CAPSTONE PROJECT
Customer Churn – DTH (CC_EDTH_02)

PREEJA RAJESH

PGP – DSBA

2020 - 2021

TABLE OF CONTENTS

1. Introduction of the business problem

Prediction whether a customer is going to churn based on the various usage parameters of the DTH company.

Dataset: Customer Churn Data.xlsx

1.1 Defining problem statement.

The data set belongs to a leading DTH company. The company wants to know the customers who are going to churn, so accordingly they can approach customer to offer some promos.

1.2 Need of the study/project

Customer churn is a metric that no one really wants to have but everyone needs to have. Since churn is the antithesis of retention, it not only affects the size of your customer base, but directly impacts your customer lifetime value.

Churn rate is a health indicator for businesses, especially the impact of retention efforts, and is pivotal to grow your business. Of course, some natural churn is inevitable, and the figure differs from industry to industry. But having a higher churn figure than that is a definite sign that a business is doing something wrong. Churn rates do correlate with lost revenue and increased acquisition spend. In addition, they play a vital role in a company's growth potential.

There are several channels and cable connections, some of them are hard to differentiate as they sell similar kind of products. Here DTH businesses will need to think how they can keep their customers engaged with their connection.

1.3 Constraints

In this company, account churn is a major issue because 1 account can have multiple customers. hence by losing one account the company might be losing more than one customer.

Another constrain is customers leaving without voicing their complaints.

The scope of this project is to identify factors contributing to customer attrition and thereby recommend strategies that may help in regain trust and improve the overall customer satisfaction levels.

1.4 Understanding business/social opportunity

Customer churn definition is also perceived as the term “Customer Attrition”, customer churn is a crucial metric unit since it is much cheaper to retain existing customers than it is to win new ones – which means working with potential leads all the way through the entire process of sales funnel. The term “Customer Retention”, on the contrary, is regularly more cost-effective as you have gained the loyalty and trust of existing customers already.

There are several benefits of having loyal customers -

1. Having a solid number for existing customers, it helps businesses to expand their market.
2. Customers appreciate your marketing strategy and are ready to try new things.
3. Real time feedback received from the customers.
4. Existing customers bring more new customers, they are the best source of marketing.
5. Customer retention also help in attracting new customers. Seeing a company give rewards and extra benefits to their existing customers, it attracts more people.

2. Data Report

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

As a part of the course the data was provided by the Institution for the capstone project for DTH Customer Churn.

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

	AccountID	Churn	Tenure	City_Tier	CC_Contacted_LY	Payment	Gender	Service_Score	Account_user_count	account_segment	CC_Agent_Score	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	

- The number of rows of the dataframe is 11260.
- The number of columns of the dataframe is 19.

2.3 Five Point Summary

	count	mean	std	min	25%	50%	75%	max
AccountID	11260.0	25629.500000	3250.626350	20000.0	22814.75	25629.5	28444.25	31259.0
Churn	11260.0	0.168384	0.374223	0.0	0.00	0.0	0.00	1.0
City_Tier	11148.0	1.653929	0.915015	1.0	1.00	1.0	3.00	3.0
CC_Contacted_LY	11158.0	17.867091	8.853269	4.0	11.00	16.0	23.00	132.0
Service_Score	11162.0	2.902526	0.725584	0.0	2.00	3.0	3.00	5.0
CC_Agent_Score	11144.0	3.066493	1.379772	1.0	2.00	3.0	4.00	5.0
Complain_ly	10903.0	0.285334	0.451594	0.0	0.00	0.0	1.00	1.0

Inferences:

- CC_Contacted_LY is in hundreds and rest all the variables are approximately below 10, so scaling would be required.

2.4 Understanding of attributes (variable info, renaming)

```
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
```

Inferences:

- **Numerical Variables** are -

  - **Discrete:** Account_user_count
  - **Continuous:** AccountID, Tenure, CC_Contacted_LY, rev_per_month, rev_growth_yoy, coupon_used_for_payment, Day_Since_CC_connect and cashback.

- **Ordinal Variables** are Service_Score and CC_Agent_Score
- There are 7 **categorical variables** (City_Tier, Payment, Gender, Account_segment, Marital_Status, Complain_ly and Login_device)

- **Rows have been renamed to maintain unique names for each row.**

  - In Gender attribute 'F' is replaced by 'Female'
  - In Gender attribute 'M' is replaced by 'Male'
  - In Payment attribute 'Cash on Delivery' is replaced by 'COD'
  - In account_segment attribute 'Regular +' is replaced by 'Regular Plus'
  - In account_segment attribute 'Super +' is replaced by 'Super Plus'

❖ **Unique counts of all Nominal Variables**

PAYMENT: 5

UPI	822
COD	1014
E wallet	1217
Credit Card	3511
Debit Card	4587

ACCOUNT_SEGMENT: 5

Regular	520
Super Plus	818
HNI	1639
Super	4062
Regular Plus	4124

Gender: 2

Female	4448
Male	6704

LOGIN_DEVICE: 2

Computer	3018
Mobile	7482

MARITAL_STATUS: 3

Divorced	1668
Single	3520
Married	5860

COMPLAIN_LY: 2

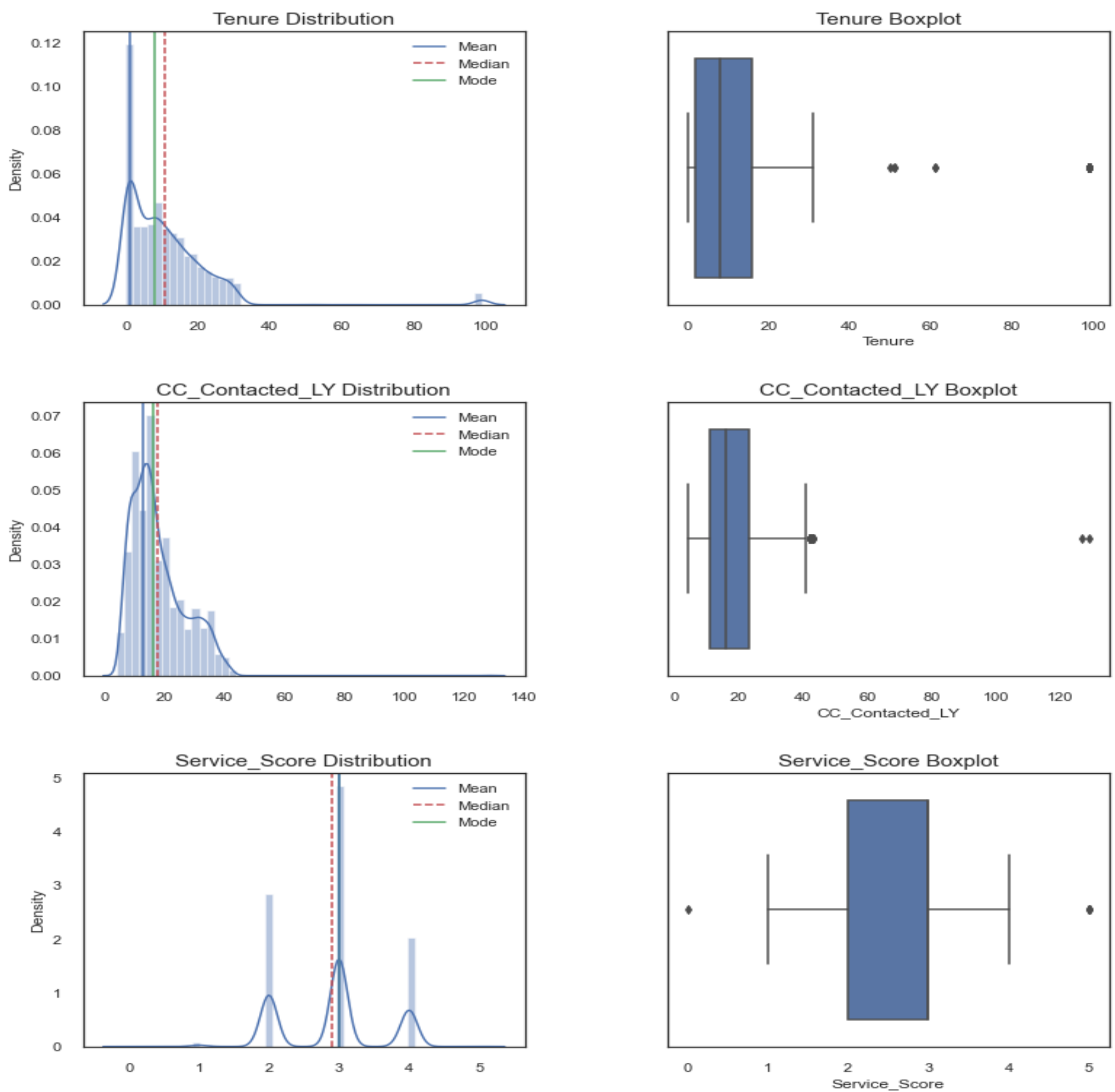
1.0	3111
0.0	7792

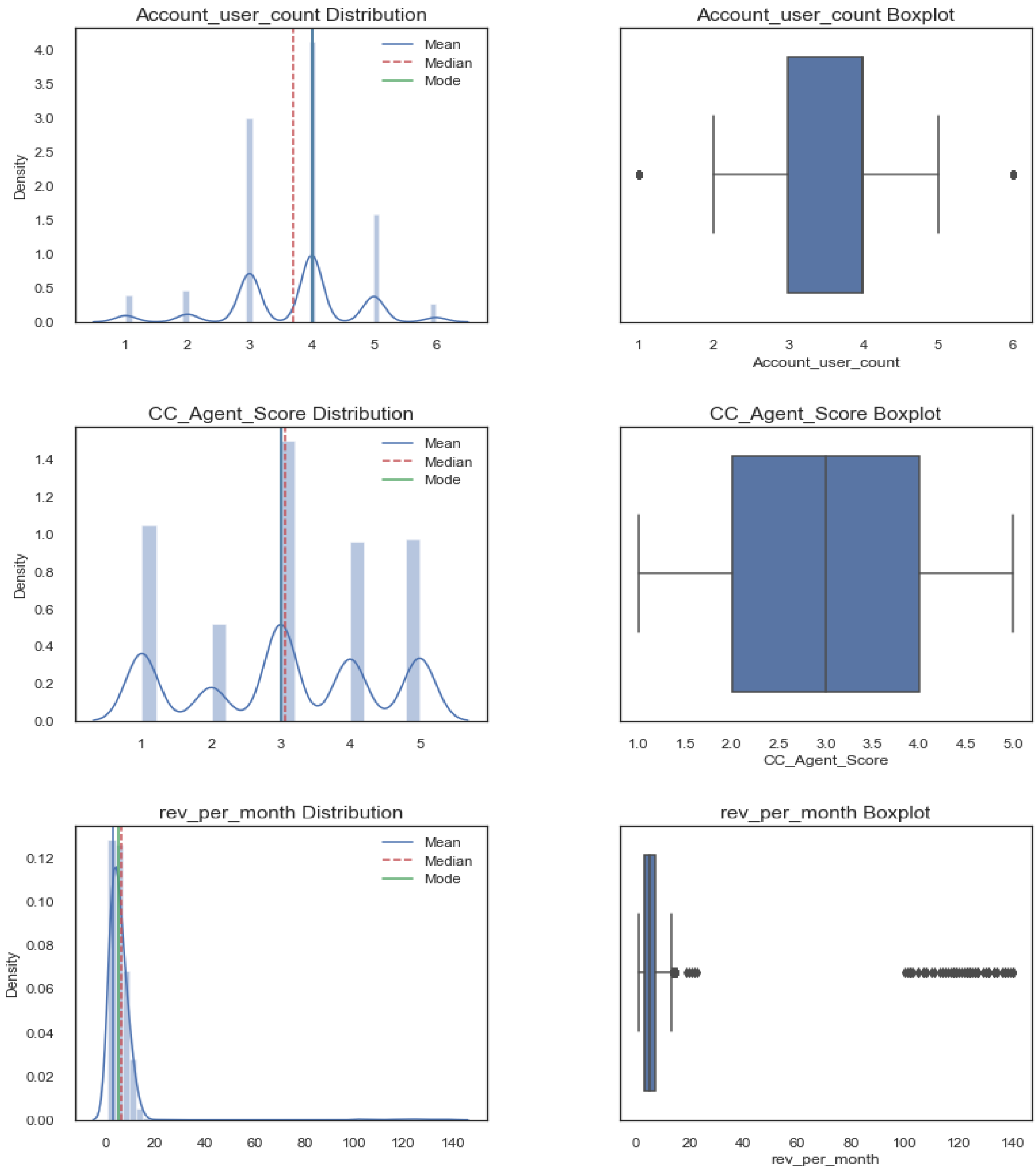
❖ **Dividing the dataset into a separate training and test dataset**

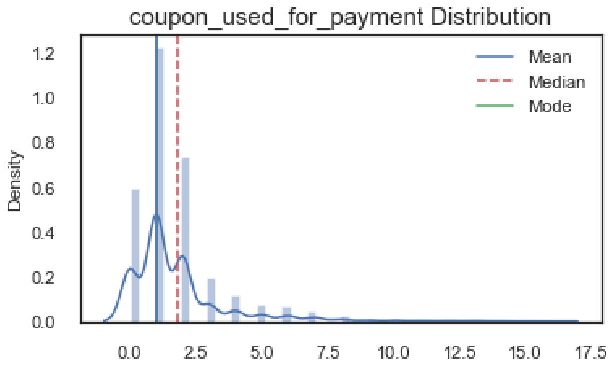
- In this step, we will randomly divide the DTH dataset into a training dataset and a test dataset where the training dataset will contain 67% of the samples and the test dataset will contain 33%, respectively.
- Model will be fitted on train set and predictions will be made on the test set.

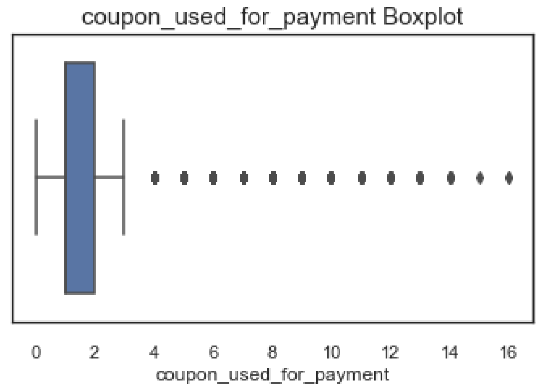
3. Exploratory data analysis

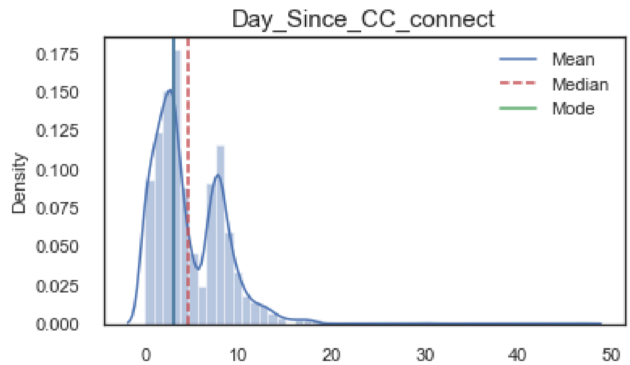
3.1 Univariate analysis (distribution and spread for every continuous attribute, distribution of data in categories for categorical ones)

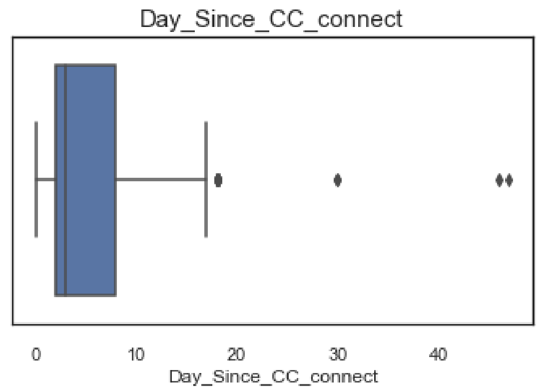


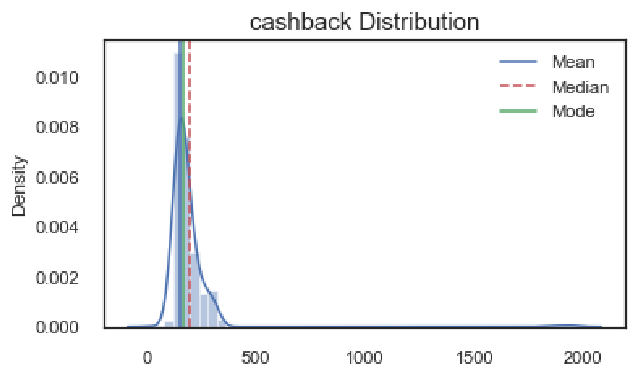

Account_user_count Distribution
Mean
Median
Mode
Density
4.0
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
1
2
3
4
5
6
Account_user_count Boxplot
1
2
3
4
5
6
Account_user_count
CC_Agent_Score Distribution
Mean
Median
Mode
Density
1.4
1.2
1.0
0.8
0.6
0.4
0.2
0.0
1
2
3
4
5
CC_Agent_Score Boxplot
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0
CC_Agent_Score
rev_per_month Distribution
Mean
Median
Mode
Density
0.12
0.10
0.08
0.06
0.04
0.02
0.00
0
20
40
60
80
100
120
140
rev_per_month Boxplot
0
20
40
60
80
100
120
140
rev_per_month

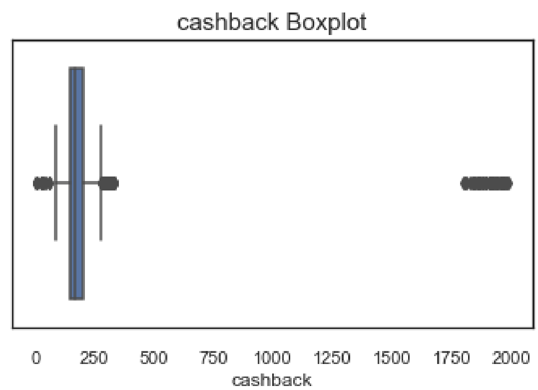

coupon_used_for_payment Distribution
Mean
Median
Mode
Density

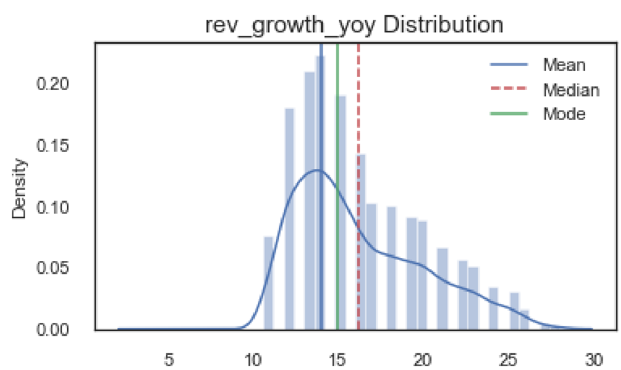

coupon_used_for_payment Boxplot
coupon_used_for_payment

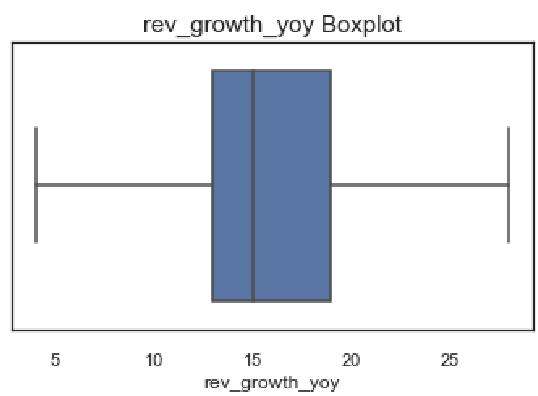

Day_Since_CC_connect
Mean
Median
Mode
Density


Day_Since_CC_connect
Day_Since_CC_connect


cashback Distribution
Mean
Median
Mode
Density


cashback Boxplot
cashback


rev_growth_yoy Distribution
Mean
Median
Mode
Density


rev_growth_yoy Boxplot
rev_growth_yoy

- **Skeweness of every attribute:**

```
1. AccountID                  -0.008665
2. Tenure                      3.912858
3. City_Tier                   0.744004
4. CC_Contacted_LY             1.414220
5. Service_Score              -0.003242
6. Account_user_count         -0.411870
7. CC_Agent_Score             -0.145050
8. rev_per_month               9.361251
9. Complain_ly                 0.963152
10. rev_growth_yoy             0.765604
11. coupon_used_for_payment    2.617799
12. Day_Since_CC_connect       1.325511
13. cashback                   8.798959
14. Churn                      1.773085
```

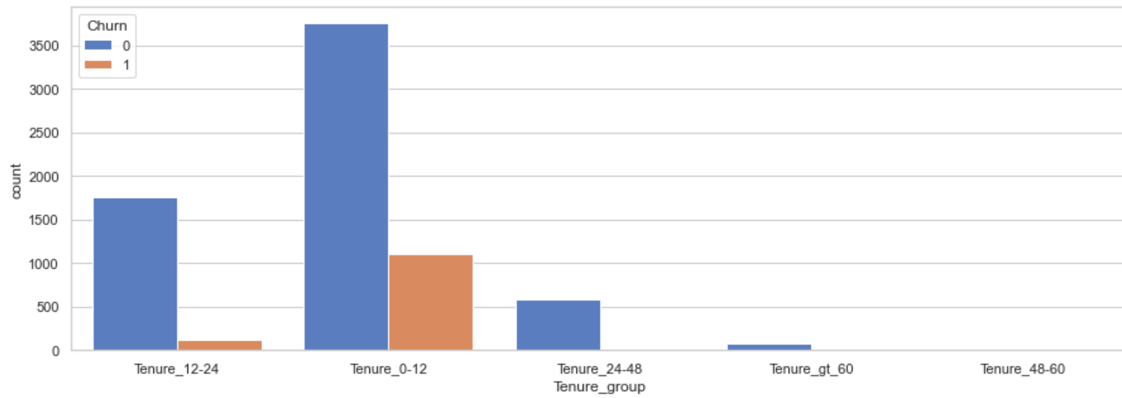
Inferences:

- The output above shows that the variables 'rev_per_month', 'cashback', 'Tenure' and 'cou pon_used_for_payment' has a right-skewed distribution with the skewness values of (9.3, 8.9, 3.8, & 2.5 resp.)
- Ideally, the skewness value should be between -1 and 1. There are many techniques of han dling these extreme values, one of which is quantile-based capping or flooring.

- **Converting Tenure to categorical column**

The tenure of customers is in no of months, we would like to bin it to get insights. We will create fixed-width bins, each bin contains a specific numeric range. Generally, these r anges are manually set, with a fixed size. Here, I have decided to group 12 into 5 bins. [0–12], [12–24], [24–48], [48–60] and [gt_60] are the 5 bins. We cannot have large gaps in the count s because it may create empty bins with no data. This problem is solved by positioning the bin s based on the distribution of the data.

- **Churn Vs Tenure**



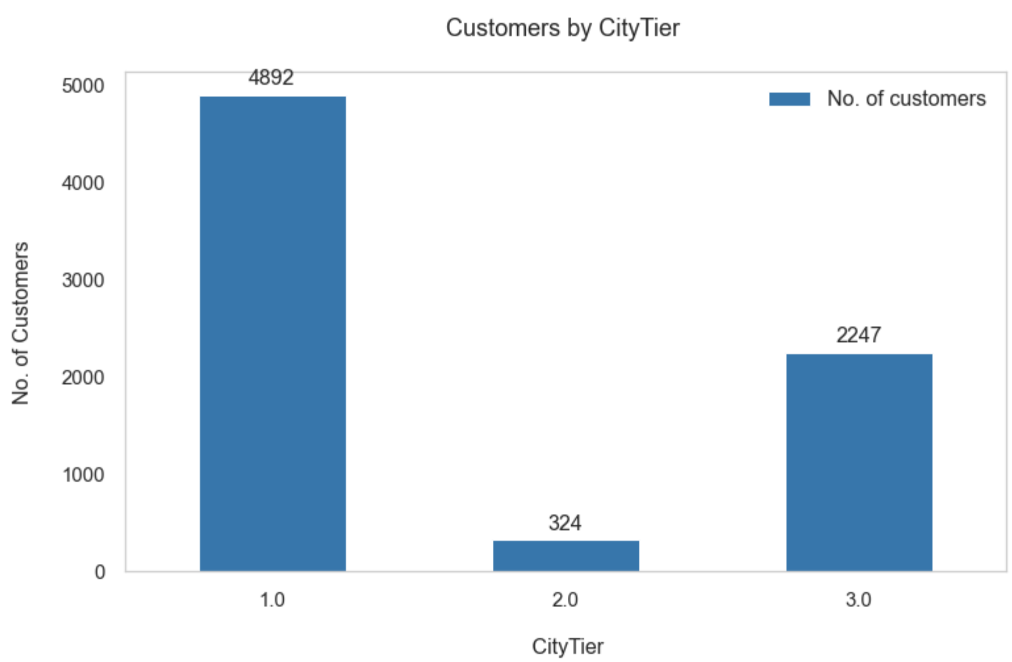
Inferences:

- As you can see, attrition within the first two years is more as compared to more tenured customers.
- **Short-term** churn is when customers churn after the initial few months. Short-term churn rates are typically high as customers test out different products and decide whether they add value or like them.

Recommendation:

- Reducing short-term churn comes down to finding the right fit between the customer and your product and proving value of the service to them quickly.
- When your short-term churn rate is extraordinarily high, examine your sales and marketing funnel to see if you are pitching the right products.

