CUSTOMER CHURN PREDICTION

Capstone Project – Final report

Abstract

This is a binary classification problem to predict customers who would churn

Kalaivani.K.G

DSBA Dec-20 batch

Contents

List of tables

List of figures

1. Introduction - Business problem definition

1.1. Problem statement

A DTH provider is facing a lot of competition in the current market and it has become a challenge to retain the existing customers in the current situation. Hence, the company wants to develop a model through which they can do churn prediction of the accounts and provide segmented offers to potential churners. 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.

The current project is aimed at developing a churn prediction model for this company and to provide business recommendations on a campaign focused on retaining customers. The campaign recommendation should be such that it does not entail a huge cost for retention of customers and should remain within a budget earmarked for this purpose.

1.2. Need for the project

A DTH provider's biggest cost is cost of acquisition of a new customer. A customer thus acquired, will need to be retained for quite a few years so that the initial cost of acquisition is recovered back and that particular account is profitable[1]. Due to this reason, customer churn directly impacts the profitability of a DTH operator. DTH providers also are in a constant pressure to increase their customer base to maintain their profitability as most of them have a fixed broadcaster/content provider fee irrespective of the number of customers in their customer base[2]. So, more the number of customers, greater their profitability. Hence it becomes very important to not only increase the customer base but also protect the current customer base.

Acquiring a new customer can cost five times more than retaining an existing customer. Increasing customer retention by 5% can increase profits from 25-95%.[3]

As customer churn directly impacts both the top-line and bottom-line revenue of the business, existing customer base needs to be protected. Providing all customers with offers to retain them would make a dent in the profitability and hence it is very important to focus only on select set of customers who are at a higher risk of churning.

1.3. Project Objectives

1) This project aims at identifying customers who have a propensity to churn so that targeted campaign offers can be provided to them. This would ensure better top-line and bottom-line revenue for the business.
2) Other insights from Exploratory data analysis and best performing model(s) will also be used to come up with business recommendations.

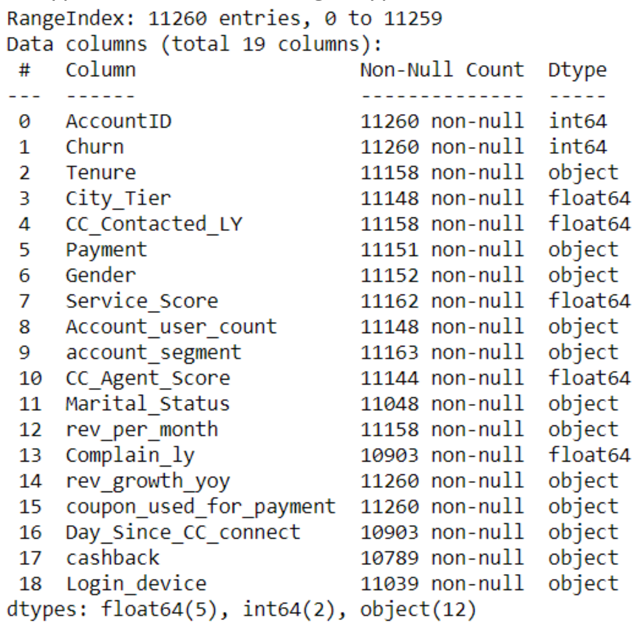
2. Data report

2.1. Data collection

- The dataset contains 'Account' level data which is master data along with features of account such as gender, marital status, city tier, account user count of primary account holder, whether the account is live or churned, segment the account belongs to. It also possibly contains certain derived features such as tenure (which probably could have been derived from account open date)
- The dataset also contains information taken/derived from transaction data and rolled up at Account level and given as a feature for the account – for e.g., Number of days since none of the account holders contacted customer care, monthly average cashback for last 12 months, number of complaints made last year, number of times customer care contacted last year, revenue per month in last 12 months, how many times customers have used coupons to pay in last 12 months, satisfaction score and customer service score.
- Most of the transaction data roll-up at account level has been done for 12 months (previous year). However, the revenue growth percentage has been taken for last year in comparison with previous one year which implies that 24-months' worth of data has been used to calculate this field.
- As the dataset has been provided, the methodology used by customer to extract data is not known. The frequency at which this dataset is extracted has also not been specified.

2.2. Visual inspection of data

- The dataset has 11260 rows and 19 columns.
- There are 5 columns of float type, 2 columns of integer type and 12 columns of object type



```
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)
```

Table 2-1 Columns and data types

- There are several columns that are supposed to be read as numeric, instead they have been read as object type for e.g., Tenure is a numeric field but has been read as object. Those columns need to be checked for special characters and need to be treated before the column can be changed to numeric for further processing.
- There are no duplicate rows in the data set. Each account id has one unique row.
- Several columns have null values
- The following table shows number of rows containing nulls and special characters that require data cleaning. All special characters present in the data set were treated with nulls so that they can be imputed. Some columns such as Gender and account_segment contained multiple values to represent the same category for e.g., 'M', 'Male'. Cleaning up of those values was also performed.

Column	Values present	% Rows with values present	Number of Nulls	% Rows with nulls	Data clean-up needed?	% Rows needing data cleaning
AccountID	11260	100.00%	0	0%	None	0.00%
Churn	11260	100.00%	0	0%	None	0.00%
Tenure	11158	99.09%	102	0.91%	Yes - #	1.03%
City_Tier	11148	99.01%	112	0.99%	None	0.00%
CC_Contacted_LY	11158	99.09%	102	0.91%	None	0.00%
Payment	11151	99.03%	109	0.97%	None	0.00%
Gender	11152	99.04%	108	0.96%	Yes - M,F	5.74%
Service_Score	11162	99.13%	98	0.87%	None	0.00%
Account_user_count	11148	99.01%	112	0.99%	Yes - @	2.95%
account_segment	11163	99.14%	97	0.86%	Yes-Regular + Super +	2.74%
CC_Agent_Score	11144	98.97%	116	1.03%	None	0.00%
Marital_Status	11048	98.12%	212	1.88%	None	0.00%
rev_per_month	11158	99.09%	102	0.91%	Yes - +	6.12%
Complain_ly	10903	96.83%	357	3.17%	None	0.00%
rev_growth_yoy	11260	100.00%	0	0.00%	Yes - $	0.03%
coupon_used_for_payment	11260	100.00%	0	0.00%	Yes - $, *, #	0.03%
Day_Since_CC_connect	10903	96.83%	357	3.17%	Yes - $	0.01%
Cashback	10789	95.82%	471	4.18%	Yes - $	0.02%
Login_device	11039	98.04%	221	1.96%	Yes - &&&&	4.79%

