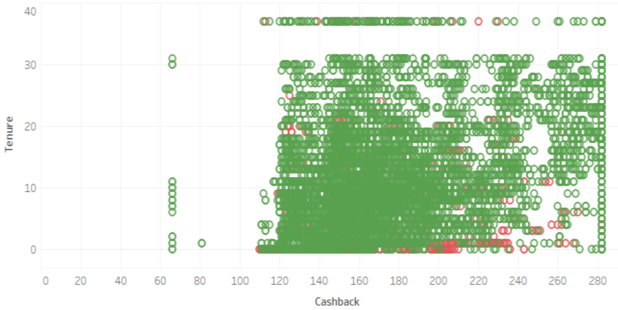
There is High Churn Rate in **CC Contacted LY** of '31'

Churn/Complian ly



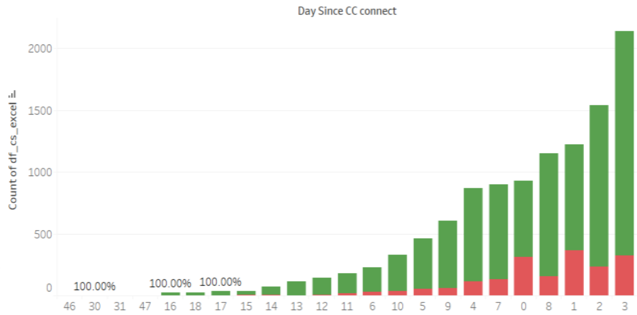
There is High Churn Rate in **Complain ly** of '1'

Tenure vs Cash back



Majority of customers are in Tenure (0-20) and Cashback (110-200). There is higher churn rate in low Tenure.

Churn/Days Since CC Connect



There is High Churn Rate in **Days Since CC Connect** of '0'

Segregating customers based on probability of churn, Revenue and Tenure . We have got the following Matrix:

Risk-Value-Tenure Matrix				
Risk (Prob)	Revenue Value	Account Tenure		
		VeryOld	Old	Latest
VeryHigh (>0.8)	VeryHigh	49	16	64
	High	56	15	83
	Medium	93	31	104
	Low	190	38	174
High (0.8-0.6)	VeryHigh	111	126	191
	High	112	80	165
	Medium	149	119	210
	Low	361	192	430
Moderate(0.5-0.6)	VeryHigh	20	33	13
	High	11	25	24
	Medium	25	60	74
	Low	54	108	156

- We can never afford to lose the customers who are providing **Very High Revenue and Very Old Tenure account**. They are the **most loyal and high monetary** value customers.
- Different strategies can be applied to retain the customers based on value of the customer.
- **Red color** Marked customers are **most important** , followed by Blue and Brown. As majority of the business happens through these customers we have to make **customer specific retention efforts.**
- **Budget preference** should be based on **High Value-High Risk**

Recommendations:

- Majority of the customers belongs to segment – **'Regular Plus' & 'Super'** and also associated with **3-4** persons per account. So, we can include **more channels** in these segments to cover **wide range of audience**.
- Most of the customers are using **mobile** as Login device. So, we can allow **multiple login** for single account through **discounted pricing**. This would help in retaining customers.
- Most of the **Payment** is done through **Credit and Debit Cards**. We can attract customers for **higher recharge** values through **reward points and high cash back** for payments.
- Majority of the customers gave an **average score** on both **Service score** and **Customer service score**. Its preferable to get **customer feedback through text** by providing **certain questionnaire**. This would be helpful in understanding the exact problem and then we can act accordingly.
- Customer **complaints** must be solved in **minimal** time. **Customer service** should be **very effective** to make it easy for the customer as per his requirements.
- There is **high churn** rate in **low tenure or recent** accounts. We can provide offers such as **Welcome bundle** etc., for first **3-6** months. So that customer will get to know about various options available for him.
- **Retention Marketing**: The at **risk customers** can be specially added to retention marketing lists so that **sufficient measures** can be used for them **specifically**.
- **Discounts or other incentives** can be offered to at risk customers to try to retain them.
- Offering Special deals such **OTT Subscriptions** could be given to retain customers.
- Proactively **emailing or calling** at-risk customers to understand their requirements
- Customers with a **low probability of churning** can be removed from **re-targeting lists**, this could lead to **cost saving in marketing.**