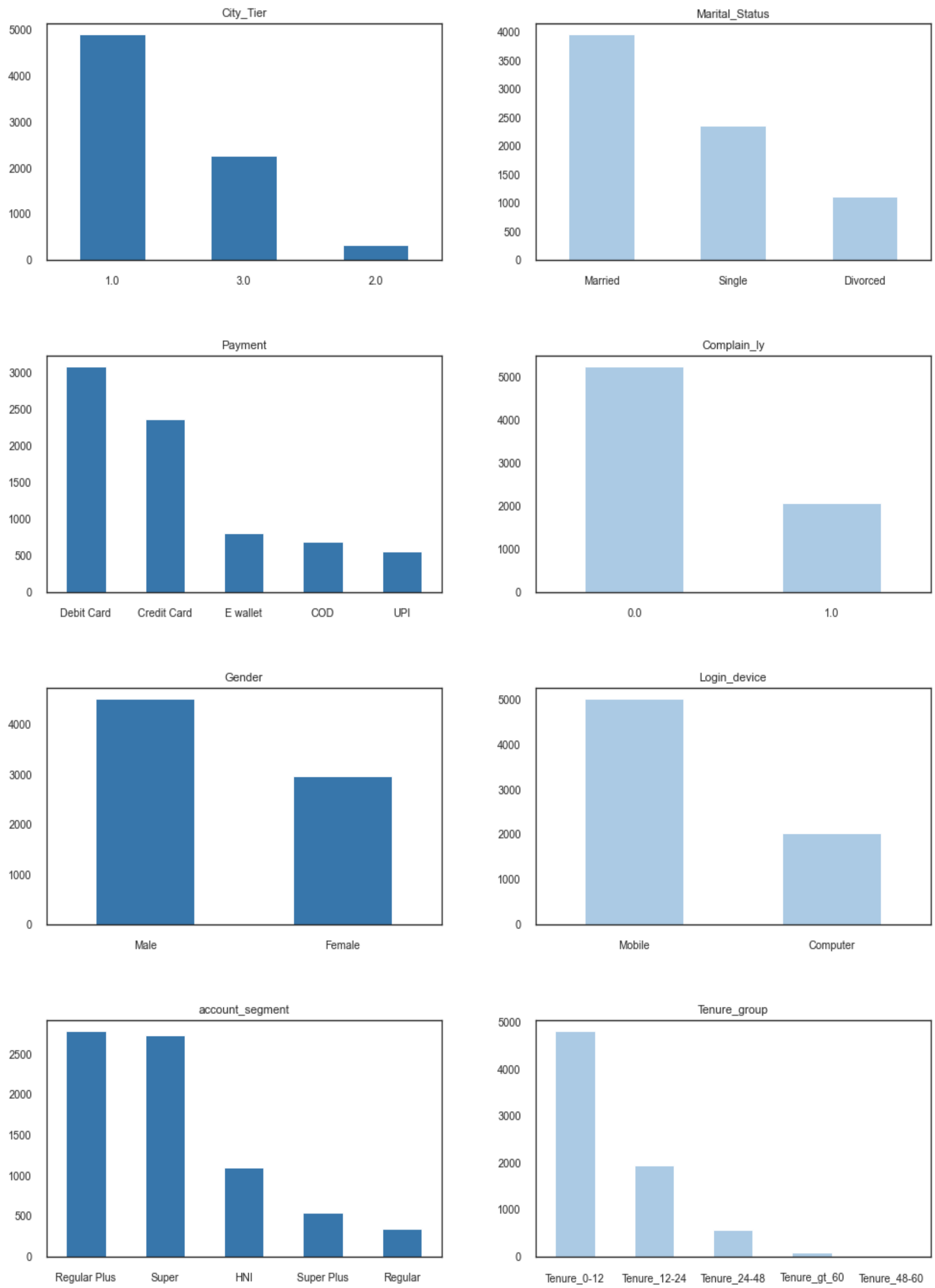
- **Distribution of CityTier:**



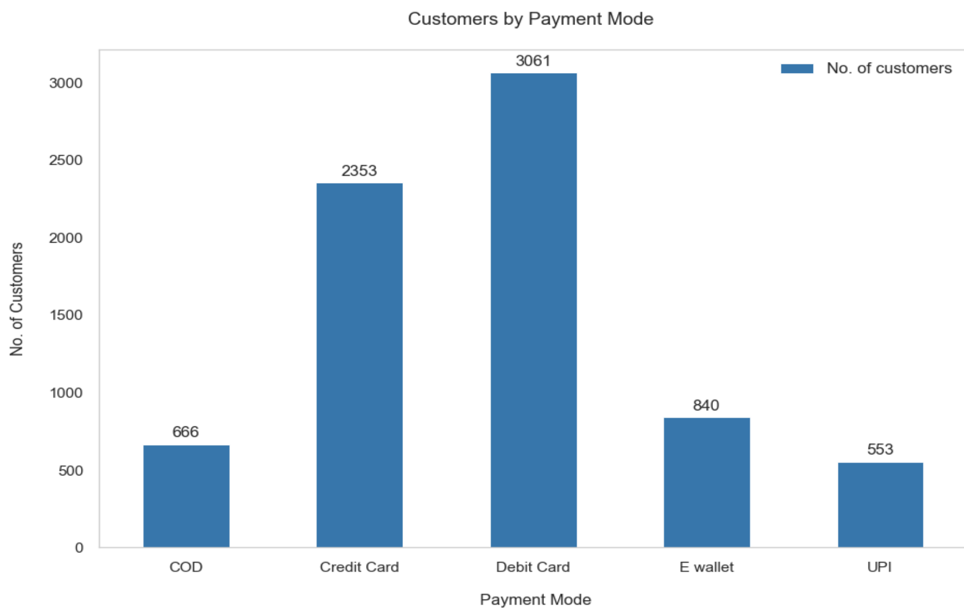
Inferences:

Most of the customers seem to be from City Tier 1. On the other hand, there are a smaller number of customers from City Tier 2.

- **Distribution of categorical variables:**



- **Distribution of payment method type:**



Inferences:

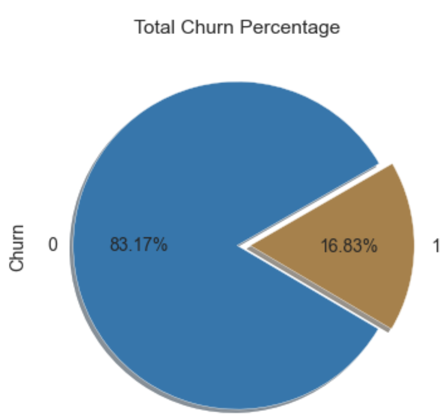
Most of the customers prefer debit card and credit card for payments. On the other hand, there are a smaller number of customers who pay through E wallet, COD and UPI.

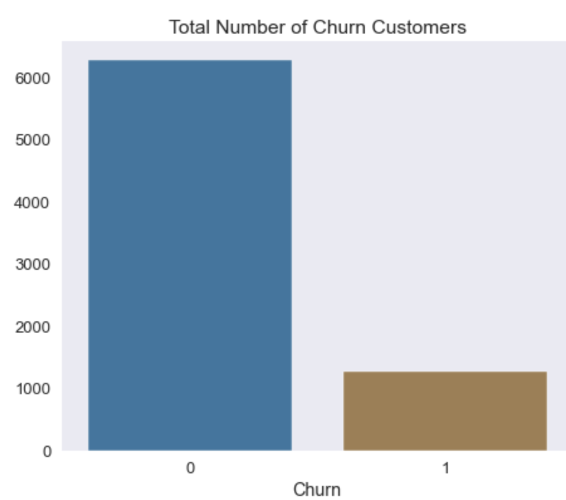
- **Check target variable distribution:**

Let us look at the distribution of churn values. This is quite a simple yet crucial step to see if the dataset upholds any class imbalance issues. As you can see below, the data set is imbalanced with a high proportion of active customers compared to their churned counterparts.

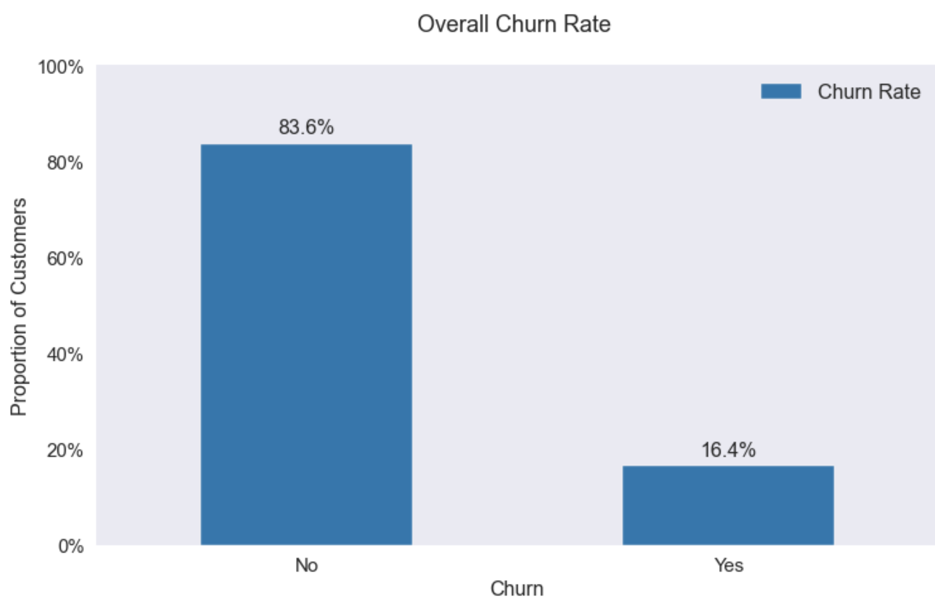
- **Proportion of observations in Target classes:**

```
0     6274
1     1270
```





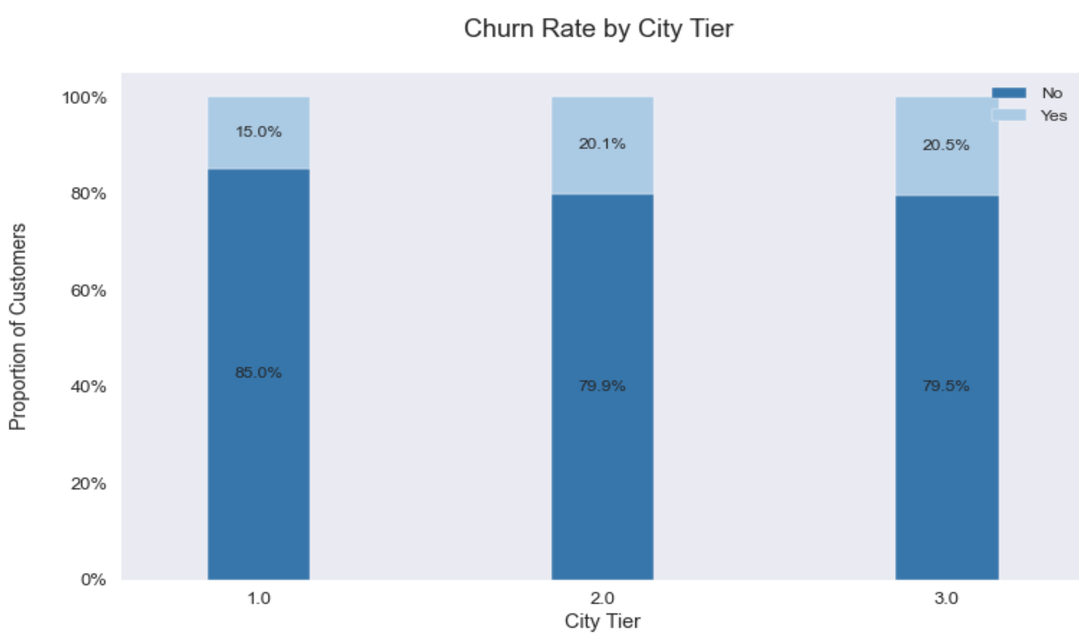
- **Overall Churn Rate:**



Inferences:

Overall churn rate: A preliminary look at the overall churn rate shows that around 83% of the customers are active. As shown in the chart below, this is an imbalanced classification problem. Machine learning algorithms work well when the number of instances of each class is roughly equal. Since the dataset is skewed, we need to keep that in mind while choosing the metrics for model selection.

3.2 Bivariate analysis (relationship between different variables, correlations)

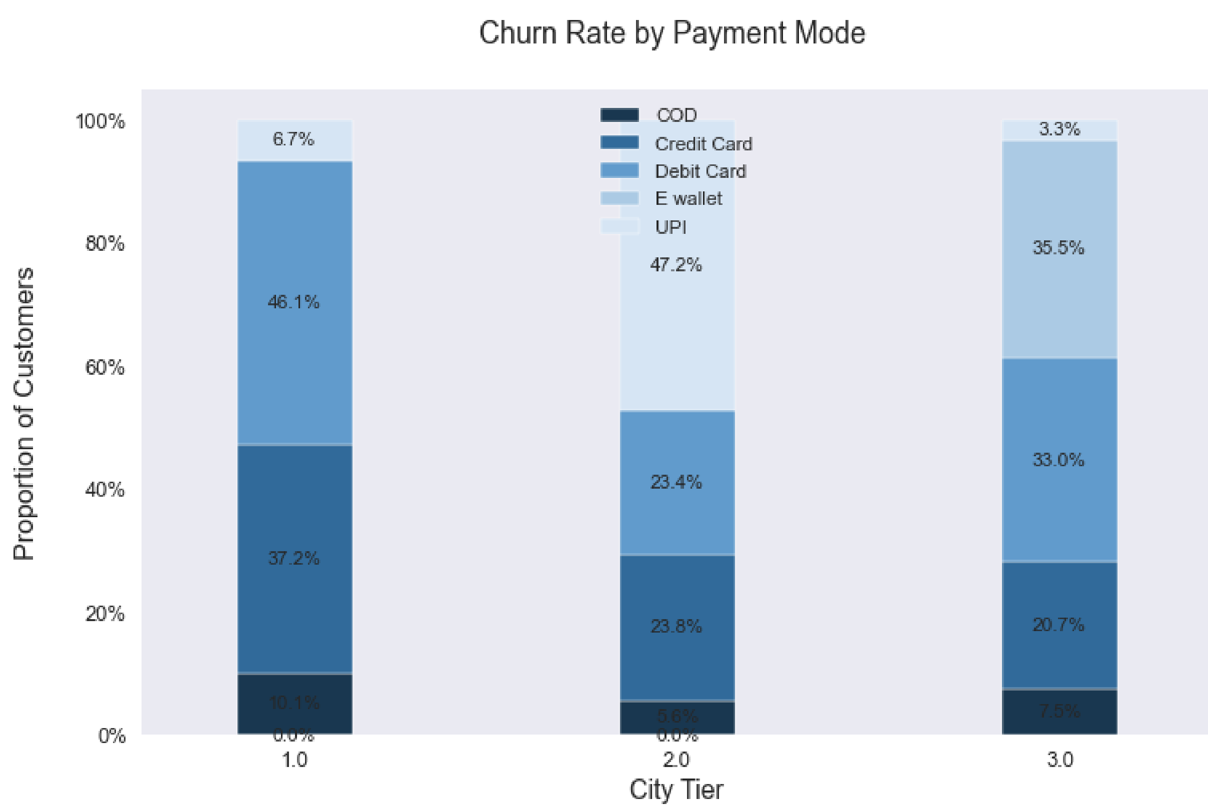


Inferences:

Proportion wise Customers of City Tier 3 or rather 2 have a remarkably high probability to churn compared to their peers on Tier 1.

Recommendation:

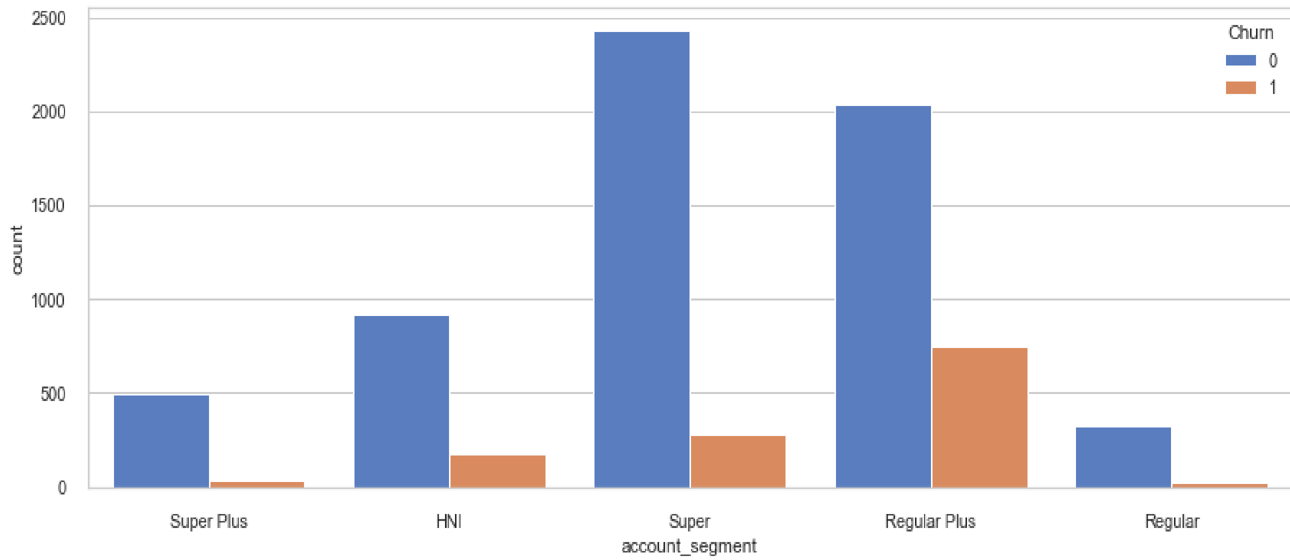
- Monitor the issues raised by the Tier 2 and Tier 3 customers.
- Surveys comprising of both close-ended and open-ended questions would help in understanding the factors leading to tier 2 and tier 3 attritions.

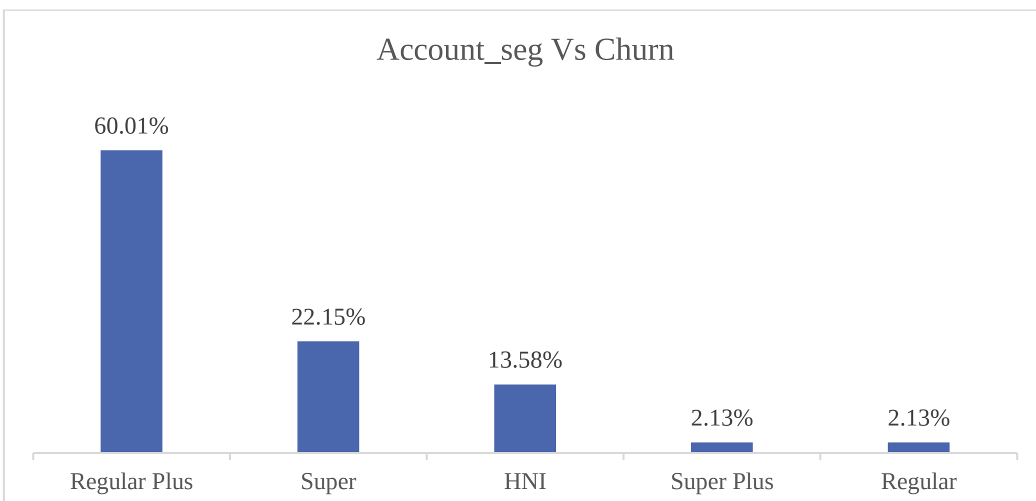


Inferences:

Customers who pay via Credit Card, Debit Card, or COD seem to have the lowest churn rate among all the payment method segments.

- **Account_segment vs Churn Relationship Analysis:**





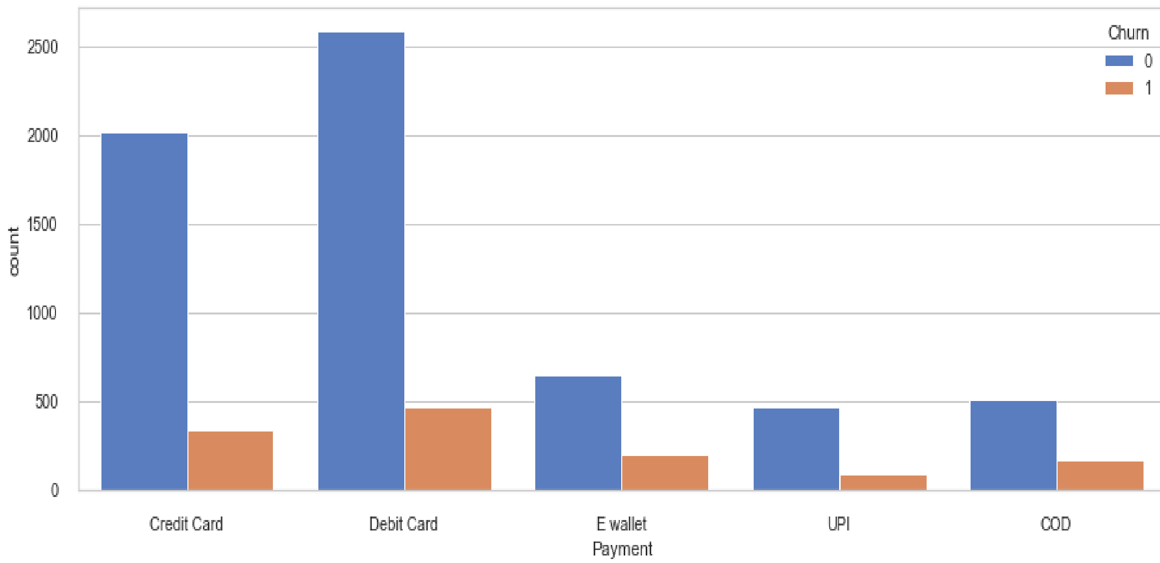
Inferences:

95.74% customer attrition is contributed by three account segments, namely Regular Plus, Super and HNI, individually contributing 60.01%, 22.15% and 13.58% respectively.

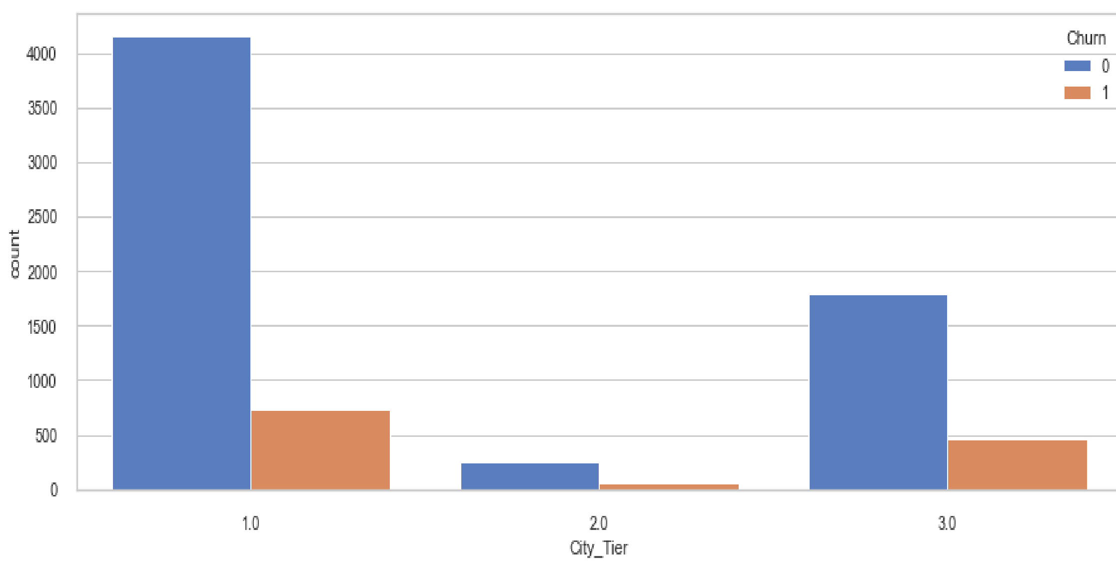
Recommendation:

Customer surveys can be conducted to identify the key issues with these segments and based on the results actions can be taken to address the issues.

- **Payment vs Churn Relationship Analysis**



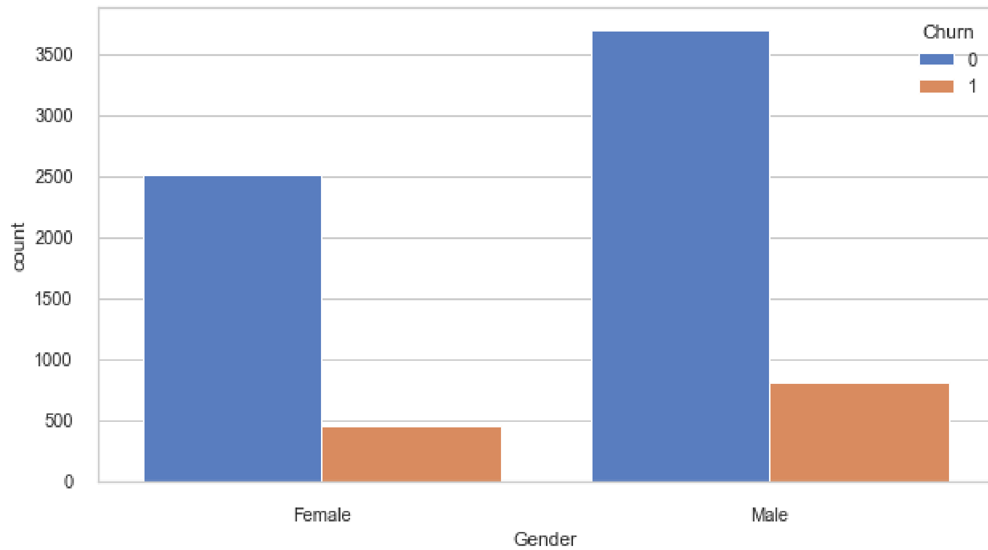
- **City Tier vs Churn Relationship Analysis**

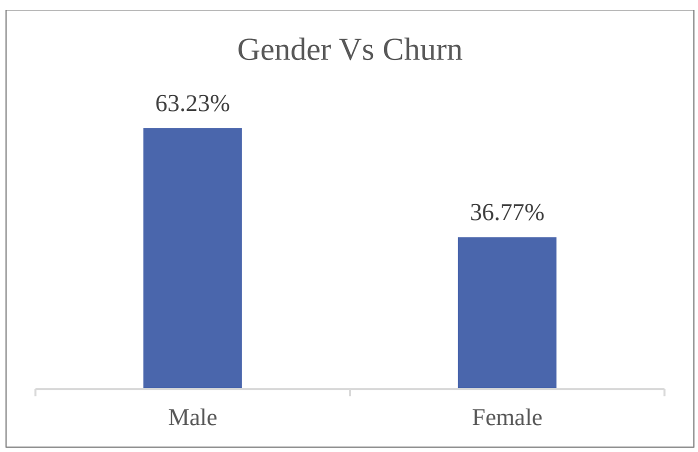


Inferences:

Customers who are in City 'Tier 1' and '3' seem to have the highest churn rate among all the other Tier.

- **Gender Vs Churn Relationship Analysis:**





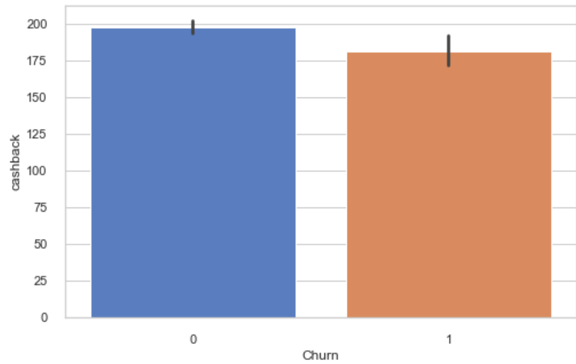
Inferences:

Male customers appear to be more dissatisfied as compared to female customers.

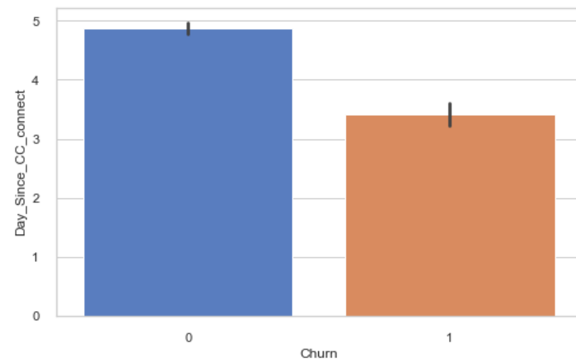
Recommendation:

Customers should be encouraged to share their preferences as part of personalizing their accounts based on which customized packages can be offered.