Table 2-2 Nulls and special characters in dataset

2.3. Understanding the attributes

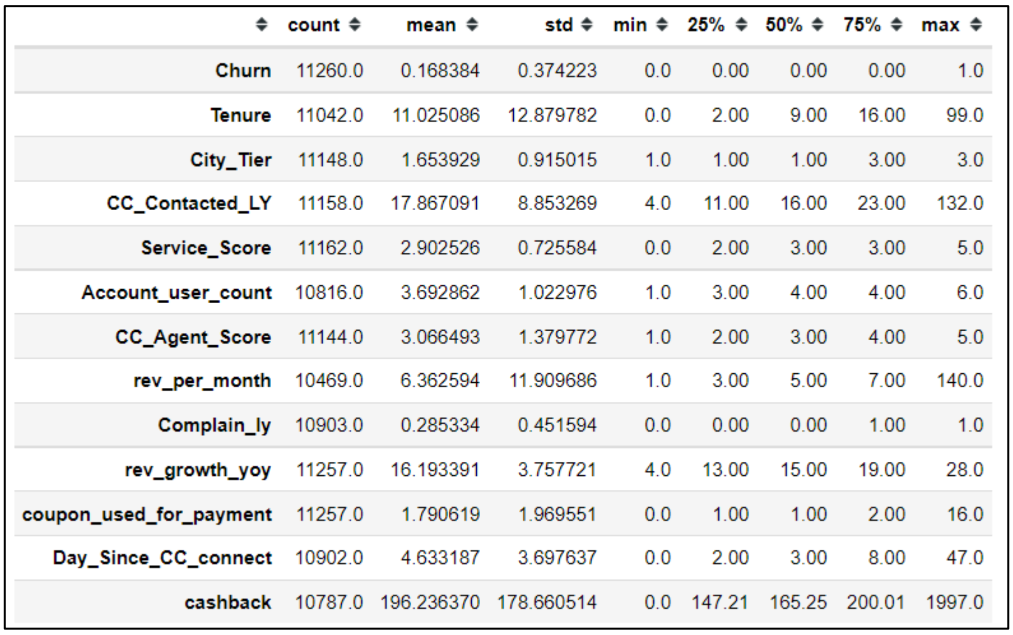
The following table shows the attribute names, their description and the kind of values that they contain. Although some of the variable names are slightly long, they do not have blanks or special characters in them. Hence, it has been decided to let the current column names stay as-is as they are self-explanatory and would be easy to understand and interpret when seen in the plots as part of univariate and bivariate analysis. The variable names would be changed later to shorten or make it uniform when one hot encoding is done in a later section.

S.no	Column	Column Description	Data description
1	AccountID	account unique identifier	Unique ID. Hence, it will not be used in modelling
2	Churn	account churn flag (Target)	Target variable. Contains 1 for churned and 0 for non-churned
3	Tenure	Tenure of account	Continuous field. Contains values ranging from 0 to 99
4	City_Tier	Tier of primary customer's city	Categorical ordinal - values 1,2,3
5	CC_Contacted_LY	How many times all the customers of the account have contacted customer care in last 12months	Continuous field. Contains values ranging from 4 to 132
6	Payment	Preferred Payment mode of the customers in the account	Categorical nominal - values Credit card, debit card, E wallet, UPI, Cash on Delivery

S.no	Column	Column Description	Data description
7	Gender	Gender of the primary customer	Categorical nominal - values Male, Female, M and F (M and F need to be converted to Male and Female)
8	Service_Score	Satisfaction score given by customers of the account on service provided by company	Categorical ordinal - values 0 to 5
9	Account_user_count	Number of customers tagged with this account	Limited range. Can be treated as categorical - values 1 to 6
10	account_segment	Account segmentation on the basis of spend	Categorical nominal - values HNI, Regular, Regular Plus, Super, Super plus and variations with +
11	CC_Agent_Score	Satisfaction score given on customer care service provided	Categorical ordinal - values 1 to 5
12	Marital_Status	Marital status of primary customer	Categorical nominal - contains values Married, Single and Divorced
13	rev_per_month	Monthly average revenue from account in last 12 months	Continuous field. Contains values ranging from 1 to 140
14	Complain_ly	Complaints raised by account in last 12 months	Categorical - 0 (for no) or 1 (for yes)
15	rev_growth_yoy	revenue growth percentage of the account (last 12 months vs last 24 to 13 month)	Continuous field. Contains values ranging from 4 to 28
16	coupon_used_for_payment	How many times customers have used coupons to do the payment in last 12 months	Continuous field, but with limited range. Contains values ranging from 0 to 16
17	Day_Since_CC_connect	Number of days since no customers in the account has contacted the customer care	Continuous field. Contains values ranging from 0 to 47
18	Cashback	Monthly average cashback generated by account in last 12 months	Continuous field. Contains values ranging from 0 to 1997
19	Login_device	Preferred login device of the customers in the account	Categorical nominal - contains values Mobile, Computer

Table 2-3 Attribute description

The following table shows a basic statistical description of the numeric columns after data clean-up. It contains a 5-point summary of the numeric fields – minimum, maximum, 25th percentile, 50th percentile and 75th percentile. In addition, it contains count of values present in each column, mean and standard deviation.



	count	mean	std	min	25%	50%	75%	max
Churn	11260.0	0.168384	0.374223	0.0	0.00	0.00	0.00	1.0
Tenure	11042.0	11.025086	12.879782	0.0	2.00	9.00	16.00	99.0
City_Tier	11148.0	1.653929	0.915015	1.0	1.00	1.00	3.00	3.0
CC_Contacted_LY	11158.0	17.867091	8.853269	4.0	11.00	16.00	23.00	132.0
Service_Score	11162.0	2.902526	0.725584	0.0	2.00	3.00	3.00	5.0
Account_user_count	10816.0	3.692862	1.022976	1.0	3.00	4.00	4.00	6.0
CC_Agent_Score	11144.0	3.066493	1.379772	1.0	2.00	3.00	4.00	5.0
rev_per_month	10469.0	6.362594	11.909686	1.0	3.00	5.00	7.00	140.0
Complain_ly	10903.0	0.285334	0.451594	0.0	0.00	0.00	1.00	1.0
rev_growth_yoy	11257.0	16.193391	3.757721	4.0	13.00	15.00	19.00	28.0
coupon_used_for_payment	11257.0	1.790619	1.969551	0.0	1.00	1.00	2.00	16.0
Day_Since_CC_connect	10902.0	4.633187	3.697637	0.0	2.00	3.00	8.00	47.0
cashback	10787.0	196.236370	178.660514	0.0	147.21	165.25	200.01	1997.0

Table 2-4 Basic descriptive statistics

- Tenure seems to have a very huge range up to 99. It could be in months in which case those would be valid values.
- The maximum limit for many of the variables seems to be very far apart from the 75th percentile for many variables such as cashback, revenue per month and customer contacted last year. There seems to be significant positive skew in these variables. A look at the boxplot and histogram will confirm the presence of outliers.

3. Exploratory data analysis

3.1. Univariate analysis

Univariate analysis is done for the purpose of observing distribution and spread for every continuous attribute and distribution of data in categories for categorical ones. It has been done by observing:

- Box plots and histograms for continuous variables
- Count plots for categorical variables

Continuous data – Box plot

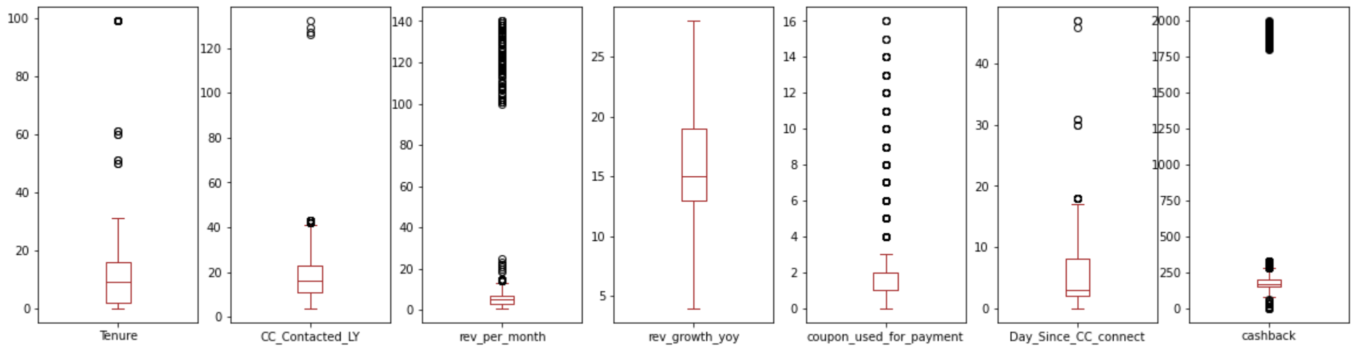
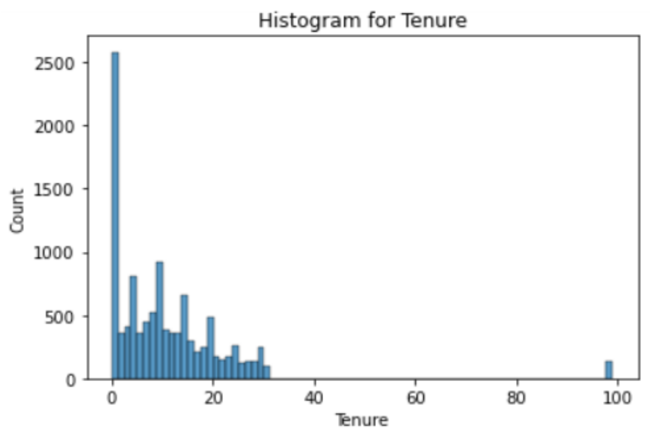
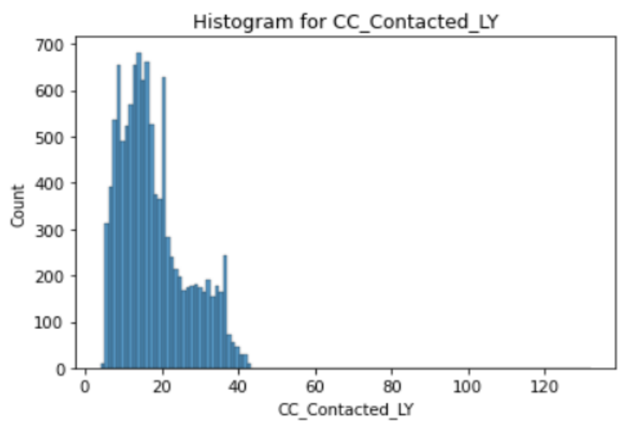
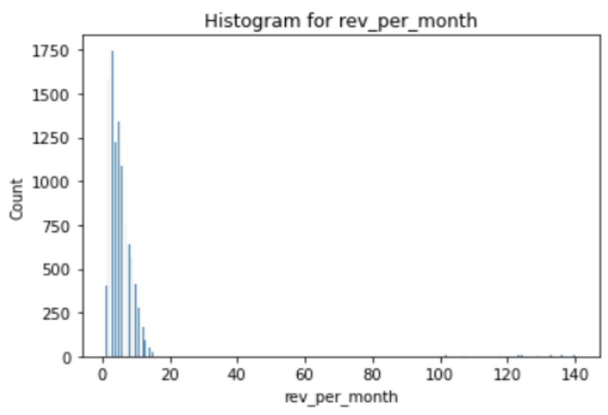