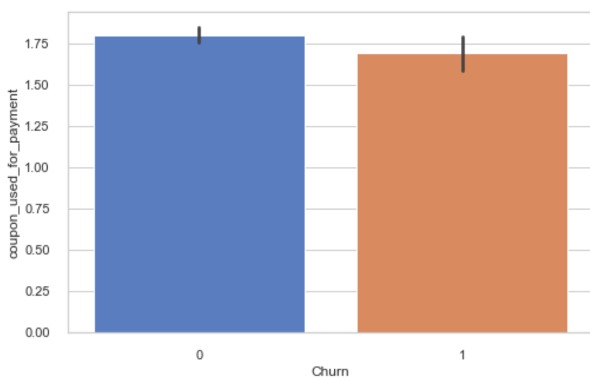
Churn Vs Cashback:



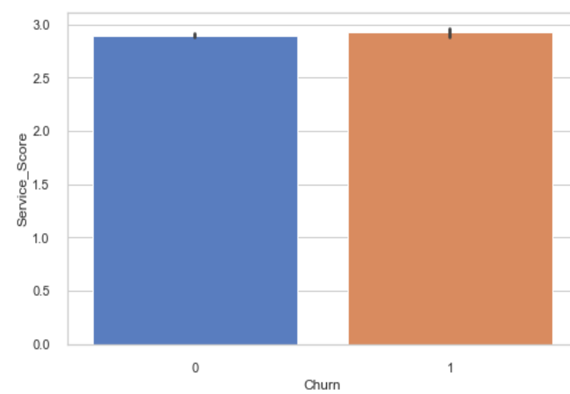
Churn Vs Day_Since_CC_connect:



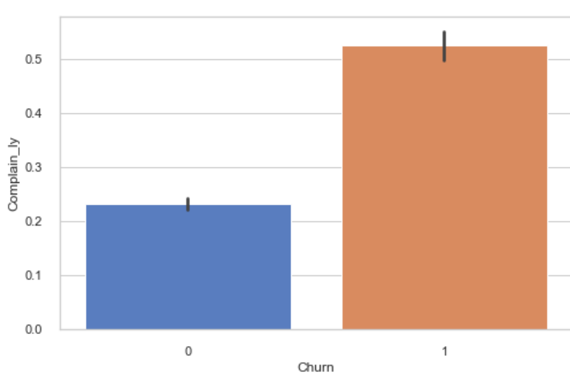
Churn Vs coupon_used_for_payment:



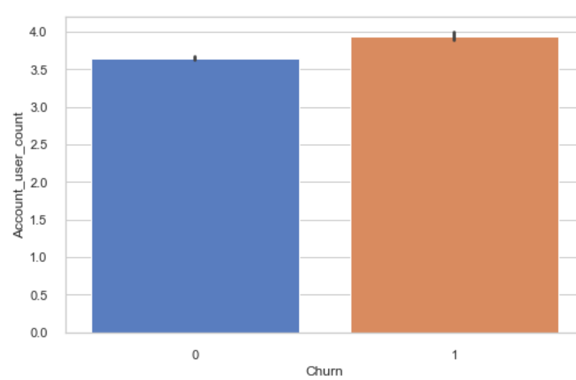
Churn Vs Service_Score:



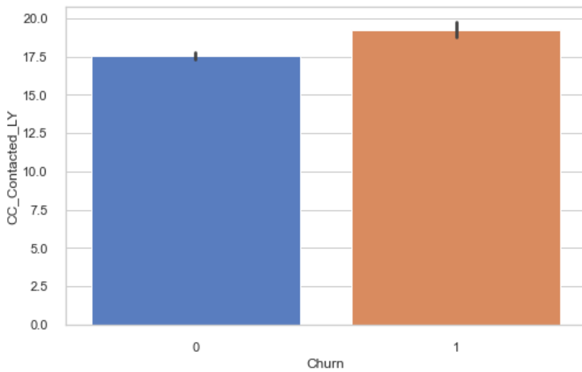
Churn Vs Complain_Ly:



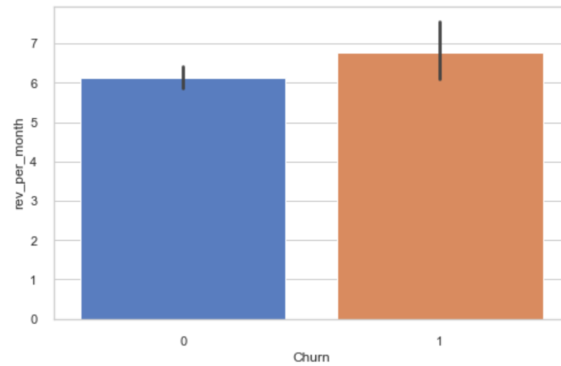
Churn Vs Account_user_count:



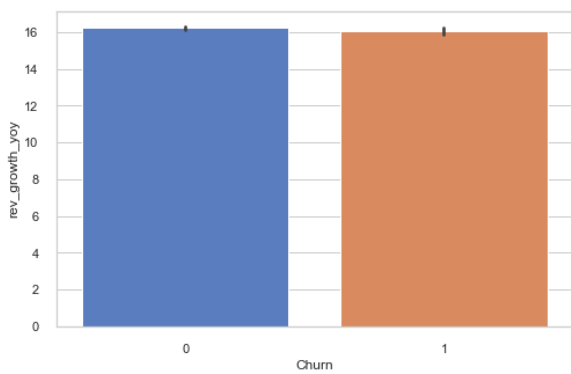
Churn Vs CC_Contacted_Ly:



Churn Vs rev_per_month:



Churn Vs rev_growth_yoy:



Inferences:

Churned Customer Profile:

For better understanding we wanted to compare these important variables from costumers who churned and costumers who did not churned.

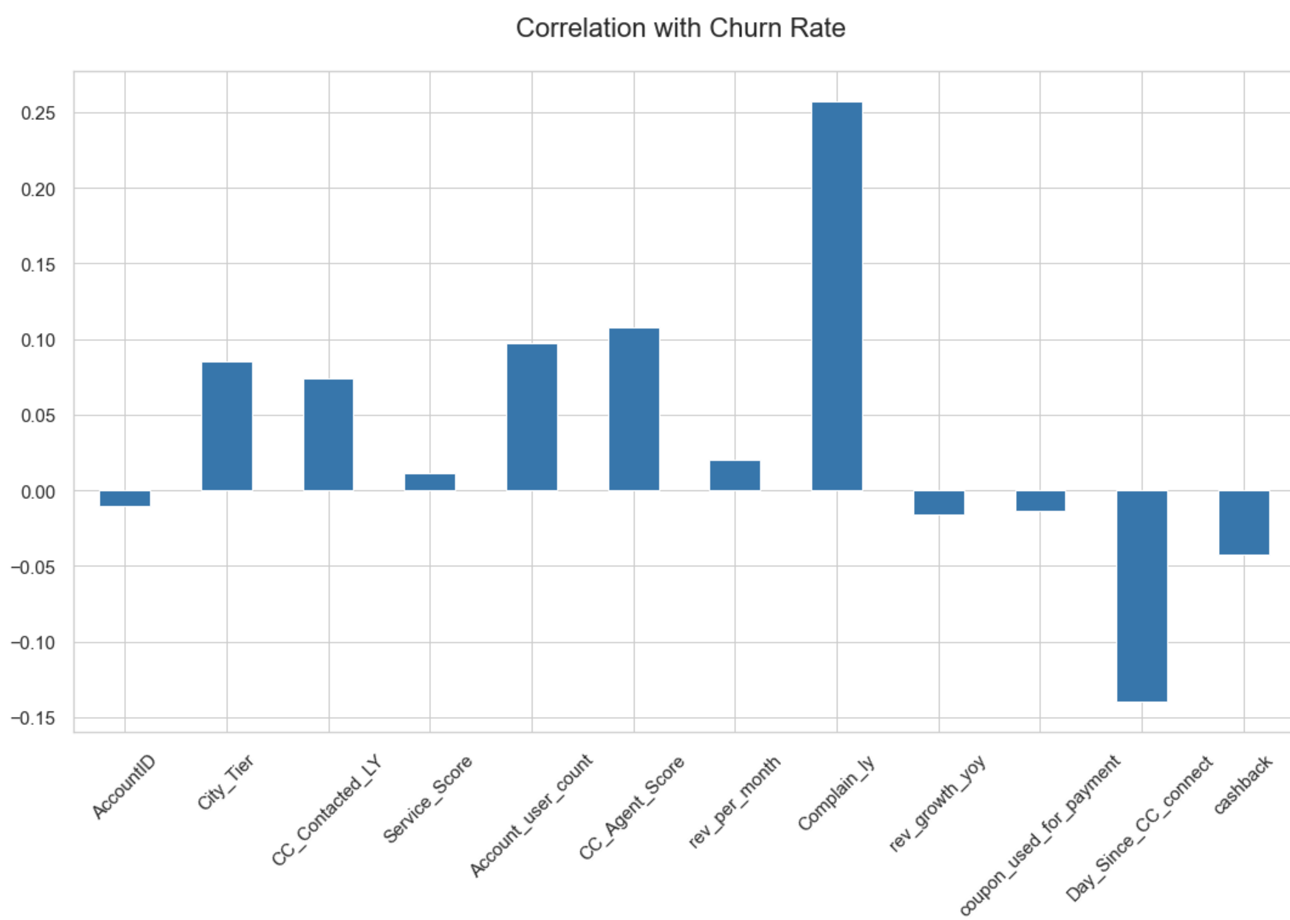
Customer that churned are those who -

- Tend to generate less cashback amounts.
- Contacted the customer care very less recently.
- Have a greater number of complains.
- Have more account users.

3.3 Multi-variate analysis:

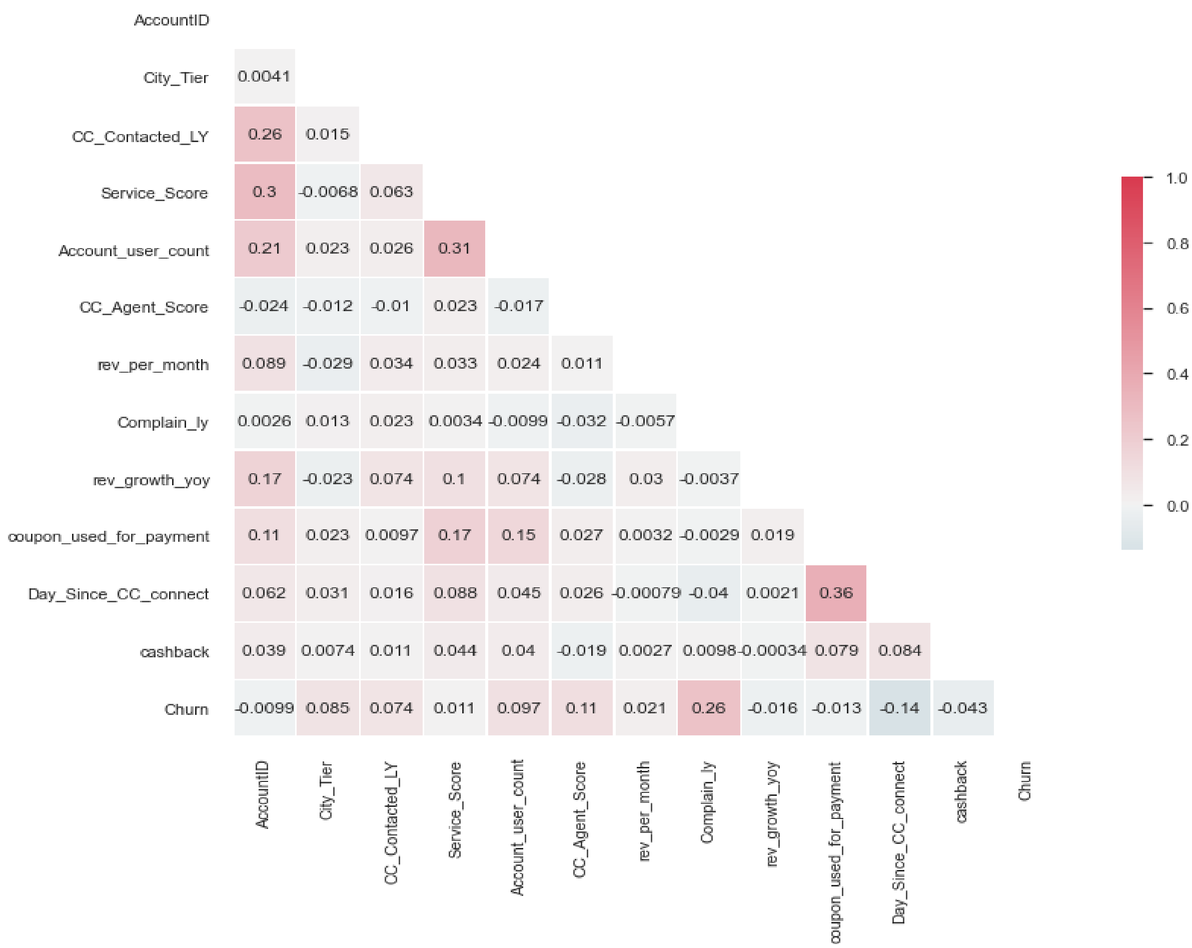
```
Most Positive Correlations:
   1. Complain_ly              0.257065
   2. CC_Agent_Score           0.107562
   3. Account_user_count       0.097248
   4. CC_Contacted_LY          0.073907
   5. City_Tier                0.085430
   6. rev_per_month            0.020572
   7. Service_Score            0.011384


Most Negative Correlations:
   1. rev_growth_yoy                 -0.016252
   2. AccountID                      -0.009916
   3. coupon_used_for_payment        -0.013266
   4. cashback                       -0.042804
   5. Day_Since_CC_connect           -0.139870
```

- **Plot positive & negative correlations:**



- **Plot Correlation Matrix of all independent variables:**

Correlation matrix helps us to discover the bivariate relationship between independent variables in a dataset.



4. Data Cleaning and Pre-processing

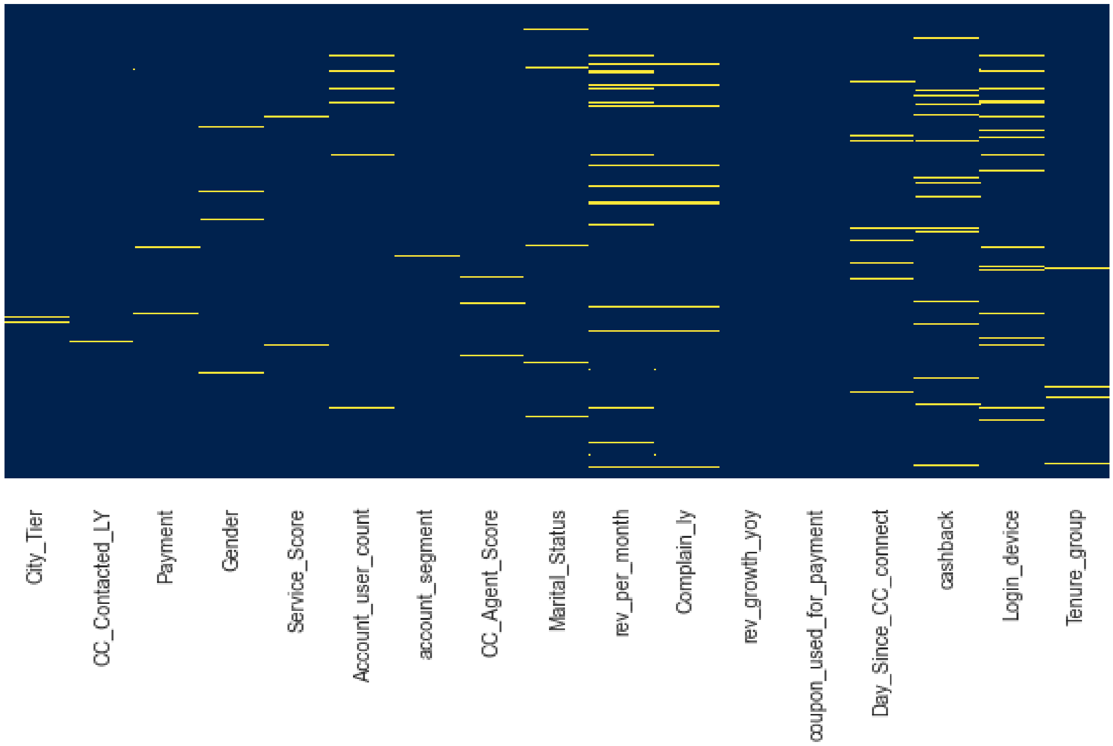
4.1 Removal of unwanted variables

- **Dropping irrelevant data**

There may be data included that is not needed to improve our results. Best is that to identify by logic thinking or by creating a correlation matrix. In this data set we have the AccountID for example. As it does not influence our predicted outcome, we drop the column with the pandas “drop()” function.

After removal of unwanted variables, we have 7544 records and 18 attributes in train dataset and 3716 records and 18 attributes in test dataset.

4.2 Missing Value treatment



Percentage of values that are null in Train:

	Total	Percent
rev_per_month	522	6.919406
Login_device	506	6.707317
cashback	312	4.135737
Account_user_count	290	3.844115
Complain_ly	239	3.168081
Day_Since_CC_connect	237	3.141569
Tenure_group	144	1.908802
Marital_Status	138	1.829268
CC_Agent_Score	82	1.086957
Gender	72	0.954401
account_segment	71	0.941145
Payment	71	0.941145
Service_Score	63	0.835101
City_Tier	63	0.835101
CC_Contacted_LY	61	0.808590
coupon_used_for_payment	3	0.039767
rev_growth_yoy	2	0.026511

Percentage of values that are null in Test:

	Total	Percent
rev_per_month	269	7.238967
Login_device	254	6.835307
cashback	161	4.332616
Account_user_count	154	4.144241
Day_Since_CC_connect	121	3.256189
Complain_ly	118	3.175457
Tenure_group	74	1.991389
Marital_Status	74	1.991389
City_Tier	49	1.318622
CC_Contacted_LY	41	1.103337
Payment	38	1.022605
Gender	36	0.968784
Service_Score	35	0.941873
CC_Agent_Score	34	0.914962
account_segment	26	0.699677
rev_growth_yoy	1	0.026911
coupon_used_for_payment	0	0.000000

Inferences:

- There are some missing values.
- Missing values are common occurrences in data. Unfortunately, most predictive modelling techniques cannot handle any missing values. Therefore, this problem must be addressed prior to modelling.
- Let us treat these categorical variables missing values with mode.

- **KNNImputer: A robust way to impute missing values.**

k-Nearest Neighbours (kNN) that identifies the neighboring points through a measure of distance and the missing values can be estimated using completed values of neighboring observations. A new sample is imputed by finding the samples in the training set "closest" to it and averages these nearby points to fill in the value.

- **Dataset after imputation of missing values:**

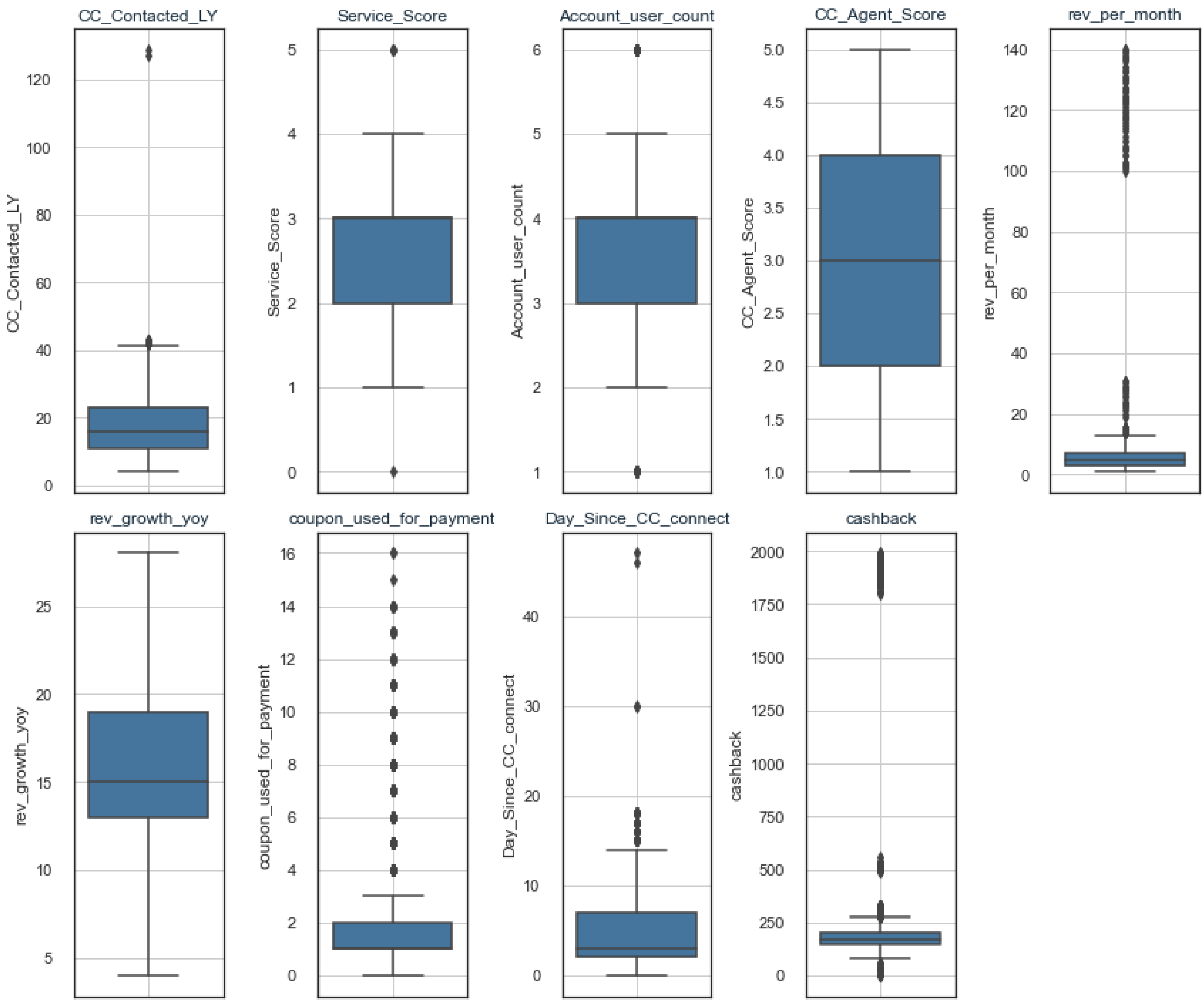
	City_Tier	CC_Contacted_LY	Payment	Gender	Service_Score	Account_user_count	account_segment	CC_Agent_Score	Marital_Status	rev_per_month	Co
0	3.0	17.0	3.0	0.0	3.0	4.0	3.0	1.0	1.0	5.0	
1	1.0	36.0	2.0	0.0	3.0	3.0	2.0	1.0	1.0	7.0	
2	1.0	11.0	1.0	0.0	4.0	4.0	2.0	3.0	1.0	6.0	
3	1.0	10.0	1.0	0.0	2.0	3.0	2.0	3.0	1.0	8.0	
4	1.0	25.0	4.0	1.0	3.0	3.2	2.0	5.0	1.0	3.2	

- **Checking for missing values after missing value treatment:**

	Total	Percent
Tenure_group	0	0.0
CC_Agent_Score	0	0.0
CC_Contacted_LY	0	0.0
Payment	0	0.0
Gender	0	0.0
Service_Score	0	0.0
Account_user_count	0	0.0
account_segment	0	0.0
Marital_Status	0	0.0
Login_device	0	0.0
rev_per_month	0	0.0
Complain_ly	0	0.0
rev_growth_yoy	0	0.0
coupon_used_for_payment	0	0.0
Day_Since_CC_connect	0	0.0
cashback	0	0.0
City_Tier	0	0.0

4.3 Outlier treatment

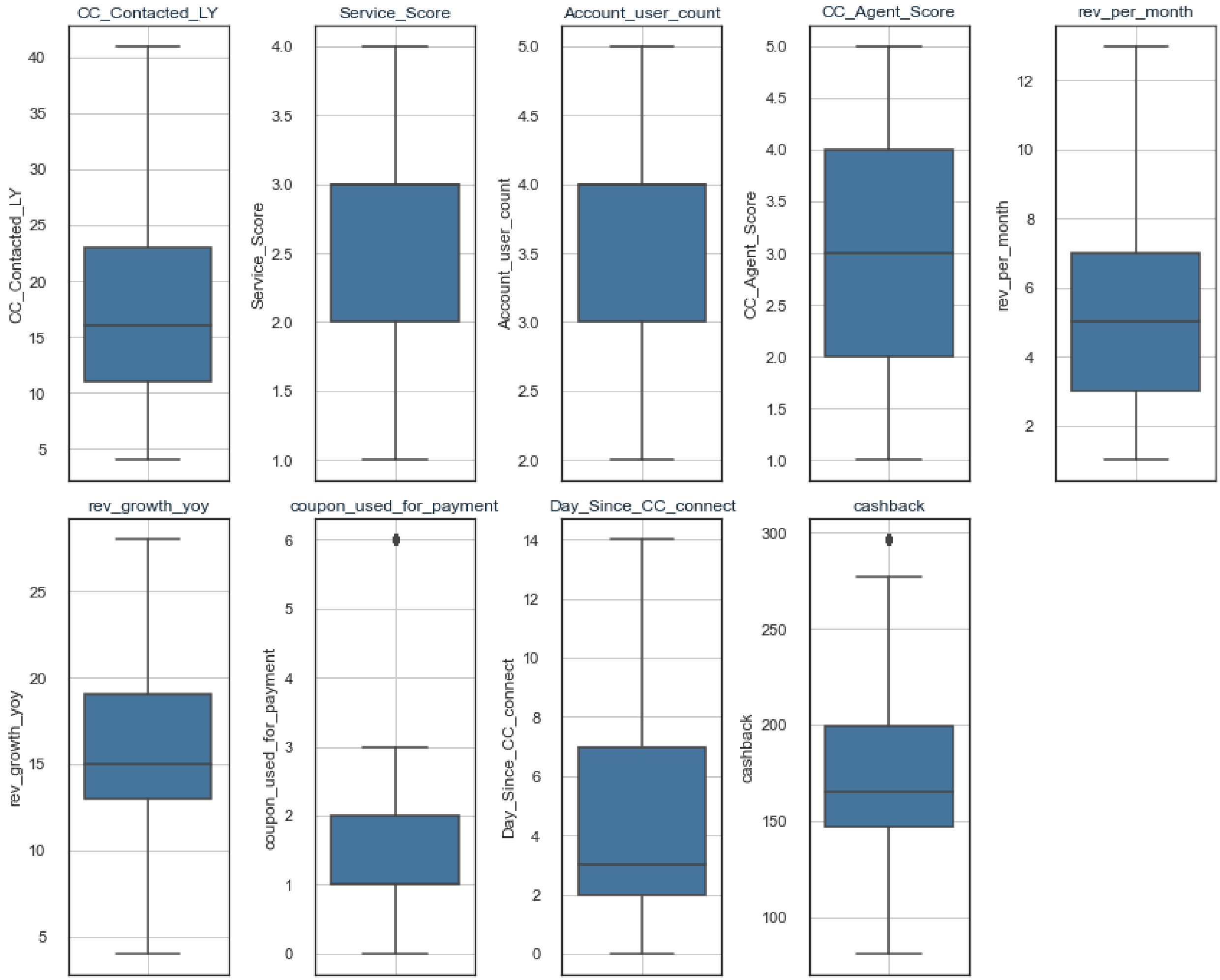
- **Checking for outliers**



Inferences:

An outlier is a record which significantly differs from typical records. This means that it has at least one feature with an atypical value. Outliers could be related to noisy data, but in some cases, they are special records, which diverge from normality. This suggests that the outlier concept assumes different faces depending on the problem. The outlier detection phase concerns the search of anomalous records, which must be removed if they represent noise.