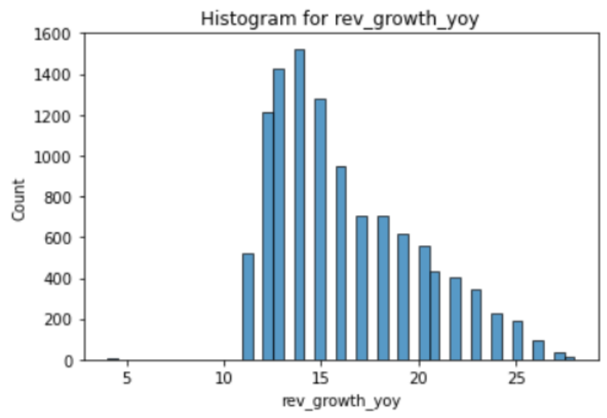
Figure 3-1 Box plot for numeric variables

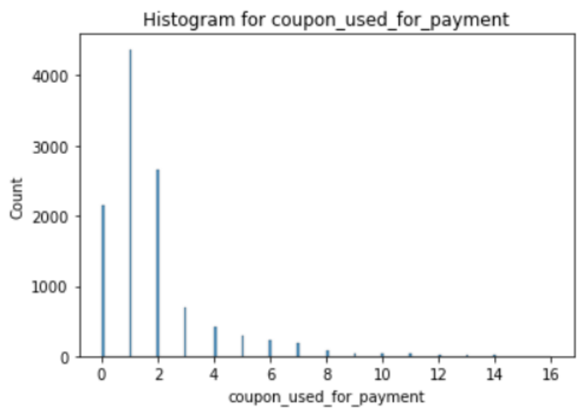
Continuous data – Histogram

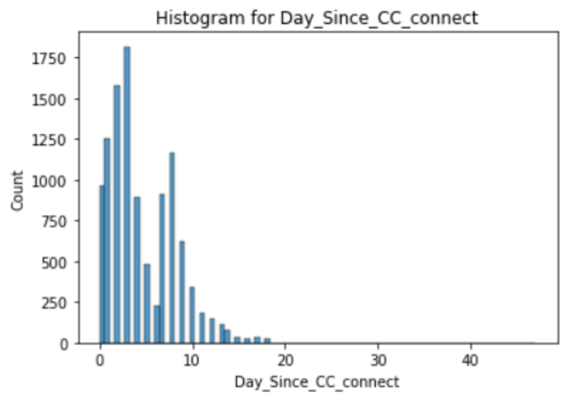


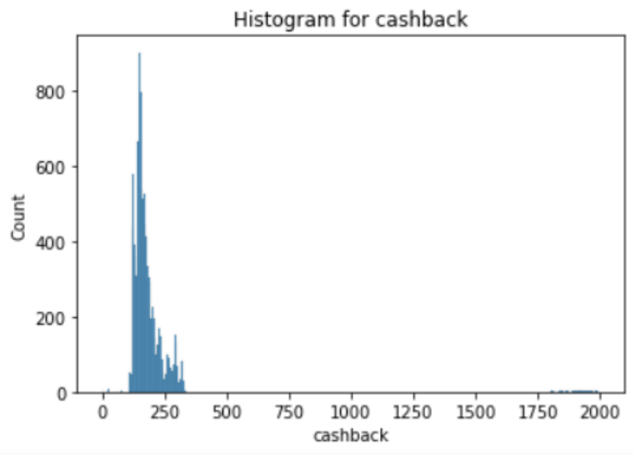












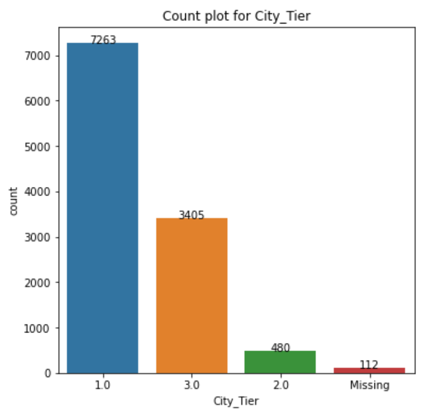
	Skewness
rev_growth_yoy	0.75
Day_Since_CC_connect	1.27
CC_Contacted_LY	1.42
coupon_used_for_payment	2.58
Tenure	3.90
cashback	8.77
rev_per_month	9.09

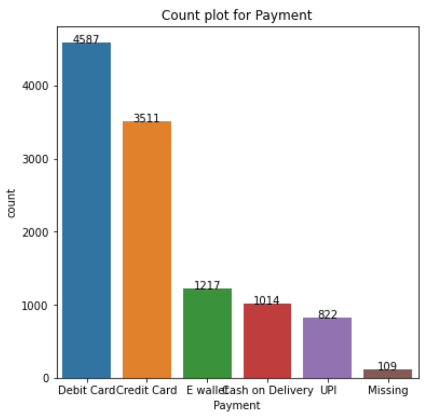
Table 3-1 Skewness of numeric variables

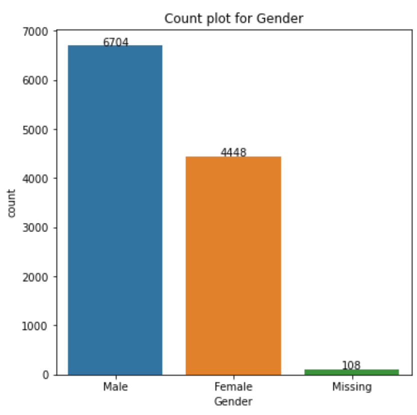
Observations:

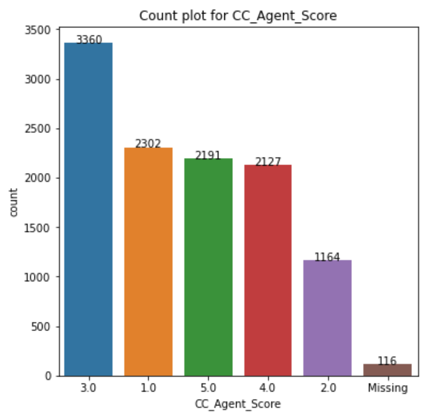
- All numeric variables with the exception of rev_growth_yoy have outliers. Some outliers for certain variables are closer to the whisker, whereas there are a group of outliers that are far beyond the whisker with no in between values. For instance, rev_per_month has a huge space between 30 and 100 indicating absence of values in that range. Those outliers in the extreme values do not correlate with corresponding outliers in cashback field. We cannot rule out these outliers as incorrect values, they may belong to hotels with many rooms. But models like logistic regression are sensitive to outliers and may not give good performance if outliers are left untreated.
- Hence, we will follow two approaches to modelling – one set of data with outliers treated for outlier sensitive models and other set of data with outliers not treated (left as-is) for outlier resistant models such as Random forest.
- Coupon_used_for_payment has a very limited range 0 to 16. Hence, for the purpose of this analysis, the outliers will not be treated (similar to categorical variables).
- All numeric variables with the exception of rev_growth_yoy have a high positive skew.

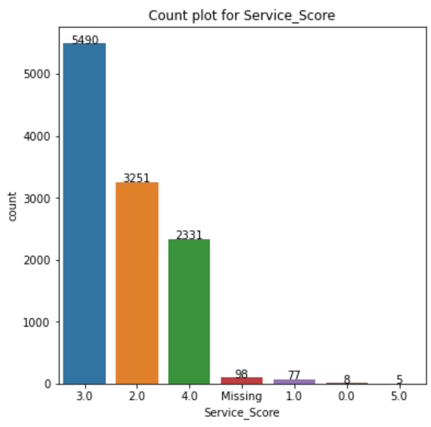
Categorical fields – Count plot

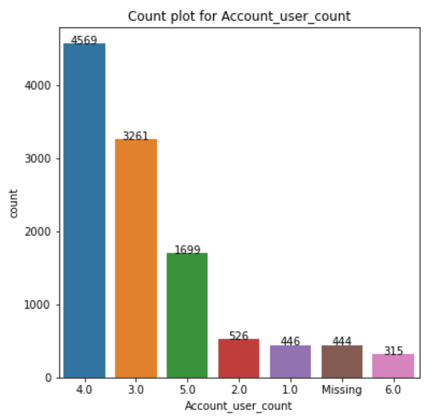


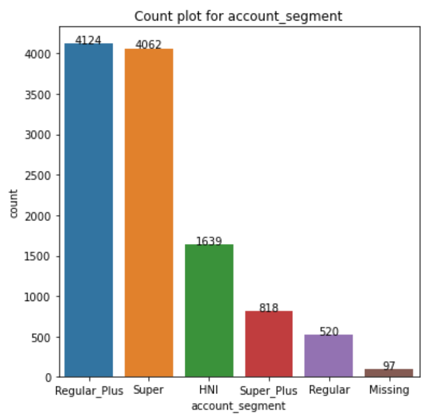


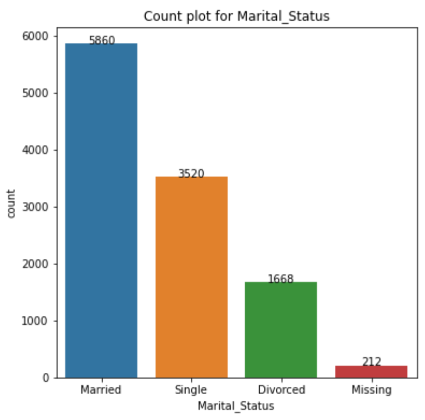


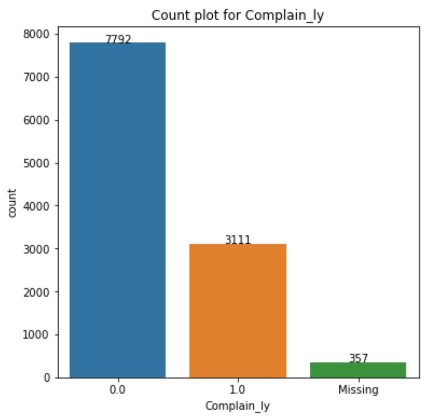


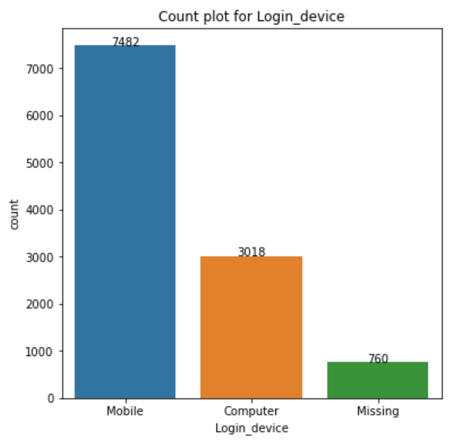


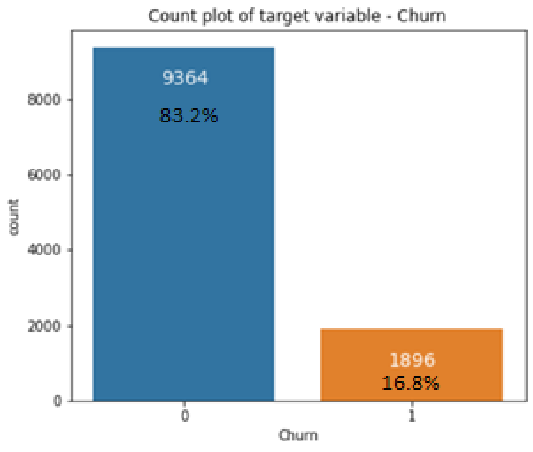












Observation: This is an imbalanced dataset with target variable containing 16.8% churn. The prediction will be for minority class.

Observations:

- This is an imbalanced dataset with target variable containing 16.8% churn
- Tier 1 cities have more accounts followed by Tier 3 cities
- Most of the accounts pay through debit card followed by credit card. UPI ranks last amongst payment methods
- Number of male account holders outnumber females
- Regular plus and Super are top two account segment types by number
- Top score for both Customer service agent and Service score is 3
- Married customers have the most accounts followed by single
- Most accounts do not have a customer complaint filed last year
- Most account holders use Mobile for logging and using services

3.2. Data imbalance

As can be seen from the plot below, the data is imbalanced with respect to the target variable which is the indicator whether customer has churned or not. The distribution given below shows that for every 100 customers acquired by the business, 17 have churned and 83 customers are active. This distribution is skewed towards active/current customers. The objective for this exercise is to be able to predict the customers who would churn i.e., the minority class ‘1’.

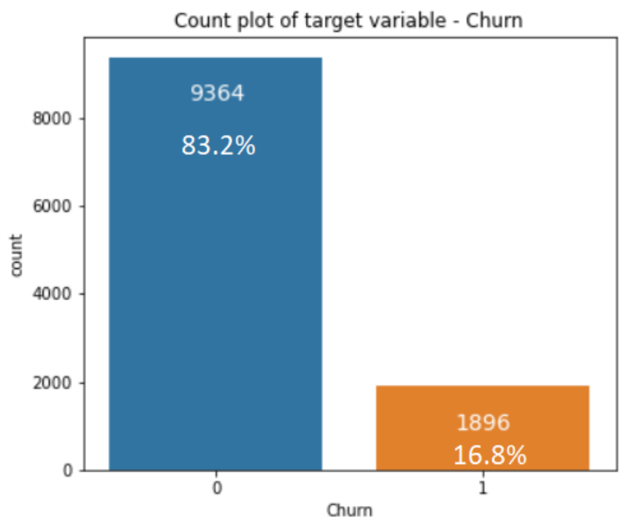


Figure 3-2 Count plot of target variable 'Churn'

If a dataset has equal distribution amongst the categorical values taken by the target variable (Churn), it would be called a balanced dataset. In a balanced dataset, the model learns to predict both classes with equal efficiency as there are equal number of observations.

In case of customer churn, any business would have less of churned customers and more of active customers. Similarly, this dataset also has close to 17% churned customers. Although this mimics real-life churn data, from a modelling perspective, this may pose a challenge.

Challenges posed by an imbalanced dataset:

- If there are no sufficient observations in the minority class, the model would be unable to learn the patterns in minority class (class of interest) well and may predict majority class better. Using resampling techniques such as oversampling minority class or under-sampling majority class may be beneficial. Compared to under-sampling majority class which would result in loss of data, oversampling using techniques like SMOTE (Synthetic Minority Oversampling) may help. This technique would help generate synthetic data for churned customers based on the existing churned customer observations. However, this technique may not always guarantee better model performance. Depending on the algorithm used and the type of SMOTE used, there could be overfitting in the train dataset.
- Accuracy as an evaluation metric would not be appropriate in imbalanced classification problems. Even if the algorithm predicts all customers as belonging to majority class, it would still result in an accuracy of 83.2%. Using Precision, Recall or F1-score for the minority class may be a better approach for evaluation.

In this problem, SMOTE would be one of the data treatments given. Models built on imbalanced data would be compared against models built on SMOTE balanced dataset and the best performance measure yielding data would be chosen. It may be possible that SMOTE does not give a good performance, hence a comparison would be required to make that decision.

3.3. Bivariate analysis & Multivariate analysis

Bivariate analysis shows the relationship between two variables. Here, the predictor variables have been taken and their relationship with the target variable has been plotted. The influence of the predictor variables on the target variable can be observed in these bivariate plots.

Continuous predictor variables that show some ability to separate the target variables