- **After treating the outliers**



4.4 Variable transformation

- **Converting Object data type into Categorical**

Payment	
COD	0
Credit Card	1
Debit Card	2
E wallet	3
UPI	4
account_segment	
HNI	0
Regular	1
Regular Plus	2
Super	3
Super Plus	4

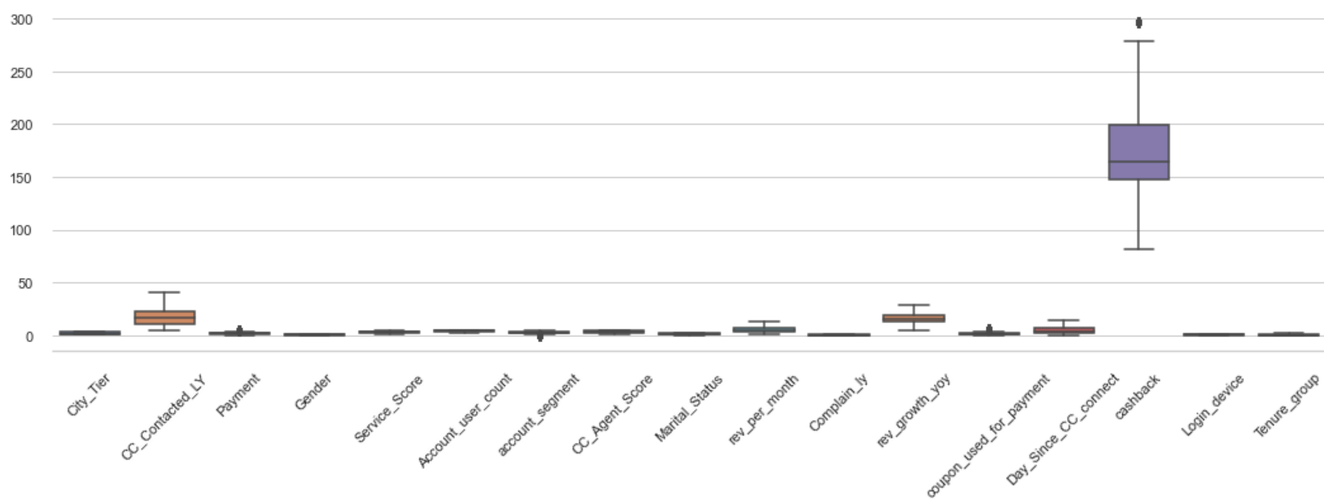
Login_device	
Computer	0
Mobile Phone	1
Gender	
Female	0
Male	1
MaritalStatus	
Divorced	0
Married	1
Single	2

- **Feature Scaling - Standardization:**

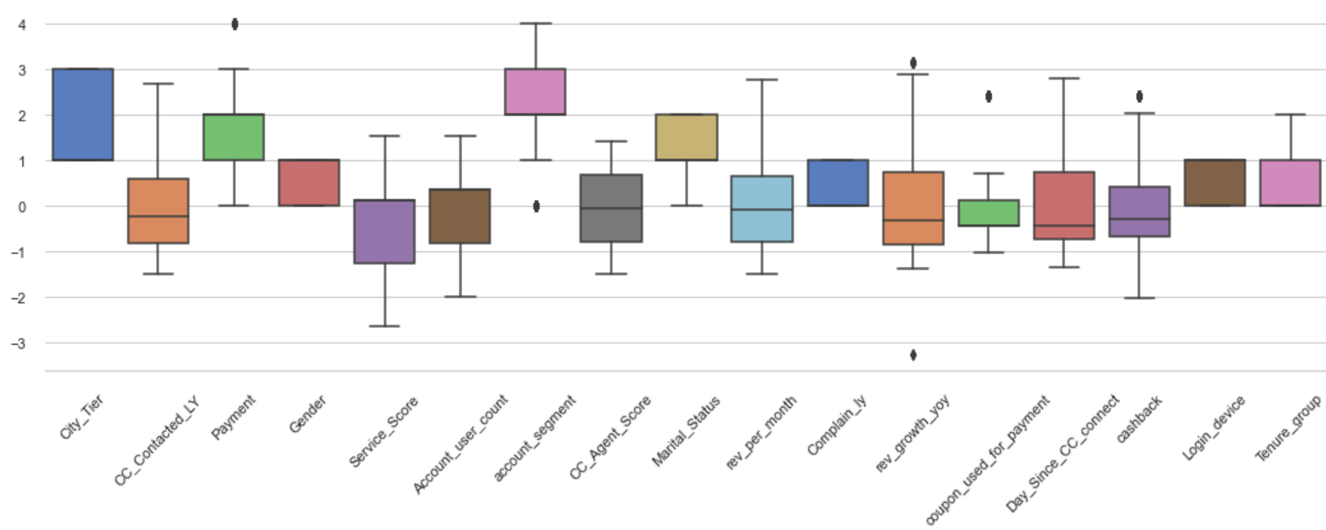
The most common techniques of feature scaling are Normalization and Standardization. Normalization is used when we want to bound our values between two numbers, typically, be tween [0,1] or [-1,1]. While **Standardization** transforms the data to have zero mean and a var iance of 1, they make our data unitless.

Since cashback variable is in 100s and the rest are in the range of 10s we need to scale the data.

Before scaling:



After scaling:



4.5 Addition of new variables

Tenure_group	
Tenure_0-12	0
Tenure_12-24	1
Tenure_24-48	2
Tenure_48-60	3
Tenure_gt_60	4

5. Business insights from EDA

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

The given dataset is unbalanced. Class imbalance is a common problem in data mining. The class imbalance problem occurs if the dataset used for the analysis is unbalanced, which means that the number of negative records is much higher than the number of positive records (churn). This disproportion leads classifiers to ignore the rare class, that is, classifying all the records as negative (not churn), because this would imply a high accuracy. This problem is much more relevant if the rare class is more important than the negative one. In fact, the negative class (not churn) has usually poor interest in being predicted, while the rare class (churn) often represents a significant event. This means that the cost of misclassifying a positive record is higher than the cost of other errors. Moreover, the positive class is more prone overfitting given its scarcity. All of this makes learning from imbalanced datasets challenging. They can be summarized in three important groups:

- Under sampling
- Oversampling
- Synthetic Minority Over-sampling Technique (SMOTE)

SMOTE is a more sophisticated technique than oversampling and under sampling. SMOTE synthesizes artificial positive records with a linear combination of some real positive ones. Moreover, SMOTE also applies under sampling to the negative class to balance the proportion without generating too many artificial records. In practice, SMOTE is immensely powerful and decreases the chance that a model overfits the rare class.

5.2 Any business insights using clustering

Clustering algorithms are used for customer churn analysis; one of the important reasons is that the cost of increasing a new customer is much higher than retaining an existing customer by using customer churn analysis.

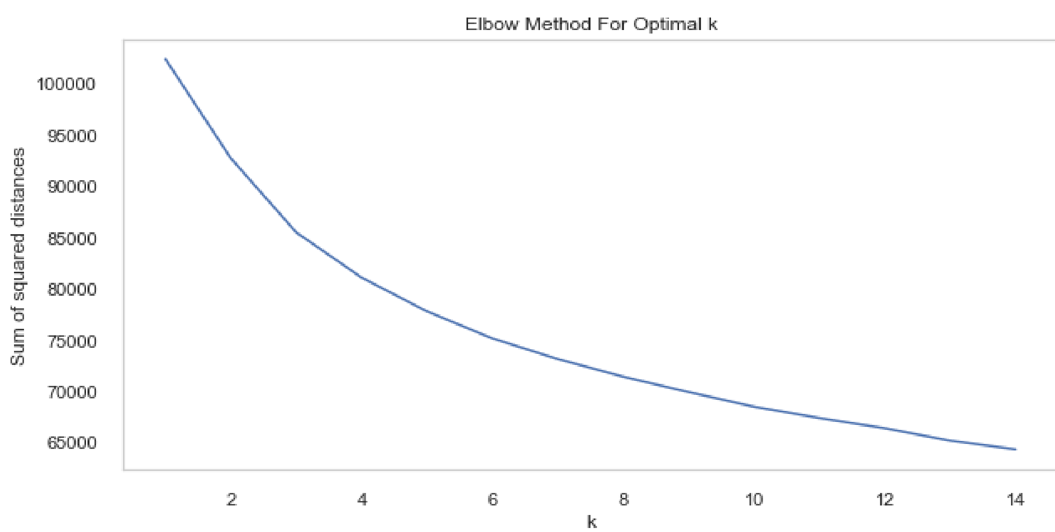
The K-means clustering algorithm is used to find groups which have not been explicitly labeled in the data. This can be used to confirm business assumptions and offer promos or coupons to retain them.

Applying K-Means clustering on scaled data and determining optimum clusters.

Within Cluster Sum of Squares (WSS)

```
[102448.10732237533,
92761.73457896683,
85530.8144139462,
81116.55509470268,
77816.08231676776,
75181.03038723084,
73165.40339226676,
71441.07717330004,
69944.76484258476,
68511.56356228488,
67405.39061547787,
66414.35081250443,
65203.306481860214,
64349.64439513568]
```

Applying elbow curve and silhouette score.

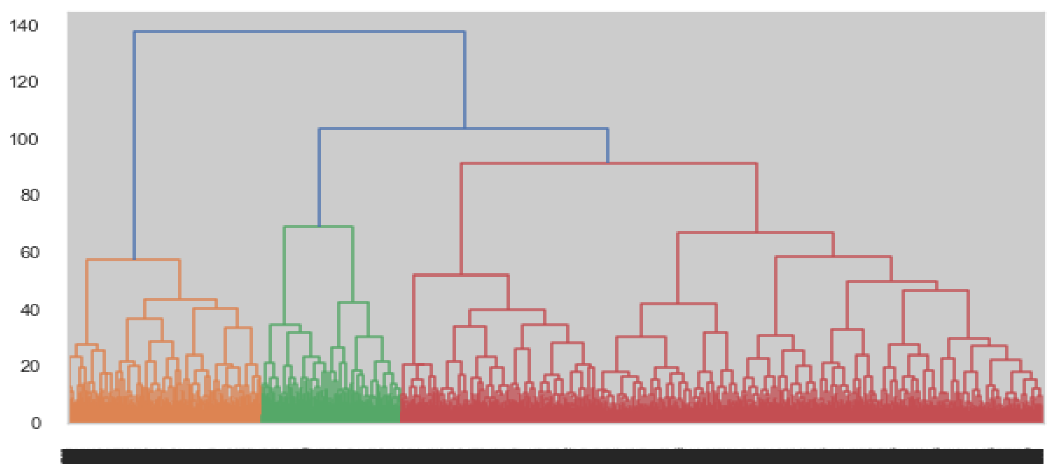


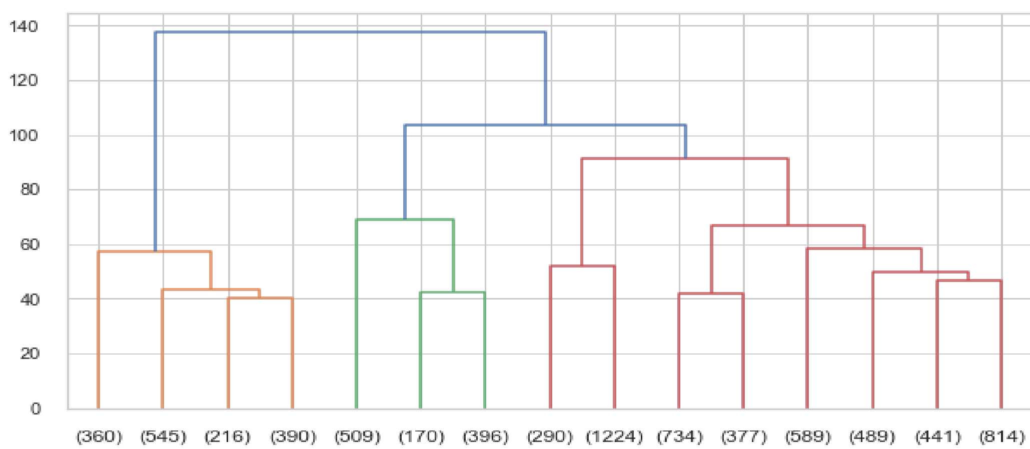
We can observe that the “elbow” is the number 3 which is optimal for this case. Now we can run a K-Means using as n_clusters the number 3

Inference:

- WSS reduces as K keeps increasing.
- Silhouette score: The average of sil-width for each observation of a dataset is called as silhouette score.
- Silhouette score for 3 Cluster is 0.09 which is closer to +1 than for 5 and 6 cluster (0.080 & 0.083) respectively.
- But selection 2 clusters do not give us any insights so we can say that the 3 Clusters are well separated from each other on an average.
- From 1 and 2 cluster shown in the (WSS) plot, there is a significant drop. Similarly, there is a significant drop between 2 and 3. Hence, 3 is a valuable addition in K-means algorithm.

Hierarchical clustering:





1	1511
2	1075
3	4958

H_clusters	City_Tier	Payment	Gender	account_segment	Marital_Status	Complain_ly	Login_device	Tenure_group	Freq	CC_Contacted_LY	Service_Score
1	1.0	2.0	1.0	0.0	1.0	0.0	1.0	0.0	1511	-0.023501	0.021897
2	1.0	2.0	1.0	4.0	1.0	0.0	1.0	0.0	1075	0.151915	-0.056197
3	1.0	2.0	1.0	2.0	1.0	0.0	1.0	0.0	4958	-0.025776	0.005511

Inference:

- Cluster 1: Customers that have HNI account segment and are in low tenure group are those who spend more for short period.
- Cluster 2: Customers that have Super Plus account segment and are in low tenure group are those who spend little more than medium for less period.
- Cluster 3: Customers that have Regular Plus account segment and are in low tenure group are those who want to try the scheme.

5.3 Other insights

- The dataset had missing values.
- Strongest positive correlation with the target features is ‘Complain_LY’, ‘Account_user_count’ and ‘CC_Agent_Score’ whilst negative correlation is with ‘Tenure’, ‘Coupons_used_for_payment’ and ‘cashback’.
- The dataset is imbalanced with many customers being active.
- Most of the customers in the dataset are Male and Married people.
- There are a lot of new customers in the organization (less than 10 months old) followed by a loyal customer base that’s above 12 months old.
- Most of the customers seem to have Regular Plus and Super as account segment.

6. Model building

True Positive (TP):

- The actual value was positive, and the model predicted a positive value.

True Negative (TN):

- The actual value was negative, and the model predicted a negative value.

False Positive (FP):

- Type 1 error: The actual value was negative, but the model predicted a positive value.

False Negative (FN):

- Type 2 error: The actual value was positive, but the model predicted a negative value.

Precision: TP/ (TP + FP)

- This metric evaluates how precise a model is in predicting positive labels. It answers the question, out of the number of times a model predicted positive, how often was it correct?
- When a positive value is predicted, how often is the prediction correct?

Recall: TP/ (TP + FN)

- Often called sensitivity, the recall calculates the percentage of actual positives a model correctly identified (True Positive).
- When the actual value is positive, how often is the prediction correct?

F1-Score: (2 x Precision x Recall) / (Precision + Recall)

- F1-Score is the weighted average of Precision and Recall used in all types of classification algorithms. Therefore, this score takes both false positives and false negatives into account. F1-Score is usually more useful than accuracy, especially if you have an uneven class distribution.

Accuracy: TP+TN/ (TP + TN + FP + FN)

- Accuracy is an evaluation metric that allows you to measure the total number of predictions a model gets right.

Model-1: Logistic Regression

Definition:

Logistic regression is a machine learning algorithm for classification. In this algorithm, the probabilities describing the possible outcomes of a single trial are modelled using a logistic function.

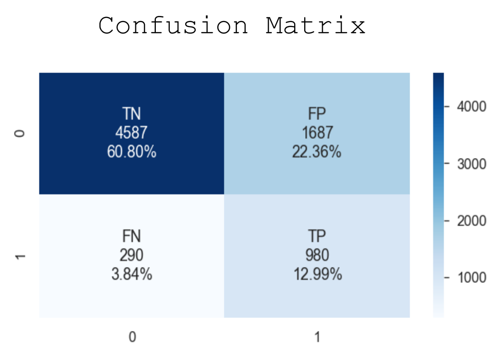
Advantages:

- Logistic regression is designed for this purpose (classification) and is most useful for understanding the influence of several independent variables on a single outcome variable.

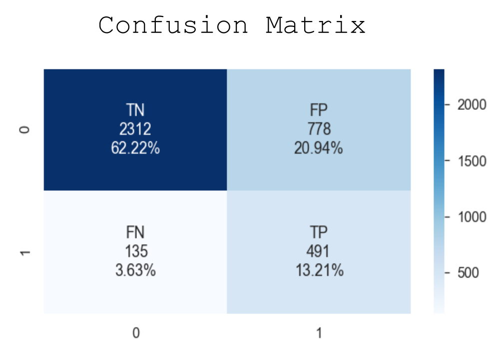
Disadvantages:

- Works only when the predicted variable is binary, assumes all predictors are independent of each other and assumes data is free of missing values.

```
LogisticRegression(C=71968.56730011529, class_weight='balanced')

Classification Report for Train dataset
=====================================================

              precision    recall  f1-score   support
           0     0.94      0.73      0.82      6274
           1     0.37      0.77      0.50      1270
    accuracy                         0.74      7544
   macro avg     0.65      0.75      0.66      7544
weighted avg     0.84      0.74      0.77      7544

AUC-ROC = 0.8248150723269888
=====================================================
```




```
Classification Report for Test dataset
===================================================

             precision     recall  f1-score   support
          0     0.94       0.75      0.84      3090
          1     0.39       0.78      0.52       626
   accuracy                          0.75      3716
  macro avg     0.67       0.77      0.68      3716
weighted avg    0.85       0.75      0.78      3716

AUC-ROC = 0.8400560397861802
===================================================
Accuracy_Train 0.7379374337221634
Accuracy_Test 0.7543057050592035
```



Inference:

From the above matrix since there is imbalance in dataset the model performs poor on the minority class.

Model-2: Naïve Bayes

Definition:

Naive Bayes algorithm based on Bayes' theorem with the assumption of independence between every pair of features. Naive Bayes classifiers work well in many real-world situations such as document classification and spam filtering.

Advantages:

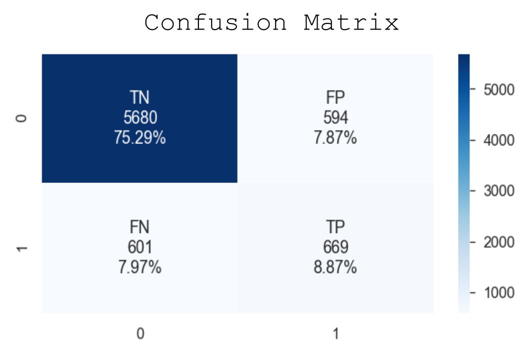
- This algorithm requires a small amount of training data to estimate the necessary parameters. Naive Bayes classifiers are extremely fast compared to more sophisticated methods.

Disadvantages:

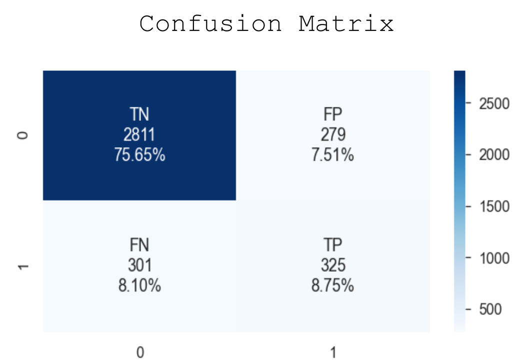
- Naive Bayes is known to be a bad estimator.

```
GaussianNB()
```

```
Classification Report for Train dataset
==================================================
            precision    recall  f1-score   support
           0     0.90      0.91      0.90      6274
           1     0.53      0.53      0.53      1270
    accuracy                         0.84      7544
   macro avg     0.72      0.72      0.72      7544
weighted avg     0.84      0.84      0.84      7544

AUC-ROC = 0.7973422373048125
===================================================
```




```
Classification Report for Test dataset
=================================================
            precision   recall  f1-score   support
           0    0.90      0.91      0.91      3090
           1    0.54      0.52      0.53       626
    accuracy                        0.84      3716
   macro avg    0.72      0.71      0.72      3716
weighted avg    0.84      0.84      0.84      3716

AUC-ROC = 0.8075157418034058
=================================================
Accuracy_Train 0.8415959703075292
Accuracy_Test 0.8439181916038752
```



Inference:

Bad estimator.

Model-3: Stochastic Gradient Descent

Definition:

Stochastic gradient descent is a simple and very efficient approach to fit linear models. It is particularly useful when the number of samples is large. It supports different loss functions and penalties for classification.

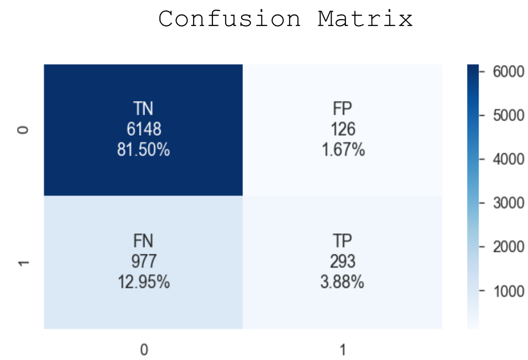
Advantages:

- Efficiency and ease of implementation.

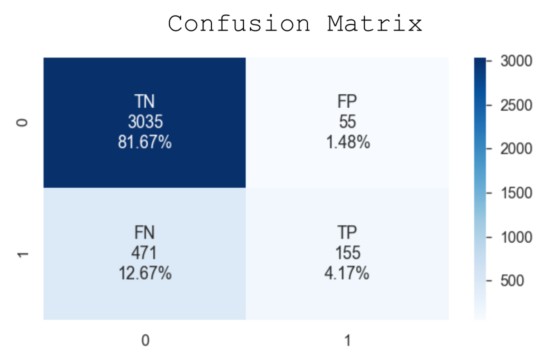
Disadvantages:

- Requires several hyper-parameters and it is sensitive to feature scaling.

```
SGDClassifier(loss='log', max_iter=1500, random_state=123)

Classification Report for Train dataset
==================================================
            precision    recall  f1-score   support

           0     0.86      0.98      0.92      6274
           1     0.70      0.23      0.35      1270
    accuracy                         0.85      7544
   macro avg     0.78      0.61      0.63      7544
weighted avg     0.84      0.85      0.82      7544

AUC-ROC = 0.81123798001501
==================================================
```




```
Classification Report for Test dataset
==================================================
            precision   recall  f1-score   support

           0     0.87     0.98      0.92      3090
           1     0.74     0.25      0.37       626
    accuracy                        0.86      3716
   macro avg     0.80     0.61      0.65      3716
weighted avg     0.84     0.86      0.83      3716

AUC-ROC = 0. 8280710733376759
=================================================
```




```
Accuracy_Train 0.8537910922587487
Accuracy_Test 0.8584499461786868
```

Model-4: K-Nearest Neighbours

Definition:

Neighbours based classification is a type of lazy learning as it does not attempt to construct a general internal model, but simply stores instances of the training data. Classification is computed from a simple majority vote of the k nearest neighbours of each point.

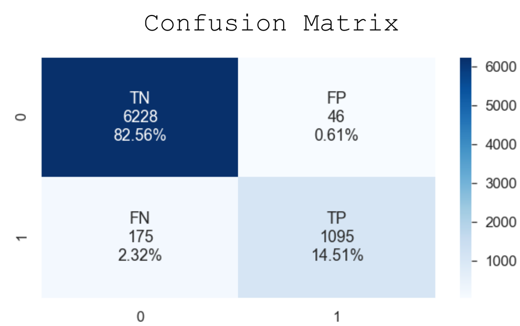
Advantages:

- This algorithm is simple to implement, robust to noisy training data, and effective if training data is large.

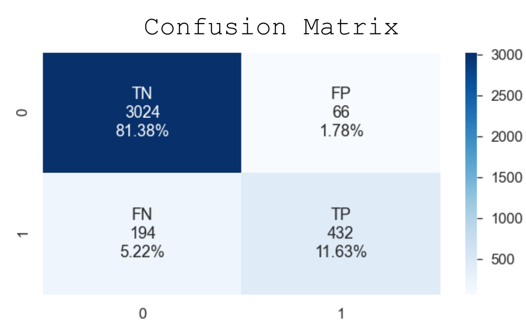
Disadvantages:

- Need to determine the value of K and the computation cost is high as it needs to compute the distance of each instance to all the training samples.

```
KNeighborsClassifier()

Classification Report for Train dataset
=====================================================
              precision    recall  f1-score   support

           0       0.97      0.99      0.98      6274
           1       0.96      0.86      0.91      1270
    accuracy                           0.97      7544
   macro avg       0.97      0.93      0.95      7544
weighted avg       0.97      0.97      0.97      7544

AUC-ROC = 0. 9932159091764788
====================================================
```




```
Classification Report for Test dataset
=====================================================
              precision    recall  f1-score   support

           0       0.94      0.98      0.96      3090
           1       0.87      0.69      0.77       626
    accuracy                           0.93      3716
   macro avg       0.90      0.83      0.86      3716
weighted avg       0.93      0.93      0.93      3716

AUC-ROC = 0.9608641707248986
===================================================

Accuracy_Train 0.9707051961823966
Accuracy_Test 0.930032292787944
```



Model-5: Decision Tree

Definition:

Given a data of attributes together with its classes, a decision tree produces a sequence of rules that can be used to classify the data.

Advantages:

- Decision Tree is simple to understand and visualise, requires little data preparation, and can handle both numerical and categorical data.

Disadvantages:

- Decision tree can create complex trees that do not generalise well, and decision trees can be unstable because small variations in the data might result in a completely different tree being generated.

Hyper parameter tuning:

```
#Number of trees in random forest
        estimators = [30,50,100,300,500]

# Number of features to consider at every split
        max_features = ['auto', 'sqrt','log2']

# Maximum number of depth in each tree:
        max_depth = [i for i in range(5,25,2)]

# Minimum number of samples to consider to split a node:
        min_samples_split = [2, 5, 10, 15, 20, 50, 100]

# Minimum number of samples to consider at each leaf node:
        min_samples_leaf = [5, 4, 7]
```

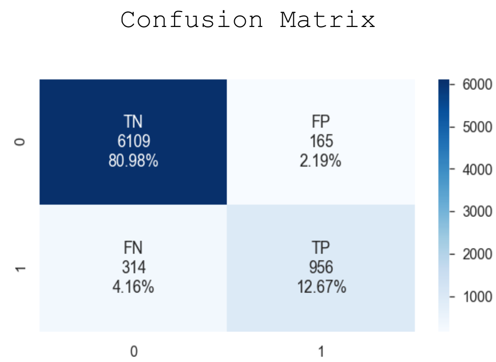
```
Fitting 5 folds for each of 10 candidates, totalling 50 fits
```

```
DecisionTreeClassifier(max_depth=29, max_features='auto', min_samples_leaf=4,
                       min_samples_split=10)
```

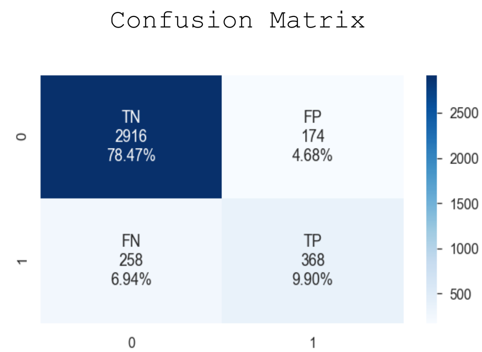
Best_params:

```
{'min_samples_split': 10, 'min_samples_leaf': 4, 'max_features': 'auto'
, 'max_depth': 29}
```

```
Classification Report for Train dataset
=====================================================

              precision    recall  f1-score   support

           0       0.95      0.97      0.96      6274
           1       0.85      0.75      0.80      1270
    accuracy                           0.94      7544
   macro avg       0.90      0.86      0.88      7544
weighted avg       0.93      0.94      0.93      7544

AUC-ROC = 0.981229621560295

====================================================
```




```
Classification Report for Test dataset
=====================================================

               precision    recall  f1-score   support

           0        0.92      0.95      0.94      3090
           1        0.71      0.60      0.65       626
     accuracy                           0.89      3716
   macro avg        0.82      0.78      0.79      3716
weighted avg        0.86      0.89      0.89      3716

AUC-ROC = 0.8598216962891736

=====================================================

Accuracy_Train 0.9365058324496288
Accuracy_Test 0.883745963401507
```



Model-6: Random Forest

Definition:

Random forest classifier is a meta-estimator that fits several decision trees on various sub-samples of datasets and uses average to improve the predictive accuracy of the model and controls over-fitting. The sub-sample size is always the same as the original input sample size, but the samples are drawn with replacement.

Advantages:

- Reduction in over-fitting and random forest classifier is more accurate than decision trees in most cases.

Disadvantages:

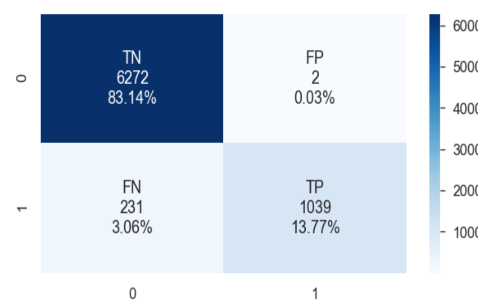
- Slow real time prediction, difficult to implement, and complex algorithm.

The main parameters used by a Random Forest Classifier are:

- criterion = the function used to evaluate the quality of a split.
- max_depth = maximum number of levels allowed in each tree.
- max_features = maximum number of features considered when splitting a node.
- min_samples_leaf = minimum number of samples which can be stored in a tree leaf.
- min_samples_split = minimum number of samples necessary in a node to cause node splitting.
- n_estimators = number of trees in the ensemble.

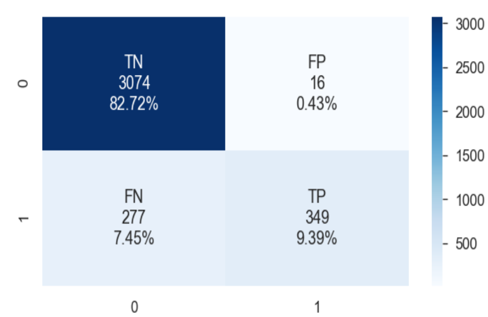
```
Fitting 5 folds for each of 20 candidates, totalling 100 fits

RandomForestClassifier(max_depth=21, min_samples_leaf=4, n_estimators=1000)

Classification Report for Train dataset
==================================================

              precision    recall  f1-score   support
           0       0.96      1.00      0.98      6274
           1       1.00      0.82      0.90      1270
    accuracy                           0.97      7544
   macro avg       0.98      0.91      0.94      7544
weighted avg       0.97      0.97      0.97      7544

AUC-ROC = 0.9990609916189548

===================================================
```

Confusion Matrix




```
Classification Report for Test dataset
==================================================

              precision    recall  f1-score   support
           0       0.92      0.99      0.95      3090
           1       0.95      0.55      0.70       626
    accuracy                           0.92      3716
   macro avg       0.94      0.77      0.83      3716
weighted avg       0.92      0.92      0.91      3716

AUC-ROC = 0.9753512826080213

==================================================
```

Confusion Matrix




```
Accuracy_Train 0.9691145281018028
Accuracy_Test 0.9211517761033369
```

Model-7: Support Vector Machine

Definition:

Support vector machine is a representation of the training data as points in space separated into categories by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall.

Advantages:

Effective in high dimensional spaces and uses a subset of training points in the decision function so it is also memory efficient. The SVM algorithm has a feature to ignore outliers and find the hyper-plane that has the maximum margin. Hence, we can say, SVM classification is robust to outliers. The best part is, SVM can also classify non-linear data.

Disadvantages:

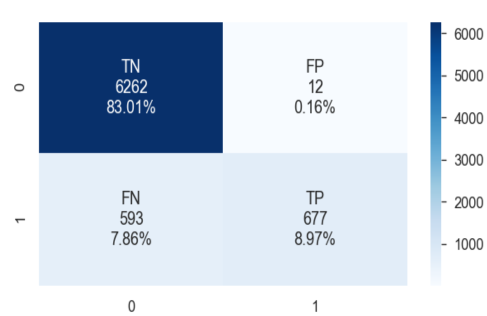
The algorithm does not directly provide probability estimates, these are calculated using an expensive five-fold cross-validation. It does not perform well when we have large data set because the required training time is higher It also does not perform very well, when the data set has more noise i.e., target classes are overlapping.

Note:

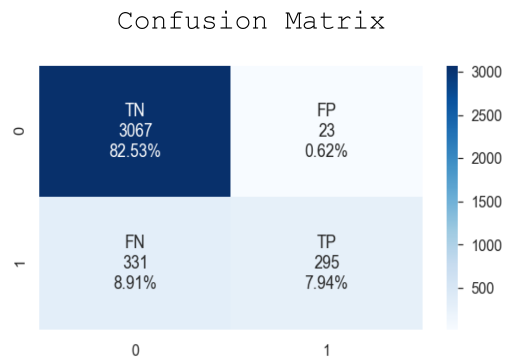
- In SVM, to avoid overfitting, we choose a Soft Margin, instead of a Hard one i.e., we let some data points enter our margin intentionally (but we still penalize it) so that our classifier does not overfit on our training sample. Here comes an important parameter Gamma (γ), which control Overfitting in SVM. The higher the gamma, the higher the hyperplane tries to match the training data. Therefore, choosing an optimal gamma to avoid Overfitting as well as Underfitting is the key.
- Linear SVM kernel is used if we have many features (>1000) because it is more likely that the data is linearly separable in high dimensional space.

```
SVC(C=0.2, degree=5, gamma='auto', kernel='poly', probability=True)

Classification Report for Train dataset
=================================================

              precision    recall  f1-score   support

           0       0.91      1.00      0.95      6274
           1       0.98      0.53      0.69      1270
    accuracy                           0.92      7544
   macro avg       0.95      0.77      0.82      7544
weighted avg       0.93      0.92      0.91      7544

AUC-ROC = 0.9610206727426523

=================================================
```

Confusion Matrix



```
Classification Report for Test dataset
==================================================

              precision    recall  f1-score   support
           0       0.90      0.99      0.95      3090
           1       0.93      0.47      0.62       626
    accuracy                           0.90      3716
   macro avg       0.92      0.73      0.79      3716
weighted avg       0.91      0.90      0.89      3716

AUC-ROC = 0.924991211472647

=================================================

Accuracy_Train 0.9198038176033935
Accuracy_Test 0.9047362755651238
```



Model Comparision for imbalanced data

Models with all Features:

	Model	Dataset	Resample	Precision	Recall	f1-score	Accuracy	AUC-ROC
0	Logistic Regression	train	actual	0.367454	0.771654	0.497841	0.737937	0.824815
1	Logistic Regression	test	actual	0.386919	0.784345	0.518206	0.754306	0.840057
2	Naive Bayes	train	actual	0.529691	0.526772	0.528227	0.841596	0.797342
3	Naive Bayes	test	actual	0.538079	0.519169	0.528455	0.843918	0.807516
4	Stochastic Gradient Descent	train	actual	0.699284	0.230709	0.346951	0.853791	0.811238
5	Stochastic Gradient Descent	test	actual	0.738095	0.247604	0.370813	0.858450	0.828071
6	K-Nearest Neighbours	train	actual	0.959684	0.862205	0.908337	0.970705	0.993216
7	K-Nearest Neighbours	test	actual	0.867470	0.690096	0.768683	0.930032	0.960864
8	Decision Tree	train	actual	0.852810	0.752756	0.799665	0.936506	0.981230
9	Decision Tree	test	actual	0.678967	0.587859	0.630137	0.883746	0.859822
10	Random Forest	train	actual	0.998079	0.818110	0.899178	0.969115	0.999061
11	Random Forest	test	actual	0.956164	0.557508	0.704339	0.921152	0.975351
12	Support Vector Machine	train	actual	0.982583	0.533071	0.691169	0.919804	0.961021
13	Support Vector Machine	test	actual	0.927673	0.471246	0.625000	0.904736	0.924991

Inference:

The Churn problem is about client retention, so it is worth to check about false positives and false negatives, so precision and recall metrics are a must for this situation. F1 Score is used to check the quality of the model predictions, as the metric is a harmonic mean of precision and recall.

A comparative analysis was done on the dataset using 7 classifier models:

- Logistic Regression
- Naive Bayes
- Stochastic Gradient Descent
- K-Nearest Neighbours
- Decision Tree
- Random Forest
- Support Vector Machine.

5.3 Interpretation of the model(s)

From the above, it can be seen on the actual imbalanced dataset, all 7 classifier models were not able to generalize well on the minority class compared to the majority class. As a result, most of the negative class samples were correctly classified. Due to this, there was less FP compared to more FN.

To compare their performances, as a first step, I applied Stratified kfold cross-validation method which is a technique that partitions the data into subsets, training the data on a subset and use the other subset to evaluate the model's performance.

One possible way to improve the results is SMOTE as data resampling.

SMOTE:

Imbalanced classification involves developing predictive models on classification datasets that have a severe class imbalance.

Generally, the minority/positive class is the class of interest and we aim to achieve the best results in this class rather. If the imbalanced data is not treated beforehand, then this will degrade the performance of the classifier model. Most of the predictions will correspond to the majority class and treat the minority class features as noise in the data and ignore them. This will result in a high bias in the model.

The challenge of working with imbalanced datasets is that most machine learning techniques will ignore, and in turn have poor performance on, the minority class, although typically it is performance on the minority class that is most important.
One approach to addressing imbalanced datasets is to oversample the minority class. The simplest approach involves duplicating examples in the minority class, although these examples do not add any new information to the model. Instead, new examples can be synthesized from the existing examples. This is a type of data augmentation for the minority class and is referred to as the Synthetic Minority Oversampling Technique or SMOTE for short.

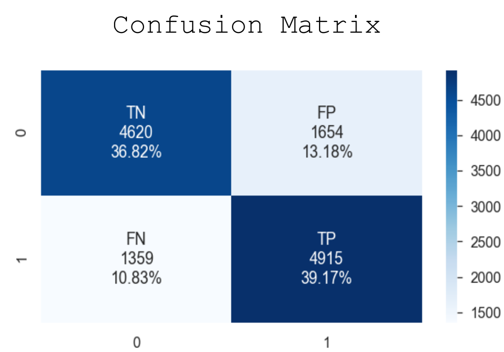
Note:
The SMOTE and its related techniques are only applied to the training dataset so that we fit our algorithm properly on the data. The test data remains unchanged so that it correctly represents the original data.

```
Before Counter({0: 6274, 1: 1270})
After Counter({0: 6274, 1: 6274})

After OverSampling, the shape of X_train: (12548, 17)
After OverSampling, the shape of y_train: (12548,)
```

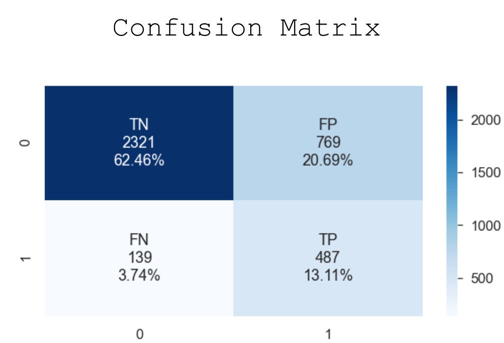
7. Model Tuning

Model-1: Logistic Regression - SMOTE Resampling

```
LogisticRegression(C=10.0)


Classification Report for Train dataset
==================================================

              precision    recall  f1-score   support

           0       0.77      0.74      0.75      6274
           1       0.75      0.78      0.77      6274
    accuracy                           0.76     12548
   macro avg       0.76      0.76      0.76     12548
weighted avg       0.76      0.76      0.76     12548

AUC-ROC = 0.8330299695074643

==================================================
```



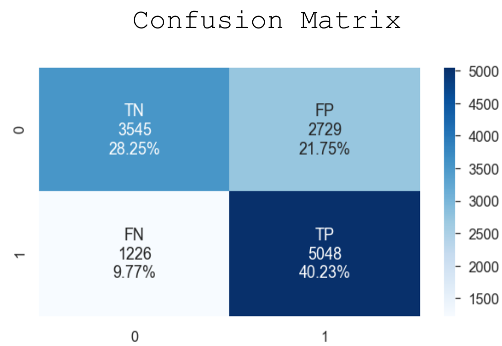

```
Classification Report for Test dataset
==================================================

              precision    recall  f1-score   support

           0       0.94      0.75      0.84      3090
           1       0.39      0.78      0.52       626
    accuracy                           0.75      3716
   macro avg       0.67      0.76      0.68      3716
weighted avg       0.85      0.76      0.78      3716

AUC-ROC = 0.840352264855196

=================================================
```



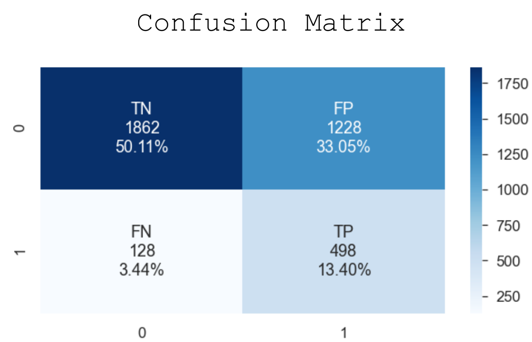

```
Accuracy_Train 0.7598820529167994
Accuracy_Test 0.7556512378902045
```

Model-2: Naïve Bayes - SMOTE Resampling

```
GaussianNB()

Classification Report for Train dataset
==================================================

              precision    recall  f1-score   support

           0       0.74      0.57      0.64      6274
           1       0.65      0.80      0.72      6274
    accuracy                           0.68     12548
   macro avg       0.70      0.68      0.68     12548
weighted avg       0.70      0.68      0.68     12548

AUC-ROC = 0.8007458817496884

==================================================
```

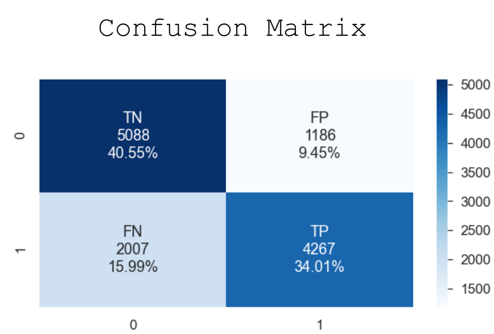



```
Classification Report for Test dataset
=================================================

              precision    recall  f1-score   support

           0       0.94      0.60      0.73      3090
           1       0.29      0.80      0.42       626
    accuracy                           0.64      3716
   macro avg       0.61      0.70      0.58      3716
weighted avg       0.83      0.64      0.68      3716

AUC-ROC =0.801571078507398

=================================================

Accuracy_Train 0.6848103283391775
Accuracy_Test 0.635091496232508
```

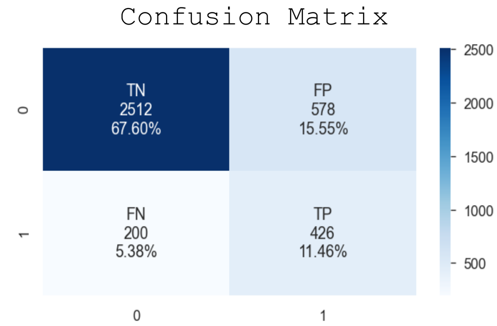


Model-3: Stochastic Gradient Descent - SMOTE Resampling

```
SGDClassifier(loss='log', max_iter=1500, random_state=123)

Classification Report for Train dataset
==================================================

              precision     recall  f1-score    support
           0       0.73       0.78      0.76       6274
           1       0.77       0.71      0.74       6274
    accuracy                            0.75      12548
   macro avg       0.75       0.75      0.75      12548
weighted avg       0.75       0.75      0.75      12548

AUC-ROC = 0.8254741067491779

==================================================
```

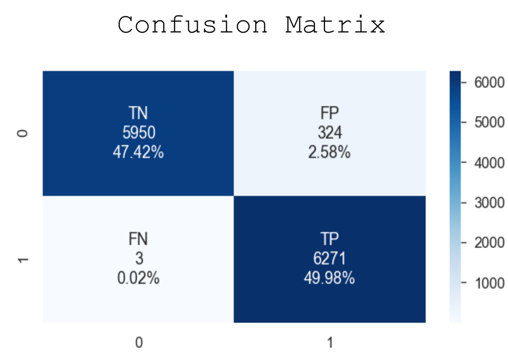


```
Classification Report for Test dataset
==================================================
              precision    recall  f1-score   support

           0    0.93       0.79       0.85      3090
           1    0.41       0.72       0.52       626
    accuracy                          0.78      3716
   macro avg    0.67       0.75       0.69      3716
weighted avg    0.84       0.78       0.80      3716

AUC-ROC = 0.8297589875616489

==================================================

Accuracy_Train 0.7455371373924131
Accuracy_Test 0.7906350914962325
```

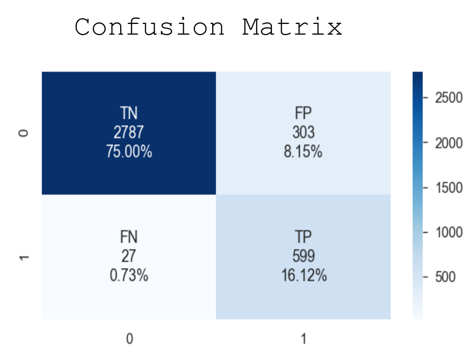


Model-4: K-Nearest Neighbours - SMOTE Resampling

```
KNeighborsClassifier()

Classification Report for Train dataset
=================================================

              precision   recall  f1-score   support

           0    1.00       0.95       0.97      6274
           1    0.95       1.00       0.97      6274
    accuracy                          0.97     12548
   macro avg    0.98       0.97       0.97     12548
weighted avg    0.98       0.97       0.97     12548

AUC-ROC = 0.9998555117999417

=================================================
```



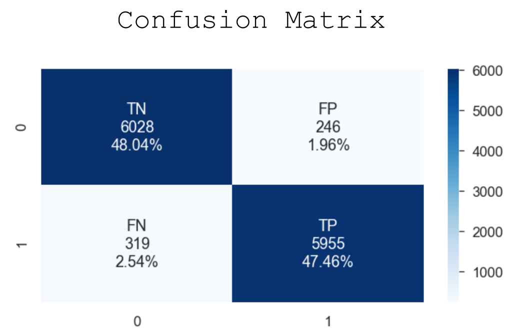

```
Classification Report for Test dataset
====================================================

               precision     recall  f1-score   support

           0      0.99       0.90       0.94      3090
           1      0.66       0.96       0.78       626
    accuracy                             0.91      3716
   macro avg      0.83       0.93       0.86      3716
weighted avg      0.94       0.91       0.92      3716

AUC-ROC = 0.9764170207926218

====================================================

Accuracy_Train 0.9739400701306982
Accuracy_Test 0.9111948331539289
```



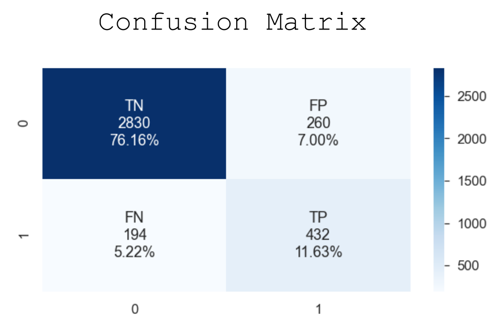
Model-5: Decision Tree - SMOTE Resampling

```
DecisionTreeClassifier(max_depth=29, max_features='auto', min_samples_leaf=4,
 min_samples_split=10)
```

```
Classification Report for Train dataset
=======================================================

             precision    recall  f1-score   support
           0    0.95        0.96      0.96      6274
           1    0.96        0.95      0.95      6274
    accuracy                          0.95     12548
   macro avg    0.96        0.95      0.95     12548
weighted avg    0.96        0.95      0.95     12548

AUC-ROC = 0.9946341083710023

=======================================================
```




```
Classification Report for Test dataset
========================================================

             precision    recall  f1-score    support
           0    0.94        0.92       0.93     3090
           1    0.62        0.69       0.66      626
    accuracy                           0.88     3716
   macro avg    0.78        0.80       0.79     3716
weighted avg    0.88        0.88       0.88     3716

AUC-ROC = 0.882850222815017

========================================================
```

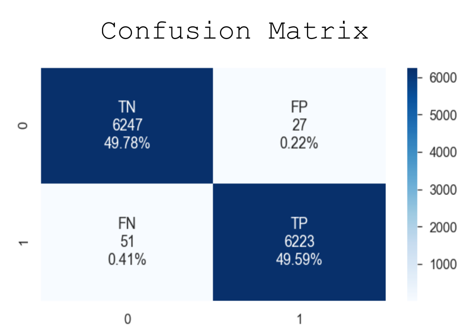



```
Accuracy_Train 0.9549729040484539
Accuracy_Test 0.8778256189451022
```

Model-6: Random Forest - SMOTE Resampling

```
RandomForestClassifier(max_depth=21, n_estimators=1000, min_samples_leaf=4)
```

```
Classification Report for Train dataset
=========================================================
             precision     recall  f1-score    support
           0    0.99       1.00       0.99       6274
           1    1.00       0.99       0.99       6274
    accuracy                           0.99      12548
   macro avg    0.99       0.99       0.99      12548
weighted avg    0.99       0.99       0.99      12548

AUC-ROC = 0.9998241245170982

=========================================================
```

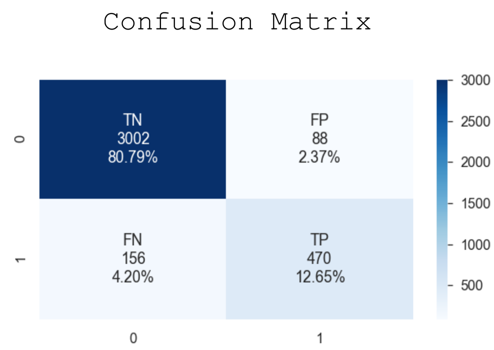


Classification Report for Test dataset
==

	precision	recall	f1-score	support
0	0.95	0.97	0.96	3090
1	0.84	0.75	0.79	626
accuracy			0.93	3716
macro avg	0.90	0.86	0.88	3716
weighted avg	0.93	0.93	0.93	3716

AUC-ROC = 0.9728755027554619

==



Accuracy_Train 0.9937838699394326
Accuracy_Test 0.9343379978471474

Model-7: Support Vector Machine

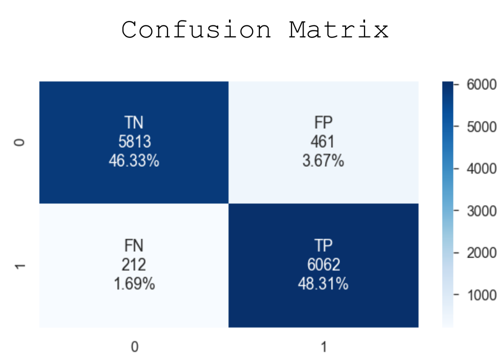
```
SVC(C=0.2, degree=5, gamma='auto', kernel='poly', probability=True,
       random_state=123)
```

Classification Report for Train dataset
==

	precision	recall	f1-score	support
0	0.96	0.93	0.95	6274
1	0.93	0.97	0.95	6274
accuracy			0.95	12548
macro avg	0.95	0.95	0.95	12548
weighted avg	0.95	0.95	0.95	12548

AUC-ROC = 0.9864007579082488

==

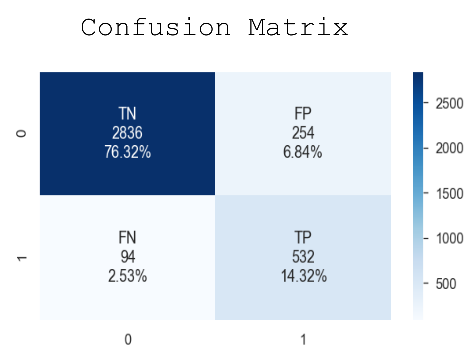


Classification Report for Test dataset
==

	precision	recall	f1-score	support
0	0.97	0.92	0.94	3090
1	0.68	0.85	0.75	626
accuracy			0.91	3716
macro avg	0.82	0.88	0.85	3716
weighted avg	0.92	0.91	0.91	3716

AUC-ROC = 0.9484382269921524

==



Accuracy_Train 0.9463659547338221
Accuracy_Test 0.9063509149623251

Model Comparison for balanced data

Models with all Features and balanced dataset:

	Model	Dataset	Resample	Precision	Recall	f1-score	Accuracy	AUC-ROC
0	Logistic Regression	train	smote	0.748211	0.783392	0.765397	0.759882	0.833030
1	Logistic Regression	test	smote	0.387739	0.777955	0.517535	0.755651	0.840352
2	Naive Bayes	train	smote	0.649093	0.804590	0.718525	0.684810	0.800746
3	Naive Bayes	test	smote	0.288528	0.795527	0.423469	0.635091	0.801571
4	Stochastic Gradient Descent	train	smote	0.782505	0.680108	0.727722	0.745537	0.825474
5	Stochastic Gradient Descent	test	smote	0.424303	0.680511	0.522699	0.790635	0.829759
6	K-Nearest Neighbours	train	smote	0.950872	0.999522	0.974590	0.973940	0.999856
7	K-Nearest Neighbours	test	smote	0.664080	0.956869	0.784031	0.911195	0.976417
8	Decision Tree	train	smote	0.960329	0.949155	0.954709	0.954973	0.994634
9	Decision Tree	test	smote	0.624277	0.690096	0.655539	0.877826	0.882850
10	Random Forest	train	smote	0.995680	0.991871	0.993772	0.993784	0.999824
11	Random Forest	test	smote	0.842294	0.750799	0.793919	0.934338	0.972876
12	Support Vector Machine	train	smote	0.929327	0.966210	0.947410	0.946366	0.986401
13	Support Vector Machine	test	smote	0.676845	0.849840	0.753541	0.906351	0.948438

Inference:

After oversampling, a clear surge in Recall is seen on the test data. To understand this better, a comparative table is shown above for all 7 models.