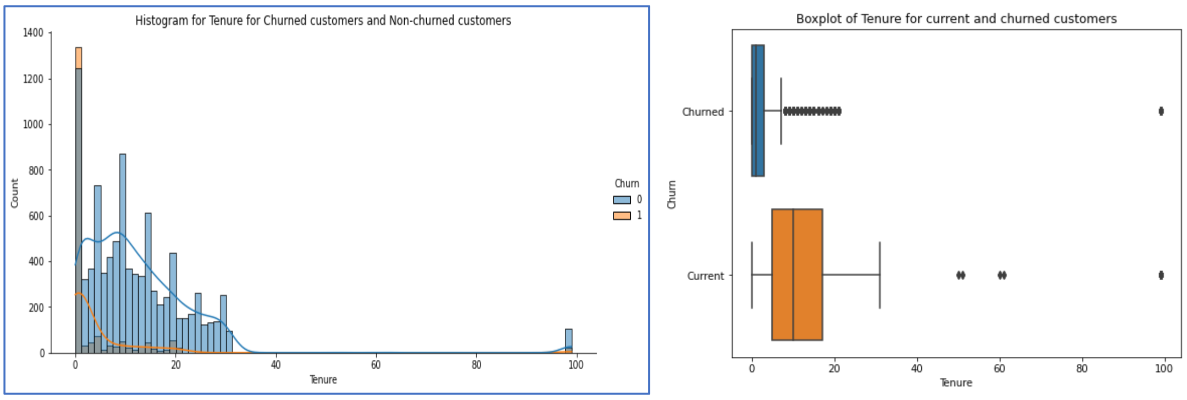


Figure 3-3 Tenure Vs Churn

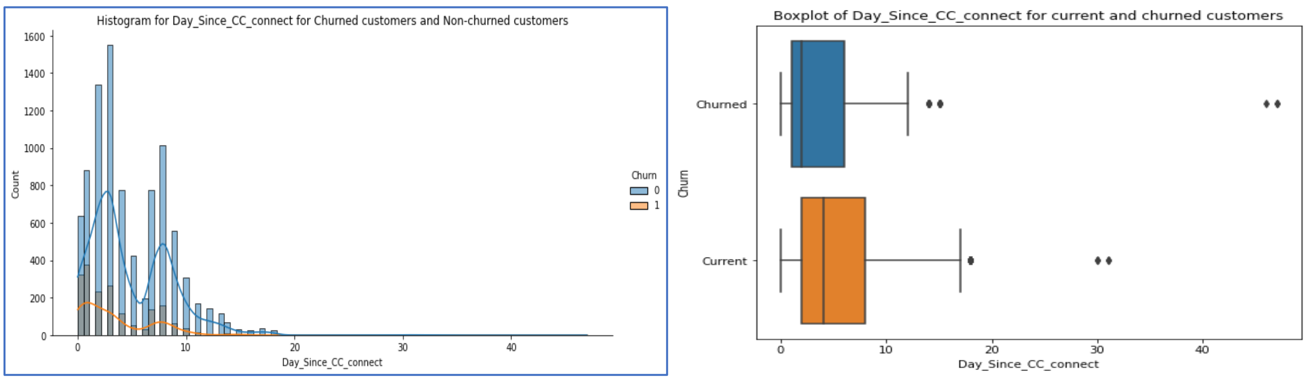


Figure 3-4 Day_since_CC_connect Vs Churn

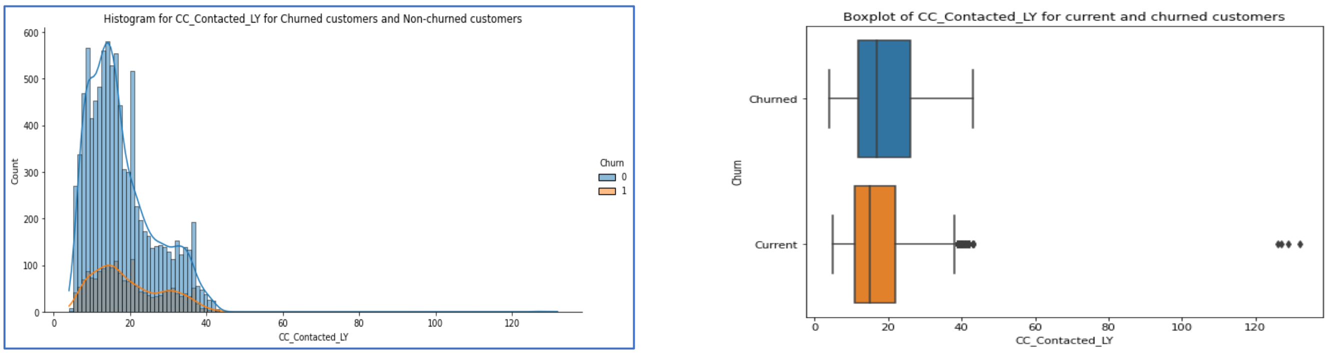


Figure 3-5 CC_Contacted_LY Vs Churn

Continuous predictor variables that aren't able to show a clear separation between target classes