Tuning via Hyperparatemers:

- To improve the overall performance when it comes to Recall metric, I tuned classifiers hyperparameters using GridSearchCV and RandomizedSearchCV for Decision Tree and Logistic Regression even applied smote to balance out the imbalance in dataset.

- The **most expressive** improvement came from SVM model, which shifted from 0.47 to 0.86. This means the implemented ML model based on SVM **delivers 65% of precision while predicting customer churn.** On the other hand, it has a **high rate of false positives**, which means 7.75% of satisfied customers (288) can be incorrectly predicted as churn. These results can be seen in the above correlation matrix, where 1 means Churn and 0 means not Churn.

- For the predictions made by the model and based on the precision and recall scores, as F1 Score try to show a balance between these two metrics, the precision was near 83%, what means that the model predicts correctly 83% of classified clients as churned, on other hand, the recall was good, where around 79% of the actually churned clients was predict correctly.

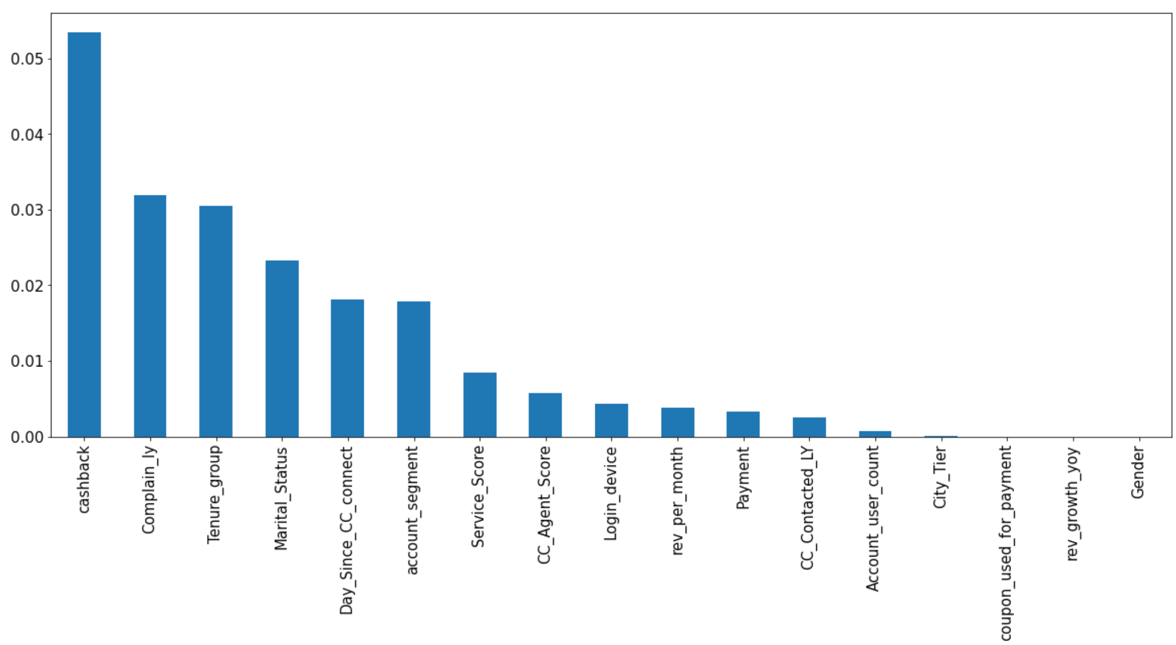
The top-performers were.

- **K-Nearest Neighbours (0.95 Recall score)**
- **Support Vector Machine (0.84 Recall score)**
- **Random Forest (0.75 Recall score)**

But there is still room for optimization.

Mutual information-based feature selection

```
cashback                   0.053366
Complain_ly                0.031839
Tenure_group               0.030476
Marital_Status             0.023284
Day_Since_CC_connect       0.018164
account_segment            0.017842
Service_Score              0.008455
CC_Agent_Score             0.005698
Login_device               0.004311
rev_per_month              0.003792
Payment                    0.003317
CC_Contacted_LY            0.002577
Account_user_count         0.000757
City_Tier                  0.000122
coupon_used_for_payment    0.000000
rev_growth_yoy             0.000000
Gender                     0.000000
```

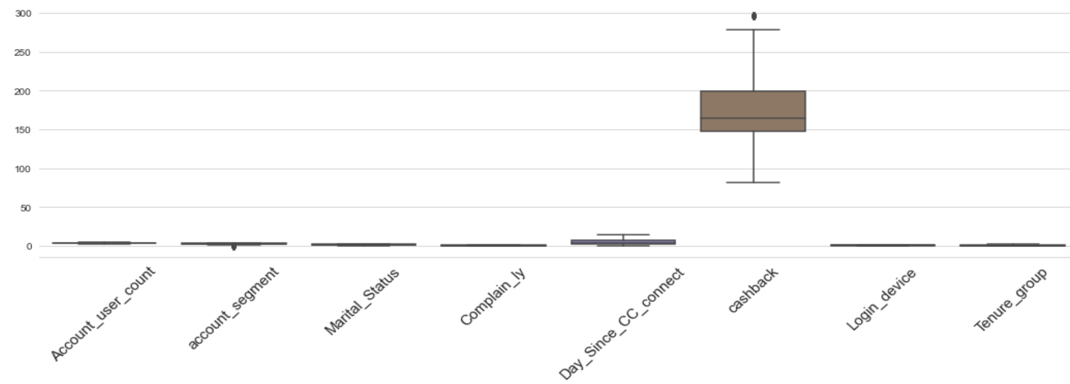


Top 8 features are selected:

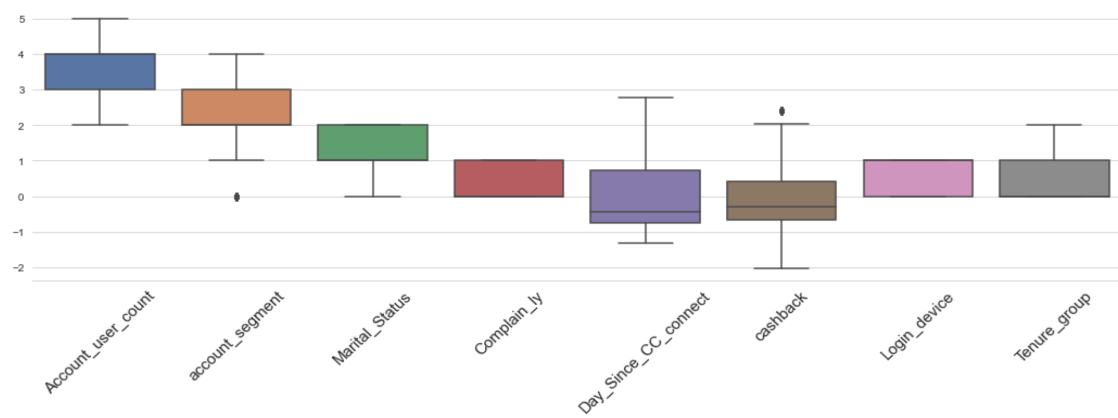
```
['Account_user_count', 'account_segment', 'Marital_Status', 'Complain_ly',
 'Day_Since_CC_connect', 'cashback', 'Login_device', 'Tenure_group']
```

Feature Scaling - Standardization:

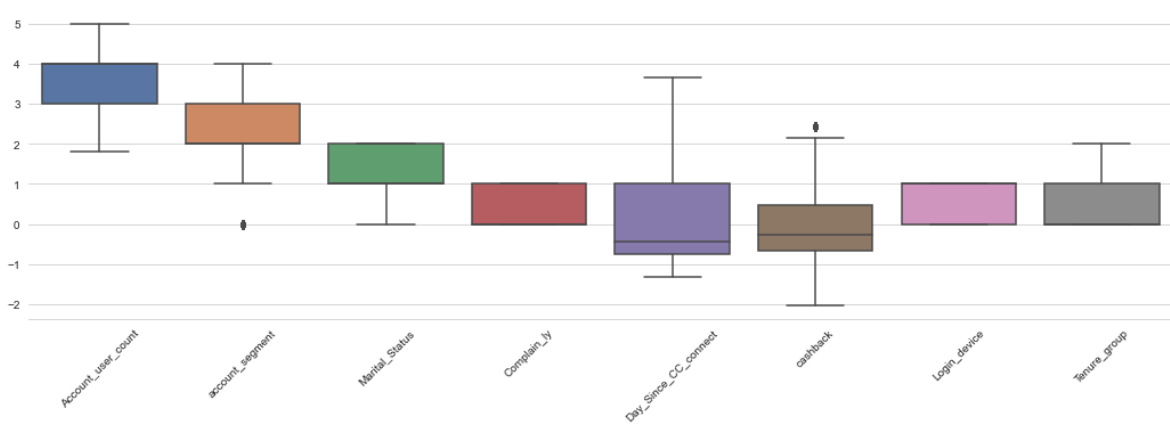
Before Scaling:



After Scaling Train:



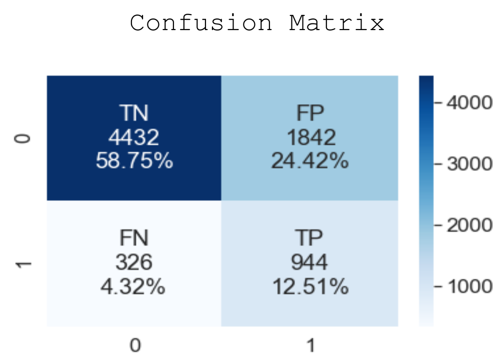
After Scaling Test:



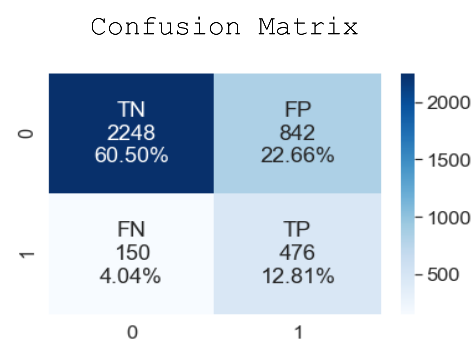
Model Building – with mutual features and Imbalanced data

Model-1: Logistic Regression

```
LogisticRegression(C=10.0, class_weight='balanced')


Classification Report for Train dataset
==================================================

              precision    recall  f1-score   support

           0       0.93      0.71      0.80      6274
           1       0.34      0.74      0.47      1270
    accuracy                           0.71      7544
   macro avg       0.64      0.72      0.63      7544
weighted avg       0.83      0.71      0.75      7544

AUC-ROC = 0.7930527561565165

==================================================
```




```
Classification Report for Test dataset
==================================================

              precision    recall  f1-score   support

           0       0.94      0.73      0.82      3090
           1       0.36      0.76      0.49       626
    accuracy                           0.73      3716
   macro avg       0.65      0.74      0.65      3716
weighted avg       0.84      0.73      0.76      3716

AUC-ROC = 0.8086145662086293

==================================================
```



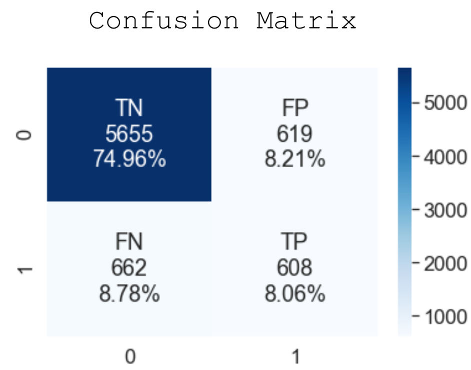

```
Accuracy_Train 0.7126193001060446
Accuracy_Test 0.7330462863293864
```

Inferences:

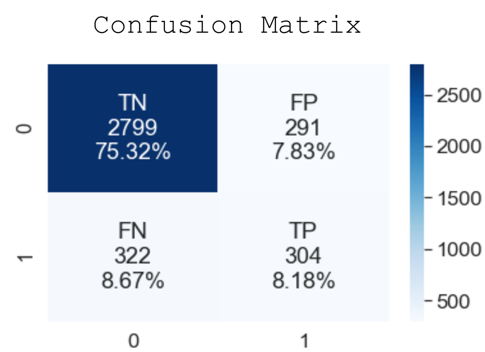
The Logistic Regression model with mutual information-based features has larger rate of false positives than false negatives which is ok. Practically, it means you will be able to engage with 12.81% of the customers who will churn, but you will miss the other 4.04%. Also, you may have 22.66% who are incorrectly predicted as churned.

Model-2: Naïve Bayes

```
GaussianNB()


Classification Report for Train dataset
==================================================

              precision    recall  f1-score   support

           0       0.90      0.90      0.90      6274
           1       0.50      0.48      0.49      1270
    accuracy                           0.83      7544
   macro avg       0.70      0.69      0.69      7544
weighted avg       0.83      0.83      0.83      7544

AUC-ROC = 0.7740609916189549

==================================================
```




```
Classification Report for Test dataset
==================================================

              precision    recall  f1-score   support

           0       0.90      0.91      0.90      3090
           1       0.51      0.49      0.50       626
    accuracy                           0.84      3716
   macro avg       0.70      0.70      0.70      3716
weighted avg       0.83      0.84      0.83      3716

AUC-ROC = 0.7881574593918339

=================================================

Accuracy_Train 0.8301961823966065
Accuracy_Test 0.8350376749192681
```



Inferences:

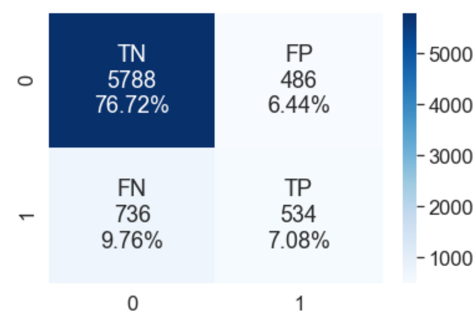
The Naïve Bayes model with mutual information-based features has larger rate of false negatives than false positives which is not ok. Practically, it means you will be able to engage with only 8.18% of the customers who will churn, but you will miss the other 8.67%. Also, you may have 7.83% who are incorrectly predicted as churned. Hence this model is not good for prediction.

Model-3: Stochastic Gradient Descent

```
SGDClassifier(loss='log', max_iter=1500, random_state=123)
```

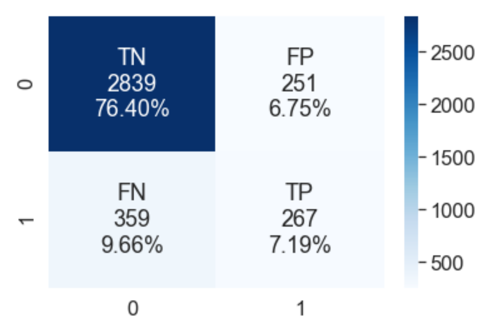
```
Classification Report for Train dataset
=====================================================

              precision    recall  f1-score   support

           0       0.89      0.92      0.91      6274
           1       0.52      0.42      0.47      1270
    accuracy                           0.84      7544
   macro avg       0.71      0.67      0.69      7544
weighted avg       0.83      0.84      0.83      7544

AUC-ROC = 0.787979324747301

=====================================================
```

Confusion Matrix




```
Classification Report for Test dataset
=====================================================

              precision    recall  f1-score   support

           0       0.89      0.92      0.90      3090
           1       0.52      0.43      0.47       626
    accuracy                           0.84      3716
   macro avg       0.70      0.67      0.68      3716
weighted avg       0.83      0.84      0.83      3716

AUC-ROC = 0.8047574366450573

=====================================================
```

Confusion Matrix



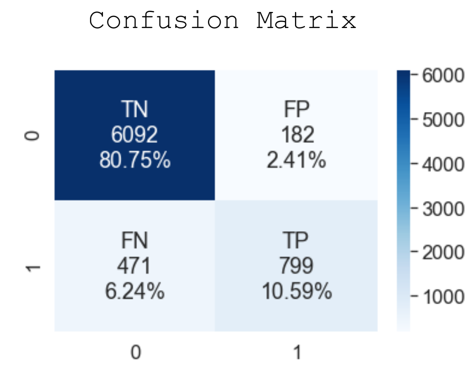

```
Accuracy_Train 0.838016967126193
Accuracy_Test 0.8358449946178687
```

Inferences:

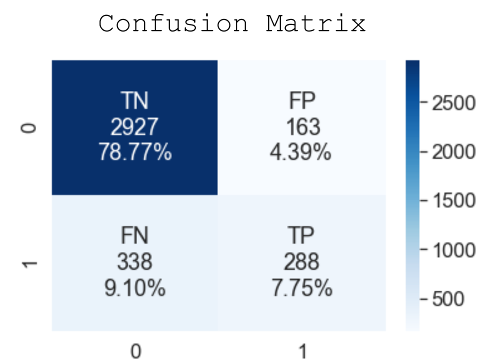
The Stochastic Gradient Descent model with mutual information-based features has larger rate of false negatives than false positives which is not ok. Practically, it means you will be able to engage with only 7.19% of the customers who will churn, but you will miss the other 9.66%. Also, you may have 6.75% who are incorrectly predicted as churned. Hence this model is not good for prediction.

Model-4: K-Nearest Neighbours

```
KNeighborsClassifier()

Classification Report for Train dataset
==================================================

              precision    recall  f1-score   support
           0       0.93      0.97      0.95      6274
           1       0.81      0.63      0.71      1270
    accuracy                           0.91      7544
   macro avg       0.87      0.80      0.83      7544
weighted avg       0.91      0.91      0.91      7544

AUC-ROC = 0.9581006729434561

==================================================
```

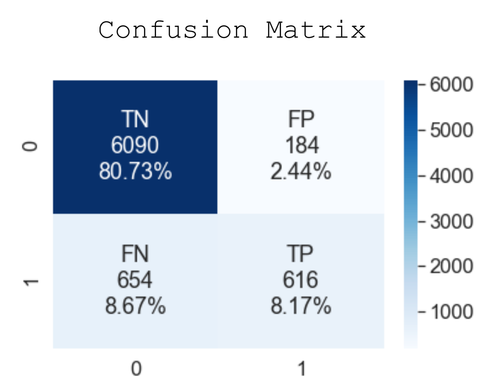



```
Classification Report for Test dataset
==================================================

              precision    recall  f1-score   support
           0       0.90      0.95      0.92      3090
           1       0.64      0.46      0.53       626
    accuracy                           0.87      3716
   macro avg       0.77      0.70      0.73      3716
weighted avg       0.85      0.87      0.86      3716

AUC-ROC = 0.8571872059720629

==================================================

Accuracy_Train 0.913441145281018
Accuracy_Test 0.8651776103336921
```

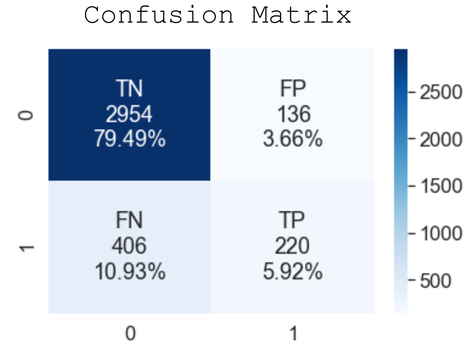


Model-5: Decision Tree

```
Fitting 5 folds for each of 10 candidates, totalling 50 fits
DecisionTreeClassifier(max_depth=21, max_features='sqrt', min_samples_leaf=5,
                      min_samples_split=10)

Classification Report for Train dataset
==================================================

              precision    recall  f1-score   support
           0       0.90      0.97      0.94      6274
           1       0.77      0.49      0.60      1270
    accuracy                           0.89      7544
   macro avg       0.84      0.73      0.77      7544
weighted avg       0.88      089      0.88      7544

AUC-ROC = 0.9326713294963089

==================================================
```



```
Classification Report for Test dataset
==================================================

              precision    recall  f1-score   support
           0       0.88      0.96      0.92      3090
           1       0.62      0.35      0.45       626
    accuracy                           0.85      3716
   macro avg       0.75      0.65      0.68      3716
weighted avg       0.84      0.85      0.84      3716

AUC-ROC = 0.8304936050539202

==================================================
```

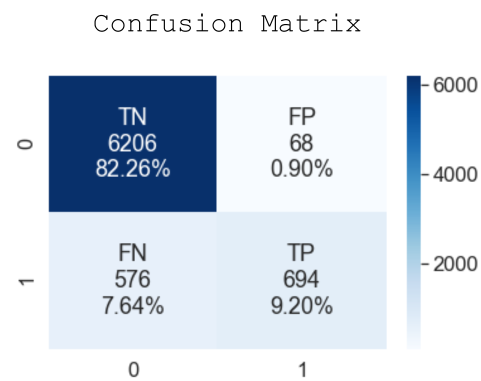



```
Accuracy_Train 0.8889183457051962
Accuracy_Test 0.8541442411194833
```

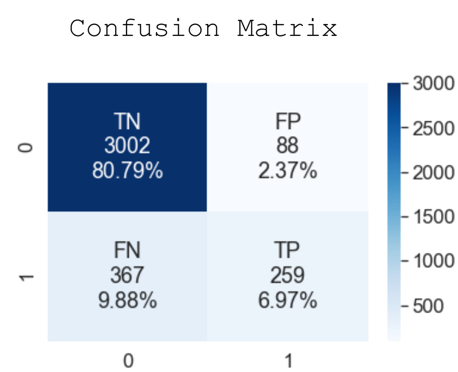
Model-6: Random Forest

```
Fitting 5 folds for each of 20 candidates, totalling 100 fits

RandomForestClassifier(max_depth=23, max_features='log2', min_samples_leaf=4,
                       min_samples_split=5, n_estimators=500)
```

```
Classification Report for Train dataset
==================================================

             precision    recall  f1-score   support
           0       0.92      0.99      0.95      6274
           1       0.91      0.55      0.68      1270
    accuracy                           0.91      7544
   macro avg       0.91      0.77      0.82      7544
weighted avg       0.91      0.91      0.91      7544

AUC-ROC = 0.9733792630001581

==================================================
```




```
Classification Report for Test dataset
==================================================

              precision    recall  f1-score   support
           0       0.91      0.98      0.94      3090
           1       0.83      0.54      0.66       626
    accuracy                           0.90      3716
   macro avg       0.87      0.76      0.80      3716
weighted avg       0.90      0.90      0.90      3716

AUC-ROC = 0.8998392216466599

==================================================
```




```
Accuracy_Train 0.9146341463414634
Accuracy_Test 0.8775565123789021
```

Inferences:

The Random Forest model with mutual information-based features has larger rate of false negatives than false positives which is not ok. Practically, it means you will be able to engage with 6.97% of the customers who will churn, but you will miss the other 9.88%. Also, you may have 2.37% who are incorrectly predicted as churned. Let's check after applying smote.

Model Comparison for imbalanced data

Models with mutual information Features and imbalanced dataset:

	Model	Dataset	Resample	Precision	Recall	f1-score	Accuracy	AUC-ROC
0	Logistic Regression	train	actual	0.338837	0.743307	0.465483	0.712619	0.793053
1	Logistic Regression	test	actual	0.361153	0.760383	0.489712	0.733046	0.808615
2	Naive Bayes	train	actual	0.495518	0.478740	0.486984	0.830196	0.774061
3	Naive Bayes	test	actual	0.510924	0.485623	0.497952	0.835038	0.788157
4	Stochastic Gradient Descent	train	actual	0.523529	0.420472	0.466376	0.838017	0.787979
5	Stochastic Gradient Descent	test	actual	0.515444	0.426518	0.466783	0.835845	0.804757
6	K-Nearest Neighbours	train	actual	0.814475	0.629134	0.709907	0.913441	0.958101
7	K-Nearest Neighbours	test	actual	0.638581	0.460064	0.534819	0.865178	0.857187
8	Decision Tree	train	actual	0.770000	0.485039	0.595169	0.888918	0.932671
9	Decision Tree	test	actual	0.617978	0.351438	0.448065	0.854144	0.830494
10	Random Forest	train	actual	0.910761	0.546457	0.683071	0.914634	0.973379
11	Random Forest	test	actual	0.746398	0.413738	0.532374	0.877557	0.899839

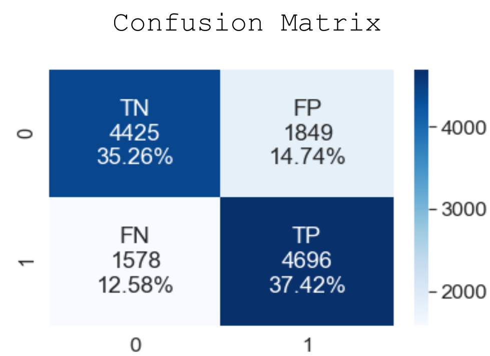
SMOTE applied:

```
Before Counter({0: 6274, 1: 1270})
After Counter({0: 6274, 1: 6274})

After OverSampling, the shape of X_train: (12548, 8)
After OverSampling, the shape of y_train: (12548,)
```

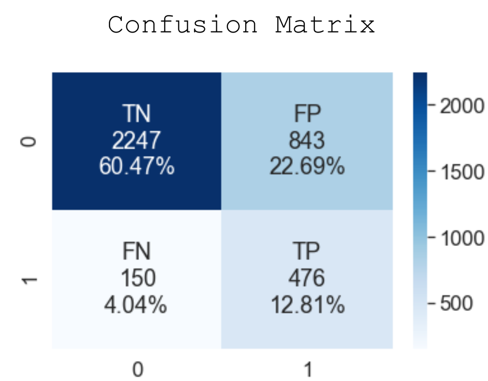
Model Building – with mutual information features and balanced data

Model-1: Logistic Regression - SMOTE Resampling

```
LogisticRegression(C=0.1)

Classification Report for Train dataset
==================================================

             precisio    recall   f1-score    support
            0     0.74     0.71       0.72       6274
            1     0.72     0.75       0.73       6274
     accuracy                         0.73      12548
    macro avg     0.73     0.73       0.73      12548
 weighted avg     0.73     0.73       0.73      12548

AUC-ROC = 0.7970134371612625

==================================================
```

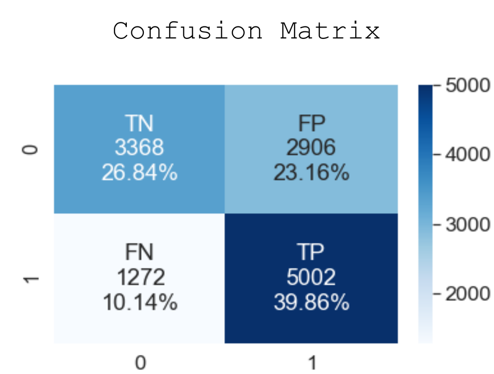



```
Classification Report for Test dataset
==================================================

             precision   recall   f1-score    support
            0     0.94     0.73       0.82       3090
            1     0.36     0.76       0.49        626
     accuracy                         0.73       3716
    macro avg     0.65     0.74       0.65       3716
 weighted avg     0.84     0.73       0.76       3716

AUC-ROC = 0.8091599718767125


===================================================

Accuracy_Train 0.7268887472107108
Accuracy_Test 0.7327771797631862
```

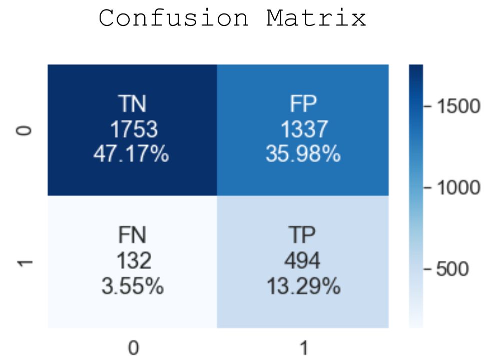


Model-2: Naïve Bayes - SMOTE Resampling

```
GaussianNB()

Classification Report for Train dataset
==================================================

             precision   recall   f1-score    support

            0    0.73      0.54       0.62       6274
            1    0.63      0.80       0.71       6274
     accuracy                         0.67      12548
    macro avg    0.68      0.67       0.66      12548
 weighted avg    0.68      0.67       0.66      12548

AUC-ROC = 0.7784171770519154

==================================================
```

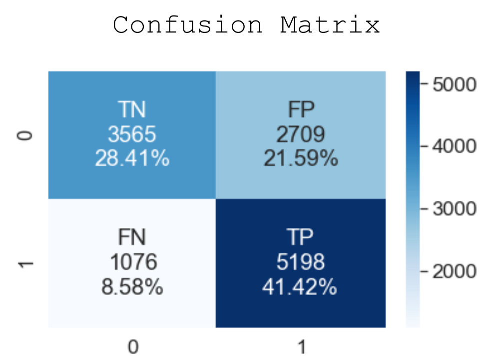


```
Classification Report for Test dataset
==================================================

              precision    recall  f1-score   support

           0       0.93      0.58      0.71      3090
           1       0.27      0.78      0.40       626
    accuracy                           0.61      3716
   macro avg       0.60      0.68      0.56      3716
weighted avg       0.82      0.61      0.66      3716

AUC-ROC = 0.7865605322745742

=================================================

Accuracy_Train 0.6670385718839655
Accuracy_Test 0.6046824542518837
```

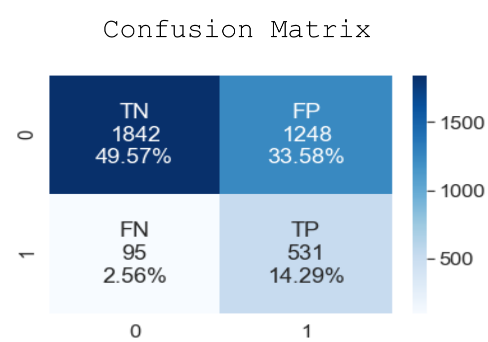


Model-3: Stochastic Gradient Descent - SMOTE Resampling

```
SGDClassifier(loss='log', max_iter=1500, random_state=123)

Classification Report for Train dataset
=================================================

              precision    recall  f1-score   support
           0       0.77      0.57      0.65      6274
           1       0.66      0.83      0.73      6274
    accuracy                           0.70     12548
   macro avg       0.71      0.70      0.69     12548
weighted avg       0.71      0.70      0.69     12548

AUC-ROC = 0.7954314469733005

=================================================
```

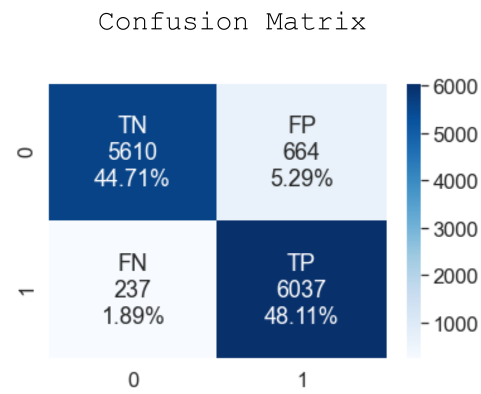



```
Classification Report for Test dataset
=================================================

              precision    recall  f1-score   support
           0       0.95      0.60      0.73      3090
           1       0.30      0.85      0.44       626
    accuracy                           0.64      3716
   macro avg       0.62      0.72      0.59      3716
weighted avg       0.84      0.64      0.68      3716

AUC-ROC = 0.8079125179647839

==================================================


Accuracy_Train 0.6983583041122091
Accuracy_Test 0.6385898815931109
```

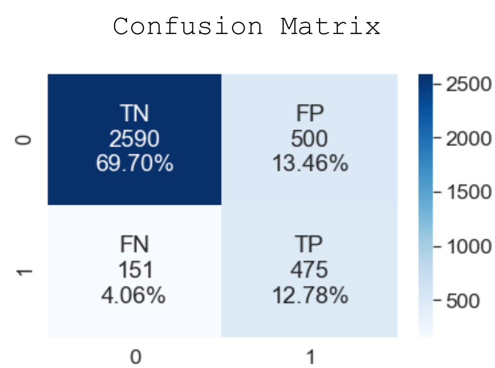


Model-4: K-Nearest Neighbours - SMOTE Resampling

```
KNeighborsClassifier()

Classification Report for Train dataset
=======================================================

              precision    recall  f1-score   support
           0       0.96      0.89      0.93      6274
           1       0.90      0.96      0.93      6274
    accuracy                           0.93     12548
   macro avg       0.93      0.93      0.93     12548
weighted avg       0.93      0.93      0.93     12548

AUC-ROC = 0.9856986786296884

======================================================
```

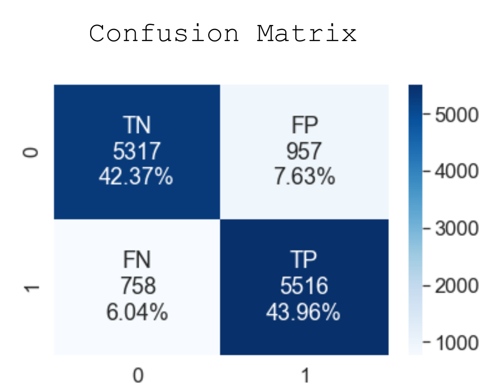



```
Classification Report for Test dataset
======================================================

              precision    recall  f1-score   support
           0       0.95      0.82      0.88      3090
           1       0.46      0.77      0.58       626
    accuracy                           0.81      3716
   macro avg       0.70      0.79      0.73      3716
weighted avg       0.86      0.81      0.83      3716

AUC-ROC = 0.8700424951146126

======================================================

Accuracy_Train 0.9281957284029327
Accuracy_Test 0.8248116254036598
```

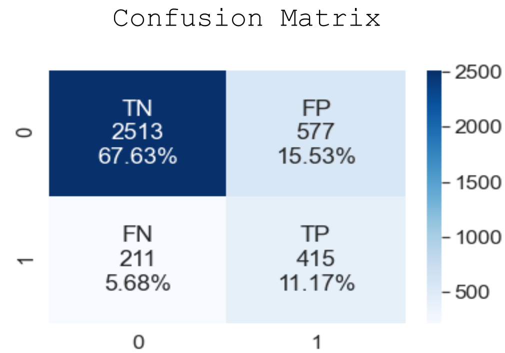


Model-5: Decision Tree - SMOTE Resampling

```
DecisionTreeClassifier(max_depth=15, max_features='sqrt', min_samples_leaf=5,
                       min_samples_split=10)

Classification Report for Train dataset
=========================================================

              precision    recall  f1-score   support
           0       0.88      0.85      0.86      6274
           1       0.85      0.88      0.87      6274
    accuracy                           0.86     12548
   macro avg       0.86      0.86      0.86     12548
weighted avg       0.86      0.86      0.86     12548

AUC-ROC = 0.9484361562597394

=========================================================
```

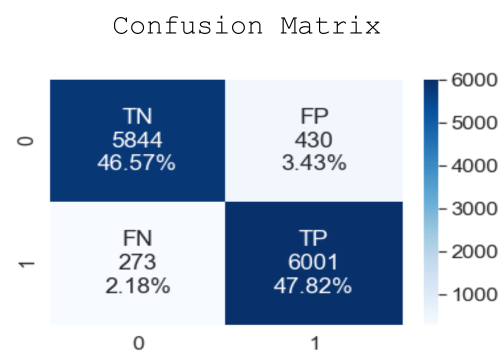


```
Classification Report for Test dataset
=================================================

              precision    recall  f1-score   support
           0       0.92      0.81      0.86      3090
           1       0.42      0.66      0.51       626
    accuracy                           0.79      3716
   macro avg       0.67      0.74      0.69      3716
weighted avg       0.84      0.79      0.81      3716

AUC-ROC = 0.8230897877312158

=================================================

Accuracy_Train 0.8633248326426523
Accuracy_Test 0.7879440258342304
```

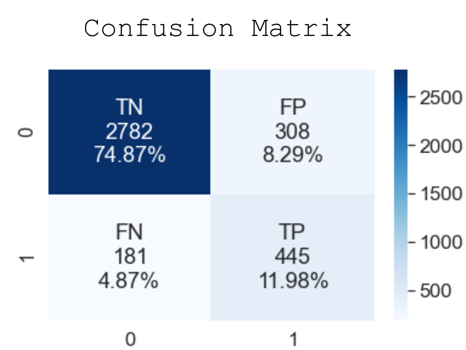


Model-6: Random Forest - SMOTE Resampling

```
RandomForestClassifier(max_depth=23, max_features='log2', min_samples_leaf=4,
                       min_samples_split=5, n_estimators=500)

Classification Report for Train dataset
=================================================

              precision    recall  f1-score   support
           0       0.96      0.93      0.94      6274
           1       0.93      0.96      0.94      6274
    accuracy                           0.94     12548
   macro avg       0.94      0.94      0.94     12548
weighted avg       0.94      0.94      0.94     12548

AUC-ROC = 0.989869935469474

=================================================
```




```
Classification Report for Test dataset
=================================================

              precision    recall  f1-score   support
           0       0.93      0.91      0.92      3090
           1       0.59      0.68      0.63       626
    accuracy                           0.87      3716
   macro avg       0.76      0.79      0.78      3716
weighted avg       0.88      0.87      0.87      3716

AUC-ROC = 0.9024507067009937

=================================================

Accuracy_Train 0.9439751354797578
Accuracy_Test 0.8684068891280947
```



Model Comparison for balanced data

Models with mutual information Features, smote and balanced dataset:

	Model	Dataset	Resample	Precision	Recall	f1-score	Accuracy	AUC-ROC
0	Logistic Regression	train	smote	0.717494	0.748486	0.732662	0.726889	0.797013
1	Logistic Regression	test	smote	0.360879	0.760383	0.489460	0.732777	0.809160
2	Naive Bayes	train	smote	0.632524	0.797259	0.705401	0.667039	0.778417
3	Naive Bayes	test	smote	0.269798	0.789137	0.402116	0.604682	0.786561
4	Stochastic Gradient Descent	train	smote	0.657392	0.828499	0.733094	0.698358	0.795431
5	Stochastic Gradient Descent	test	smote	0.298482	0.848243	0.441580	0.638590	0.807913
6	K-Nearest Neighbours	train	smote	0.900910	0.962225	0.930559	0.928196	0.985699
7	K-Nearest Neighbours	test	smote	0.487179	0.758786	0.593379	0.824812	0.870042
8	Decision Tree	train	smote	0.852155	0.879184	0.865459	0.863325	0.948436
9	Decision Tree	test	smote	0.418347	0.662939	0.512979	0.787944	0.823090
10	Random Forest	train	smote	0.933136	0.956487	0.944667	0.943975	0.989870
11	Random Forest	test	smote	0.590969	0.710863	0.645395	0.868407	0.902451

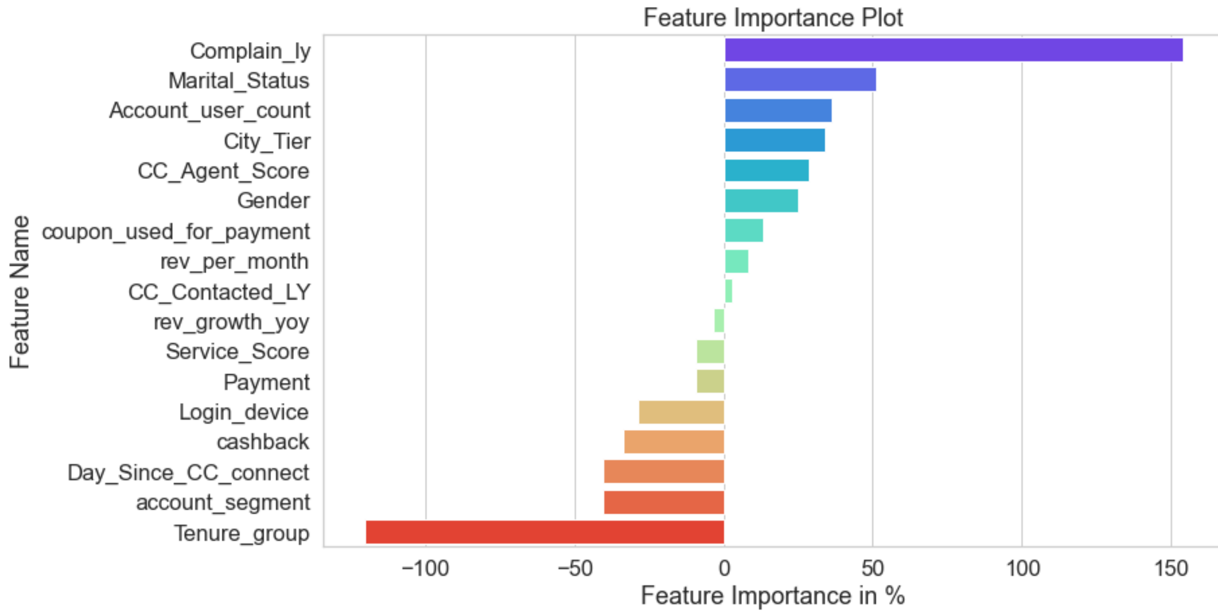
Model Building – with individual model feature selection and Imbalanced data

Model-1: Logistic Regression

```
LogisticRegression(C=1048.1131341546863, class_weight='balanced')
```

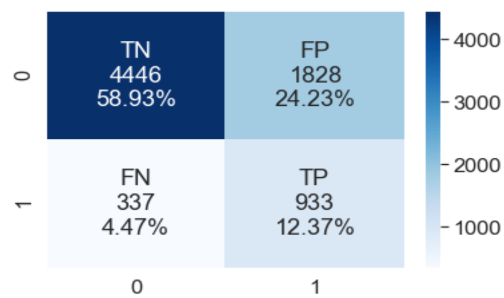
Feature Importance

```
                              Imp
Complain_ly              1.540296
Marital_Status           0.513472
Account_user_count       0.362529
City_Tier                0.341649
CC_Agent_Score           0.286120
Gender                   0.252069
coupon_used_for_payment  0.133674
rev_per_month            0.084379
CC_Contacted_LY          0.028846
rev_growth_yoy          -0.034260
Service_Score           -0.092014
Payment                 -0.092837
Login_device            -0.284675
cashback                -0.335583
Day_Since_CC_connect    -0.401111
account_segment         -0.403785
Tenure_group            -1.202421
```



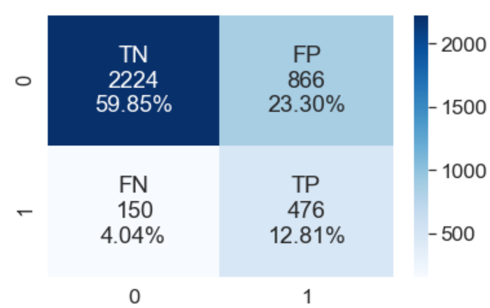

```
Classification Report for Train dataset
=======================================================

              precision    recall  f1-score   support

           0       0.93      0.71      0.80      6274
           1       0.34      0.73      0.46      1270
    accuracy                           0.71      7544
   macro avg       0.63      0.72      0.63      7544
weighted avg       0.83      0.71      0.75      7544

AUC-ROC = 0.7977883980632482

=======================================================
```

Confusion Matrix




```
Classification Report for Test dataset
=====================================================

              precision    recall  f1-score   support

           0       0.95      0.72      0.81      3090
           1       0.35      0.76      0.48       626
    accuracy                           0.73      3716
   macro avg       0.65      0.74      0.65      3716
weighted avg       0.84      0.73      0.76      3716

AUC-ROC = 0.8128979393488218

=====================================================
```

Confusion Matrix



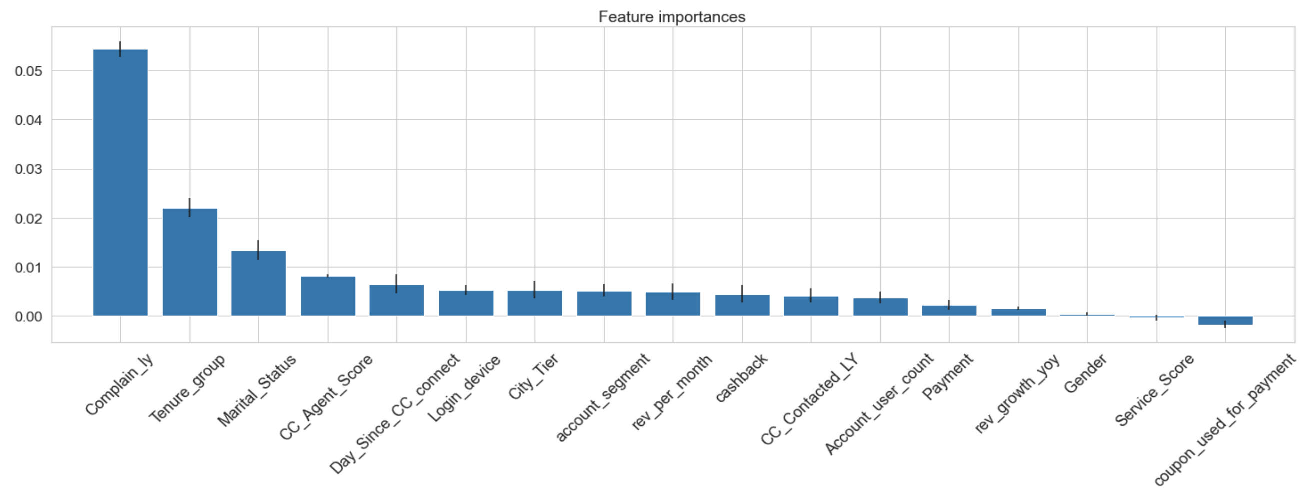

```
Accuracy_Train 0.713016967126193
Accuracy_Test 0.7265877287405813
```

Model-2: Naïve Bayes

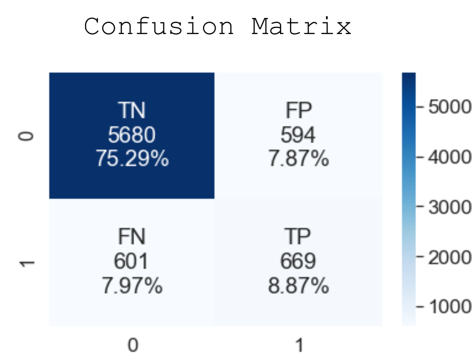
```
GaussianNB()
```

Feature Importance

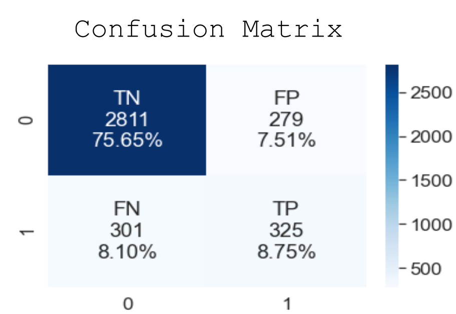
```
Feature ranking:
1. Complain_ly (0.054374)
2. Tenure_group (0.022004)
3. Marital_Status (0.013441)
4. CC_Agent_Score (0.008165)
5. Day_Since_CC_connect (0.006522)
6. Login_device (0.005329)
7. City_Tier (0.005329)
8. account_segment (0.005170)
```




```
Classification Report for Train dataset
=====================================================

              precision    recall  f1-score   support
           0       0.90      0.91      0.90      6274
           1       0.53      0.53      0.53      1270
    accuracy                           0.84      7544
   macro avg       0.72      0.72      0.72      7544
weighted avg       0.84      0.84      0.84      7544

AUC-ROC = 0.7973422373048125
=====================================================
```




```
Classification Report for Test dataset
=====================================================

              precision    recall  f1-score   support
           0       0.90      0.91      0.91      3090
           1       0.54      0.52      0.53       626
    accuracy                           0.84      3716
   macro avg       0.72      0.71      0.72      3716
weighted avg       0.84      0.84      0.84      3716

AUC-ROC = 0.807516258775603
```

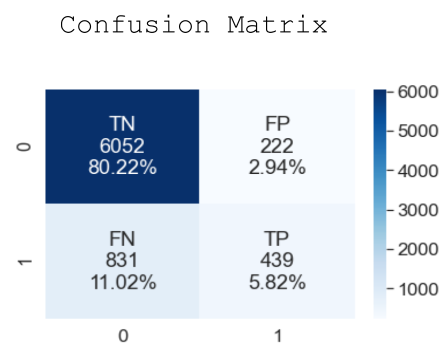


```
Accuracy_Train 0.8415959703075292
Accuracy_Test 0.8439181916038752
```

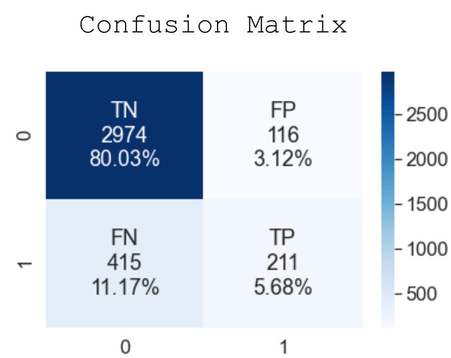
Model-3: Stochastic Gradient Descent

```
SGDClassifier(alpha=0.0005, loss='modified_huber', penalty='l1',
              random_state=123)
```

```
Classification Report for Train dataset
====================================================

              precision    recall  f1-score   support
           0       0.88      0.96      0.92      6274
           1       0.66      0.35      0.45      1270
    accuracy                           0.86      7544
   macro avg       0.77      0.66      0.69      7544
weighted avg       0.84      0.86      0.84      7544

AUC-ROC = 0.796020572340794
====================================================
```




```
Classification Report for Test dataset
====================================================

             precision    recall  f1-score   support
           0       0.88      0.96      0.92      3090
           1       0.65      0.34      0.44       626
    accuracy                           0.86      3716
   macro avg       0.76      0.65      0.68      3716
weighted avg       0.84      0.86      0.84      3716

AUC-ROC = 0.8112232079158782

===================================================
```

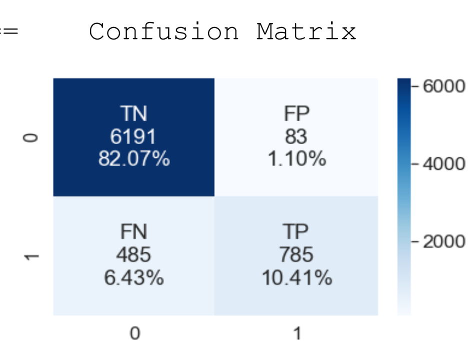



```
Accuracy_Train 0.8604188759278897
Accuracy_Test 0.8571044133476857
```

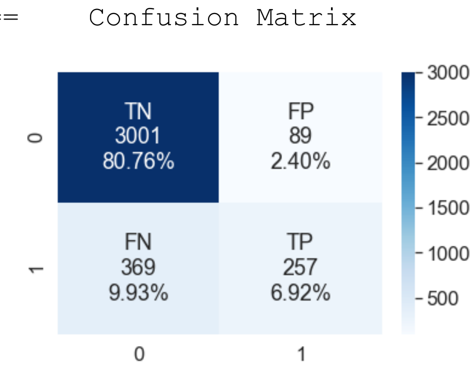
Model-4: K-Nearest Neighbours

```
KNeighborsClassifier()
```

```
Classification Report for Train dataset
=====================================================

              precision  recall  f1-score   support
           0       0.93    0.99      0.96      6274
           1       0.90    0.62      0.73      1270
    accuracy                         0.92      7544
   macro avg       0.92    0.80      0.85      7544
weighted avg       0.92    0.92      0.92      7544

AUC-ROC = 0.9648370101330577
===================================================
```



```
Classification Report for Test dataset
=======================================================

              precision    recall  f1-score   support
           0       0.89      0.97      0.93      3090
           1       0.74      0.41      0.53       626
    accuracy                           0.88      3716
   macro avg       0.82      0.69      0.73      3716
weighted avg       0.87      0.88      0.86      3716

AUC-ROC = 0.8554705997911431

=====================================================
```



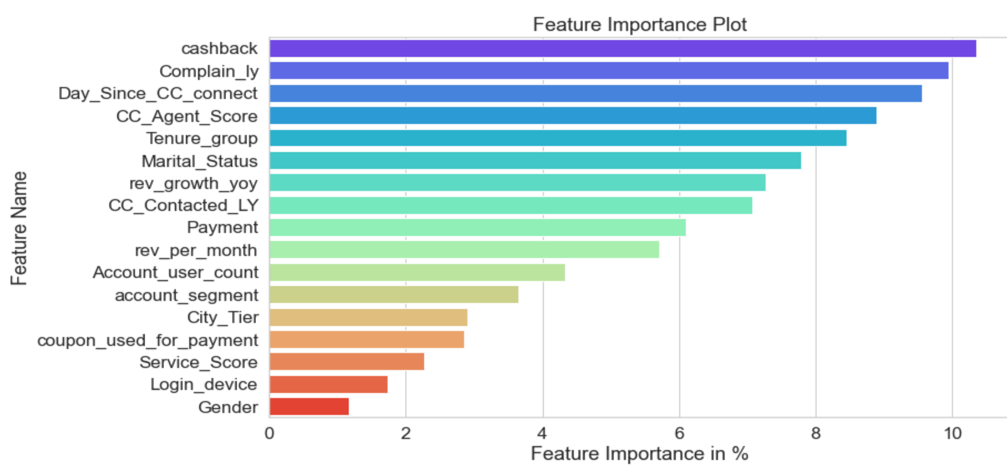

```
Accuracy_Train 0.9247083775185578
Accuracy_Test 0.8767491926803014
```

Model-5: Decision Tree

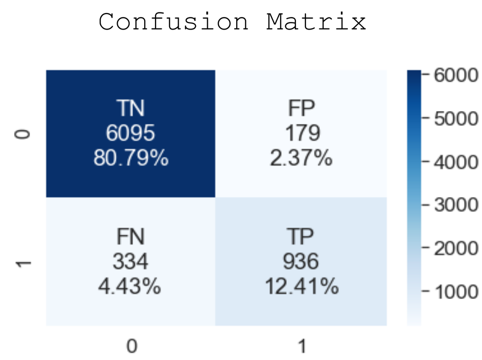
```
DecisionTreeClassifier(max_depth=23, max_features='sqrt', min_samples_leaf=4,
                       min_samples_split=10)
```

Feature Importance

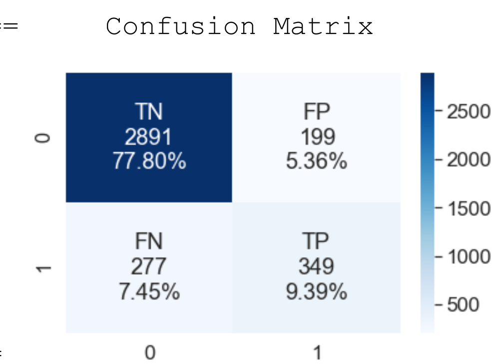
	Imp
cashback	0.103494
Complain_ly	0.099497
Day_Since_CC_connect	0.095586
CC_Agent_Score	0.088833
Tenure_group	0.084474
Marital_Status	0.077873
rev_growth_yoy	0.072608
CC_Contacted_LY	0.070685
Payment	0.060948
rev_per_month	0.057017
Account_user_count	0.043266
account_segment	0.036504
City_Tier	0.029096
coupon_used_for_payment	0.028502
Service_Score	0.022707
Login_device	0.017269
Gender	0.011640



```
Classification Report for Train dataset
==================================================

              precision    recall  f1-score   support
           0       0.95      0.97      0.96      6274
           1       0.84      0.74      0.78      1270
    accuracy                           0.93      7544
   macro avg       0.89      0.85      0.87      7544
weighted avg       0.93      0.93      0.93      7544

AUC-ROC = 0.9785947630390639

==================================================
```




```
Classification Report for Test dataset
====================================================

              precision    recall  f1-score   support
           0       0.91      0.94      0.92      3090
           1       0.64      0.56      0.59       626
    accuracy                           0.87      3716
   macro avg       0.77      0.75      0.76      3716
weighted avg       0.87      0.87      0.87      3716

AUC-ROC = 0.846337510468687

===================================================
```



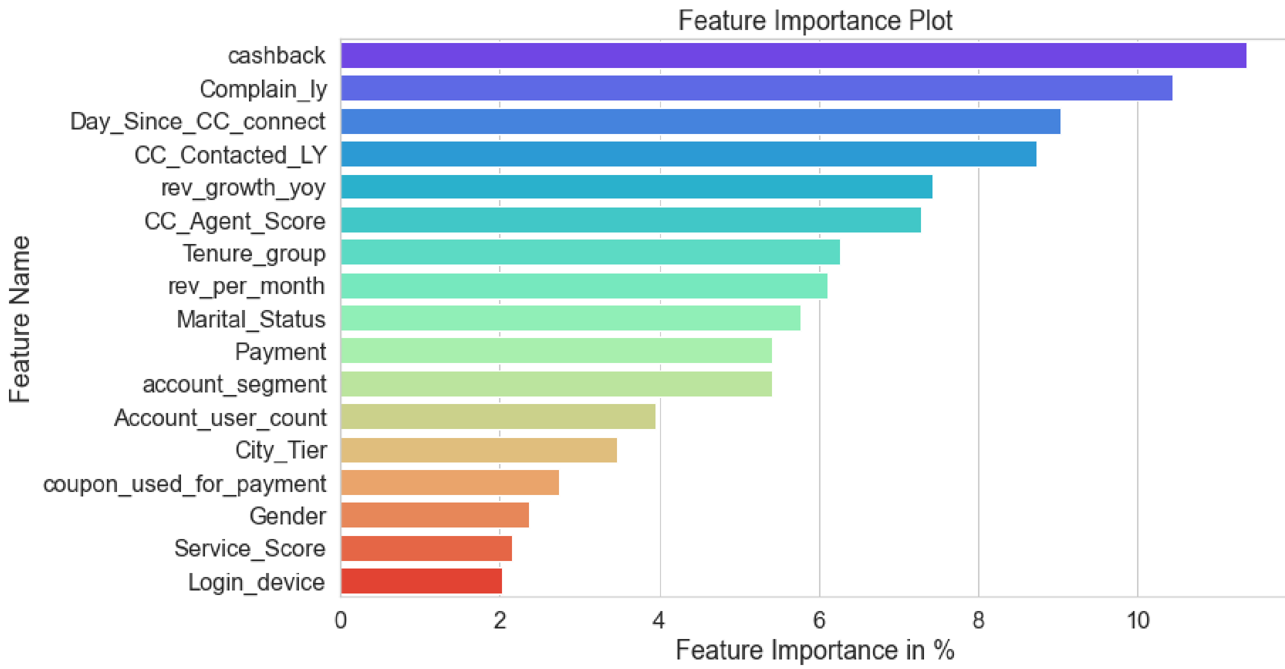

```
Accuracy_Train 0.931998939554613
Accuracy_Test 0.8719052744886975
```

Model-6: Random Forest

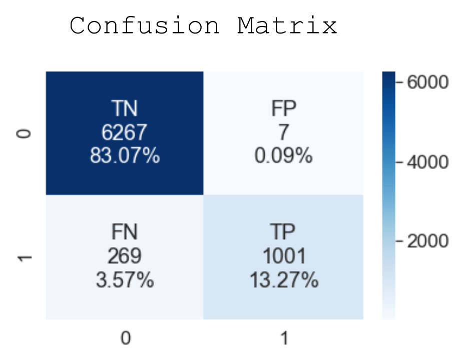
```
RandomForestClassifier(max_depth=23, max_features='log2', min_samples_leaf=4,
                       min_samples_split=5, n_estimators=500)
```

Feature Importance

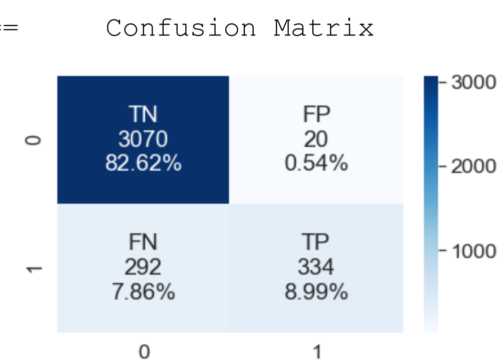
```
                             Imp
cashback                0.113712
Complain_ly             0.104388
Day_Since_CC_connect    0.090306
CC_Contacted_LY         0.087353
rev_growth_yoy          0.074238
CC_Agent_Score          0.072857
Tenure_group            0.062743
rev_per_month           0.061111
Marital_Status          0.057824
Payment                 0.054184
account_segment         0.054095
Account_user_count      0.039485
City_Tier               0.034796
coupon_used_for_payment 0.027417
Gender                  0.023625
Service_Score           0.021609
Login_device            0.020256
```




```
Classification Report for Train dataset
====================================================

              precision    recall  f1-score   support
           0       0.96      1.00      0.98      6274
           1       0.99      0.79      0.88      1270
    accuracy                           0.96      7544
   macro avg       0.98      0.89      0.93      7544
weighted avg       0.96      0.96      0.96      7544

AUC-ROC = 0.9977772283564968

====================================================
```




```
Classification Report for Test dataset
=====================================================

              precision    recall  f1-score   support
           0       0.91      0.99      0.95      3090
           1       0.94      0.53      0.68       626
     accurac                           0.92      3716
   macro avg       0.93      0.76      0.82      3716
weighted avg       0.92      0.92      0.91      3716

AUC-ROC = 0.9658824198434609

===================================================
```




```
Accuracy_Train 0.9634146341463414
Accuracy_Test 0.9160387513455328
```

Model Comparison for imbalanced data

Models with model-based Features selection and imbalanced dataset:

	Model	Dataset	Resample	Precision	Recall	f1-score	Accuracy	AUC-ROC
0	Logistic Regression	train	actual	0.337921	0.734646	0.462912	0.713017	0.797788
1	Logistic Regression	test	actual	0.354694	0.760383	0.483740	0.726588	0.812898
2	Naive Bayes	train	actual	0.529691	0.526772	0.528227	0.841596	0.797342
3	Naive Bayes	test	actual	0.538079	0.519169	0.528455	0.843918	0.807516
4	Stochastic Gradient Descent	train	actual	0.664145	0.345669	0.454687	0.860419	0.796021
5	Stochastic Gradient Descent	test	actual	0.645260	0.337061	0.442812	0.857104	0.811223
6	K-Nearest Neighbours	train	actual	0.904378	0.618110	0.734331	0.924708	0.964837
7	K-Nearest Neighbours	test	actual	0.742775	0.410543	0.528807	0.876749	0.855471
8	Decision Tree	train	actual	0.839462	0.737008	0.784906	0.931999	0.978595
9	Decision Tree	test	actual	0.636861	0.557508	0.594549	0.871905	0.846338
10	Random Forest	train	actual	0.993056	0.788189	0.878841	0.963415	0.997777
11	Random Forest	test	actual	0.943503	0.533546	0.681633	0.916039	0.965882

SMOTE applied:

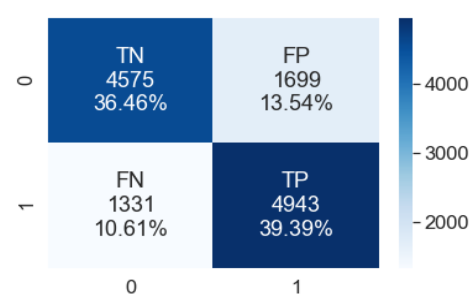
```
Before Counter({0: 6274, 1: 1270})
After Counter({0: 6274, 1: 6274})

After OverSampling, the shape of X_train: (12548, 17)
After OverSampling, the shape of y_train: (12548,)
```

Model Building - with individual model feature selection and balanced data

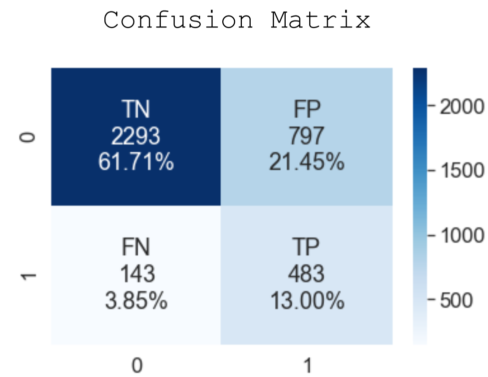
Model-1: Logistic Regression - SMOTE Resampling

```
LogisticRegression(C=100000.0)

Classification Report for Train dataset
====================================================

              precision    recall  f1-score   support

           0       0.77      0.73      0.75      6274
           1       0.74      0.79      0.77      6274
    accuracy                           0.76     12548
   macro avg       0.76      0.76      0.76     12548
weighted avg       0.76      0.76      0.76     12548

AUC-ROC = 0.8298422105020451

==================================================
```

Confusion Matrix



```
Classification Report for Test dataset
==================================================

              precision    recall  f1-score   support
           0       0.94      0.74      0.83      3090
           1       0.38      0.77      0.51       626
    accuracy                           0.75      3716
   macro avg       0.66      0.76      0.67      3716
weighted avg       0.85      0.75      0.78      3716

AUC-ROC = 0.8305411664960658

==================================================

Accuracy_Train 0.7585272553394964
Accuracy_Test 0.7470398277717977
```

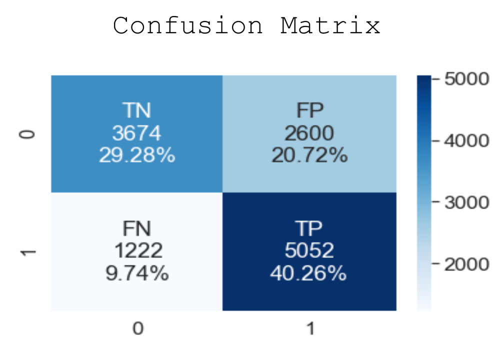


Inferences:

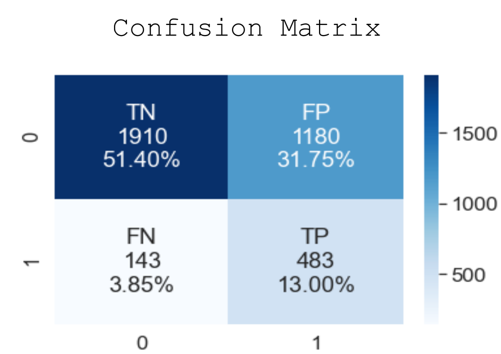
This model has a **higher rate of false positives.** Practically, it means you **will be able to eng age with 13% of the customers who will churn**, but you will miss the other 3.85%. Also, you may h ave 21.45% who are incorrectly predicted as churned.

Model-2: Naïve Bayes - SMOTE Resampling

```
GaussianNB()

Classification Report for Train dataset
=================================================

             precision    recall  f1-score   support
           0      0.75      0.59      0.66      6274
           1      0.66      0.81      0.73      6274
    accuracy                          0.70     12548
   macro avg      0.71      0.70      0.69     12548
weighted avg      0.71      0.70      0.69     12548

AUC-ROC = 0.8113571459710112

=================================================
```



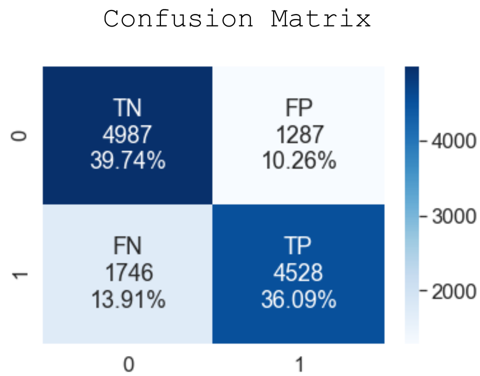

```
Classification Report
=================================================

             precision    recall  f1-score   support
           0      0.93      0.62      0.74      3090
           1      0.29      0.77      0.42       626
    accuracy                          0.64      3716
   macro avg      0.61      0.69      0.58      3716
weighted avg      0.82      0.64      0.69      3716

AUC-ROC = 0.7917568783150841

=================================================
```



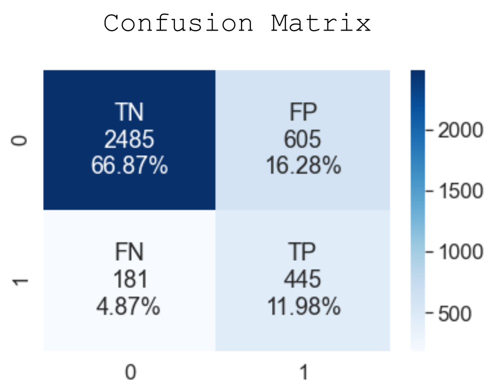
```
Accuracy_Train 0.6954096270321963
Accuracy_Test 0.6439720129171151
```

Model-3: Stochastic Gradient Descent - SMOTE Resampling

```
SGDClassifier(alpha=0.0005, loss='modified_huber', penalty='l1',
              random_state=123)

Classification Report for Train dataset
=================================================

              precision    recall  f1-score   support
           0       0.74      0.79      0.77      6274
           1       0.78      0.72      0.75      6274
    accuracy                           0.76     12548
   macro avg       0.76      0.76      0.76     12548
weighted avg       0.76      0.76      0.76     12548

AUC-ROC = 0.7585077446691412

=================================================
```



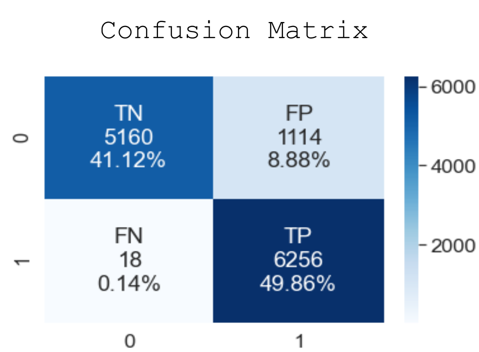

```
Classification Report for Test dataset
=================================================

              precision    recall  f1-score   support
           0       0.93      0.80      0.86      3090
           1       0.42      0.71      0.53       626
    accuracy                           0.79      3716
   macro avg       0.68      0.76      0.70      3716
weighted avg       0.85      0.79      0.81      3716

AUC-ROC = 0.7576031101047386

=================================================
```




```
Accuracy_Train 0.7582881734140899
Accuracy_Test 0.7884822389666308
```

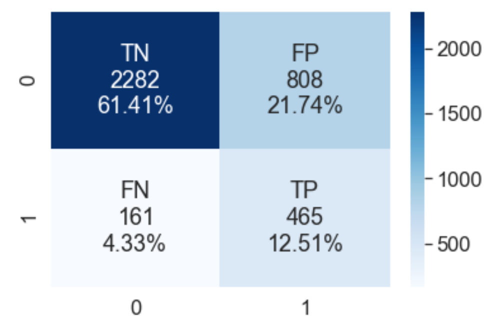
Model-4: K-Nearest Neighbours - SMOTE Resampling

```
KNeighborsClassifier()

Classification Report for Train dataset
==================================================

              precision    recall  f1-score   support
           0       1.00      0.82      0.90      6274
           1       0.85      1.00      0.92      6274
    accuracy                           0.91     12548
   macro avg       0.92      0.91      0.91     12548
weighted avg       0.92      0.91      0.91     12548

AUC-ROC = 0.9967236046288659

==================================================
```



```
Classification Report for Test dataset
==================================================

              precision    recall  f1-score   support
           0       0.93      0.74      0.82      3090
           1       0.37      0.74      0.49       626
    accuracy                           0.74      3716
   macro avg       0.65      0.74      0.66      3716
weighted avg       0.84      0.74      0.77      3716

AUC-ROC = 0.8106323086944385

==================================================

Accuracy_Train 0.9097864201466369
Accuracy_Test 0.7392357373519914
```

Confusion Matrix

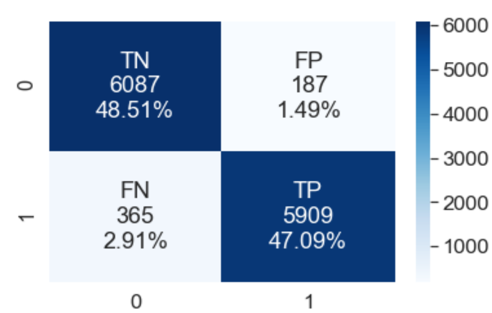


Model-5: Decision Tree - SMOTE Resampling

```
DecisionTreeClassifier(max_depth=23, max_features='sqrt', min_samples_leaf=4,
                       min_samples_split=10)
```

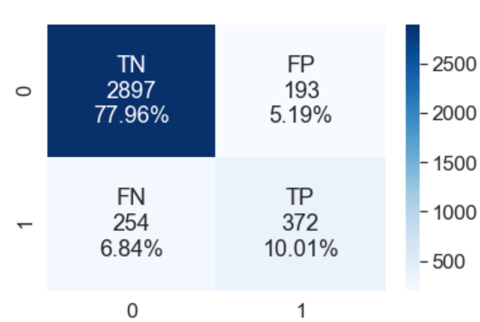
```
Classification Report for Train dataset
==================================================

              precision    recall  f1-score   support
           0       0.94      0.97      0.96      6274
           1       0.97      0.94      0.96      6274
    accuracy                           0.96     12548
   macro avg       0.96      0.96      0.96     12548
weighted avg       0.96      0.96      0.96     12548

AUC-ROC = 0.9949430146160325

==================================================
```

Confusion Matrix




```
Classification Report for Test dataset
==================================================

              precision    recall  f1-score   support
           0      0.92       0.94      0.93      3090
           1      0.66       0.59      0.62       626
    accuracy                           0.88      3716
   macro avg      0.79       0.77      0.78      3716
weighted avg      0.88       0.88      0.88      3716

AUC-ROC = 0.8683080533928886

==================================================

Accuracy_Train 0.9560089257252151
Accuracy_Test 0.8797093649085038
```

Confusion Matrix



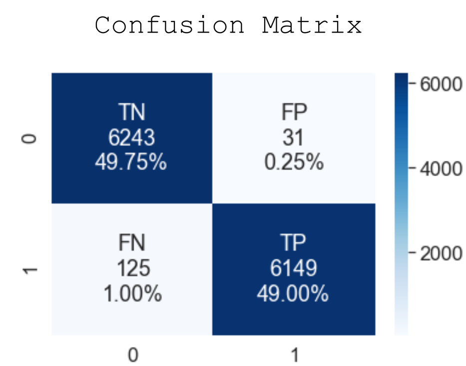
Inferences:

This model has a **smaller rate of false positives.** Practically, it means you **will be able to en gage with 10% of the customers who will churn**, but you will miss the other 6.84%. Also, you may have 5.19% who are incorrectly predicted as churned.

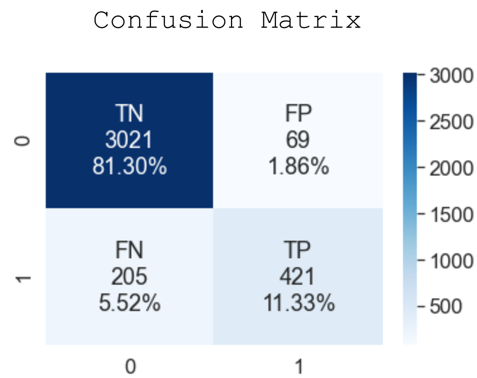
Model-6: Random Forest - SMOTE Resampling

```
RandomForestClassifier(max_depth=23, max_features='log2', min_samples_leaf=4,
                       min_samples_split=5, n_estimators=500)
```

```
Classification Report for Train dataset
================================================

              precision    recall  f1-score   support
           0       0.98      1.00      0.99      6274
           1       0.99      0.98      0.99      6274
    accuracy                           0.99     12548
   macro avg       0.99      0.99      0.99     12548
weighted avg       0.99      0.99      0.99     12548

AUC-ROC = 0.9994837293711498

================================================
```




```
Classification Report for Test dataset
================================================

              precision    recall  f1-score   support
           0       0.94      0.98      0.96      3090
           1       0.86      0.67      0.75       626
    accuracy                           0.93      3716
   macro avg       0.90      0.83      0.86      3716
weighted avg       0.92      0.93      0.92      3716

AUC-ROC = 0.9635462224841549

================================================

Accuracy_Train 0.9875677398788651
Accuracy_Test 0.926264800861141
```



Inferences:

Random Forest model has a **smaller rate of false positives.** Practically, it means you **will be able to engage with 11.33% of the customers who will churn**, but you will miss the other 5.52%. A lso, you may have 1.86% who are incorrectly predicted as churned.

Model Comparison for balanced data

Models with model-based Feature selection, smote - balanced dataset:

	Model	Dataset	Resample	Precision	Recall	f1-score	Accuracy	AUC-ROC
0	Logistic Regression	train	smote	0.744204	0.787855	0.765407	0.758527	0.829842
1	Logistic Regression	test	smote	0.377344	0.771565	0.506821	0.747040	0.830541
2	Naive Bayes	train	smote	0.660220	0.805228	0.725549	0.695410	0.811357
3	Naive Bayes	test	smote	0.290439	0.771565	0.422018	0.643972	0.791757
4	Stochastic Gradient Descent	train	smote	0.778676	0.721709	0.749111	0.758288	0.758508
5	Stochastic Gradient Descent	test	smote	0.423810	0.710863	0.531026	0.788482	0.757603
6	K-Nearest Neighbours	train	smote	0.848847	0.997131	0.917033	0.909786	0.996724
7	K-Nearest Neighbours	test	smote	0.365279	0.742812	0.489731	0.739236	0.810632
8	Decision Tree	train	smote	0.969324	0.941823	0.955376	0.956009	0.994943
9	Decision Tree	test	smote	0.658407	0.594249	0.624685	0.879709	0.868308
10	Random Forest	train	smote	0.994984	0.980077	0.987474	0.987568	0.999484
11	Random Forest	test	smote	0.859184	0.672524	0.754480	0.926265	0.963546

Inferences:

Always models with the balanced dataset performs better than imbalanced data. The Random Forest model with all features and the **SVM Model** performs well. Nonetheless, it has a smaller rate of **false positives**. Practically, it means you will be able to engage with **14.32%** of the customers who will churn, but you will miss the other **2.53%.** Also, you may have **6.84%** who are incorrectly predicted as churned.

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

- In this project, I have tried to divide customer churn prediction problem into steps like exploration, profiling, clustering, model selection & evaluation. Based on this analysis, we can help retention team to analyse high risk churn customers before they leave the company.
- Moreover, we can add on different data sources like customer inquiries, seasonality in sales, more demographic information to make our prediction more accurate.
- From the results and explanations presented here, some conclusion can be draw:

The type of account segment has a strict relationship with churned clients, Low Tenure with high complaints could lead a client to leave the service. Clients with a greater number of account_user_count tend to leave.

We can now see that the main factors are.

1. **Customer Demography:**

    ▪ Tenure

        - Longtime customers are less likely to leave the company.
        - Loyalty

    ▪ Gender

        - Male customers tend to churn more.
        - Attrack them with some sports, games, and discovery type channels combo packages.

2. **Customer record analysis:**

    ▪ Account_segment

        - People having Regular Plus are more likely to leave.
        - Is there something wrong with the Regular Plus segment?

3. **Customer care service analysis:**

    ▪ Complain_ly

        - People having more complaints are more likely to leave.
        - Are customers unhappy with the solution given by customer service.

    ▪ Service score

        - Low score given by customers tend to churn.
        - Resolve their problems by giving them good offers like cashback or free channels for one month.

9. Business Implications:

Depending on the re-engagement campaign, it can be a good trade-off to target the highest possible number of customers at risk to churn, and in parallel unintentionally reach some happy customers, than to leave a high number of customers to cancel without taking proper actions.

It is generally thought to cost five times more to gain a new customer than it costs to keep an existing customer, and from research it shows that boosting your customer retention rate by 5% leads to a profit increase of 25% to 95%.

This makes intuitive sense if you think about all the steps involved — there's a high associated acquisition cost with acquiring and educating a new customer. You must find a customer, learn their needs, position your product, onboard the new client, and then wow them to stay in the first critical months. That is a lot of steps…

Wouldn't it be nicer to keep the customers you already have than work twice as hard on the acquisition front? I think so…

10. Business Recommendations:

- Complaint redressal needs to be a major focus point as a large number of customer attrition is attributed to complaints. To address this, a dedicated customer service team needs to be formed that is trained to be sensitive about complaints and trained to appropriately handle such complaints. Customers should have the option to directly contact this team for faster resolution, through dedicated IVR (Interactive Voice Response) options or dedicated chat links on the company's website.
- The fact that customers who have recently contacted the CC have attritted more than customers who have not contacted, points to the fact that these customers were unhappy with the way they were dealt with. To address this, the CC staff needs to be trained on technical and soft skills to enable them to provide outstanding customer service. A dedicated training team working in sync with the supervisors would be effective in understanding where the staff is lacking and then accordingly train them to be more effective. This will enhance the customer experience and would greatly help in regaining the customer's trust.
- Tier 2 & 3 customers are more likely to attrite. A survey comprising of both close-ended and open-ended questions would help in understanding the factors leading to tier 2 and tier 3 attritions.
- Monitor the issues raised by the Tier 2 and Tier 3 customers to understand the major areas of improvement.

11. Conclusion:

No algorithm will predict churn with 100% accuracy. There will always be a trade-off between precision and recall. That is why it is important to test and understand the strengths and weaknesses of each classifier and get the best out of each. If the goal is to engage and reach out to the customers to prevent them from churning, it is acceptable to engage with those who are mistakenly tagged as ‘not churned,’ as it does not cause any negative impact. It could potentially make them even happier with the service. This is the kind of model that can add value from day one if proper action is taken out of meaningful information it produces.

END