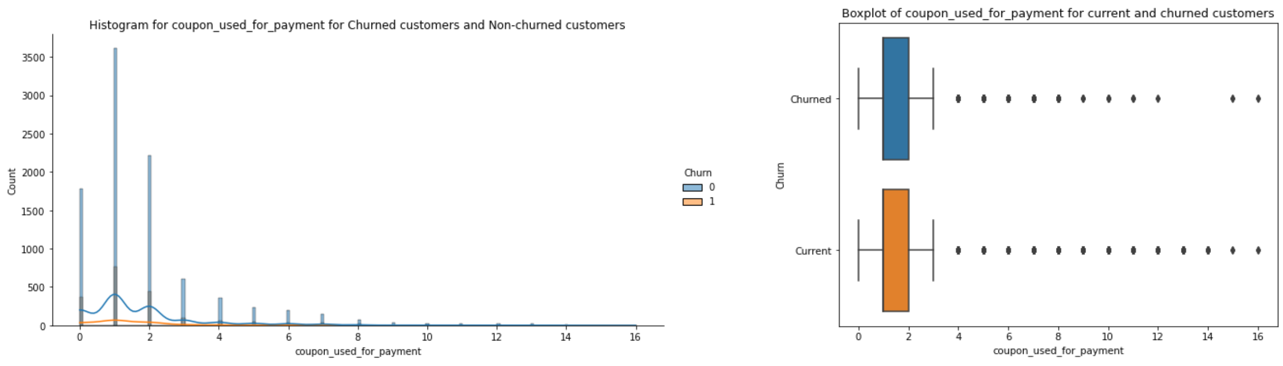


Figure 3-6 Coupon used for payment Vs Churn

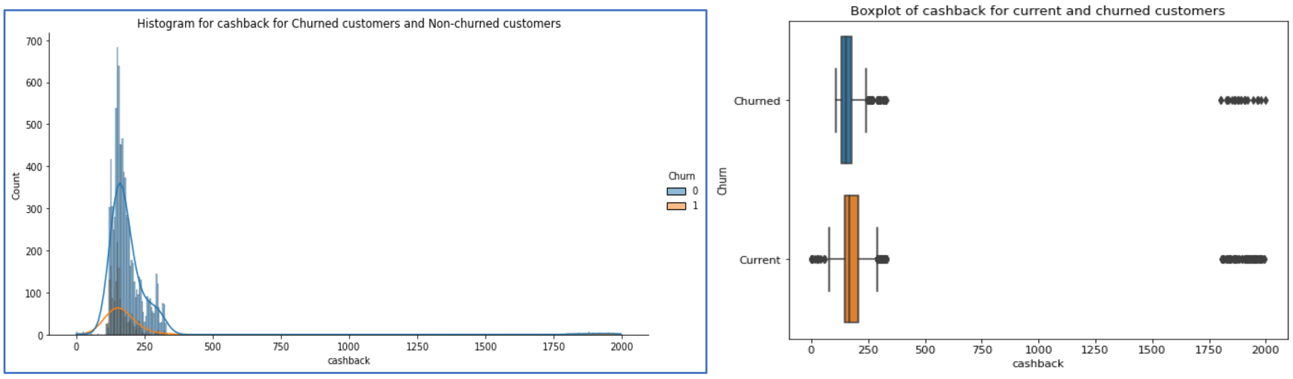
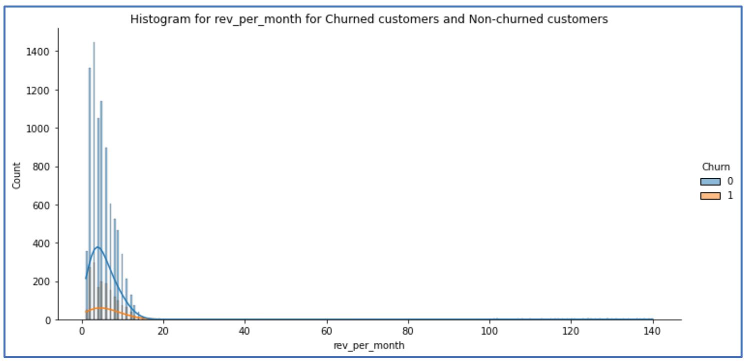


Figure 3-7 Cashback Vs Churn



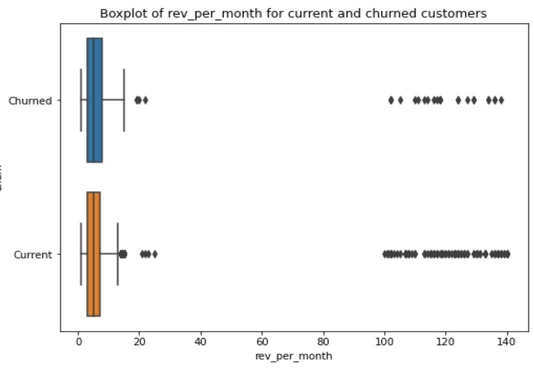
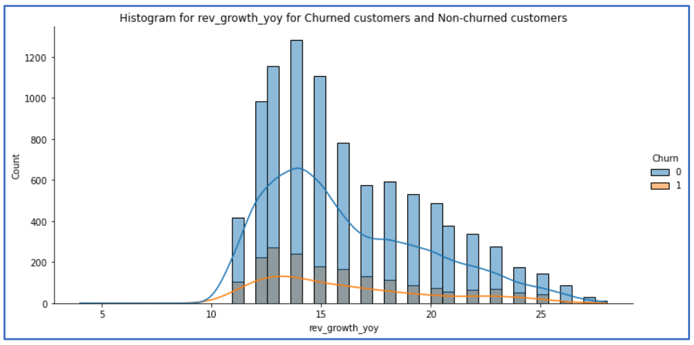


Figure 3-8 Revenue per month Vs Churn



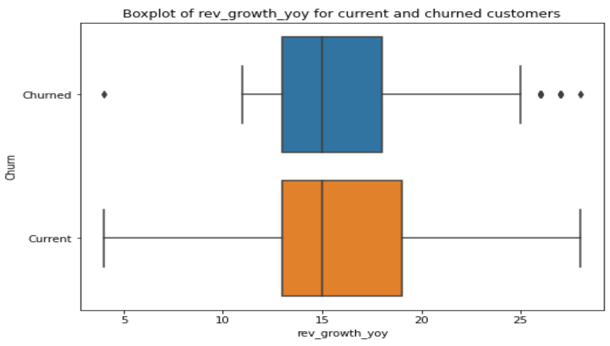


Figure 3-9 Revenue growth y-o-y Vs Churn

Observations: From the above plots we can see that variables such as Tenure, Days_since_CC_connect have some influence on the target variable. The median lines for Churned and Non-churned observations when plotted against these variables show a difference. Whereas, the medians in boxplots for variables such as coupon_used_for_payment, rev_growth_yoy do not show much difference in churned vs non-churned distributions.

Continuous variables: Anova (Analysis of Variance)

Analysis **o**f **Va**riance is a statistical method, used to check if the means of two or more groups that are significantly different from each other. Hypothesis for the test is as follows:

H0: *Means of all groups are equal*
Ha: *At least means of one pair of the groups is different*

Statsmodel library was used to perform the Anova test

Results of Anova test for following variables and churn

Variable	F-statistic	Probability of > F	Inference at significance level of 5%
Tenure	634.6	3.3E-36	Reject null hypothesis. The means are different. Variable significant to model building
CC_Contacted_LY	58.25	2.49E-14	Reject null hypothesis. The means are different. Variable significant to model building
rev_per_month	5.32	0.021	Reject null hypothesis. The means are different. Variable significant to model building
rev_growth_yoy	2.17	0.141	Cannot reject null hypothesis. The means are equal. Variable can be dropped
coupon_used_for_payment	2.47	0.116	Cannot reject null hypothesis. The means are equal. Variable can be dropped

Day_Since_CC_connect	243.9	2.1E-54	Reject null hypothesis. The means are different. Variable significant to model building
Cashback	11.32	0.001	Reject null hypothesis. The means are different. Variable significant to model building

Table 3-2 Results of Anova test

Observations: At a significance level of 0.05 (5%), the tests for the variables rev_growth_yoy and coupon_used_for_payment have given a p-value of greater than 0.05. In these two cases, Ho cannot be rejected, i.e., the means for the two groups churn=0 and churn=1 for these variables are the same. This implies that since the groups are not too different, these two variables cannot be significant predictors of the target variable. This is in line with what was visually observed using the bivariate box plots for these two variables.

Bivariate plots for categorical variables Vs churned

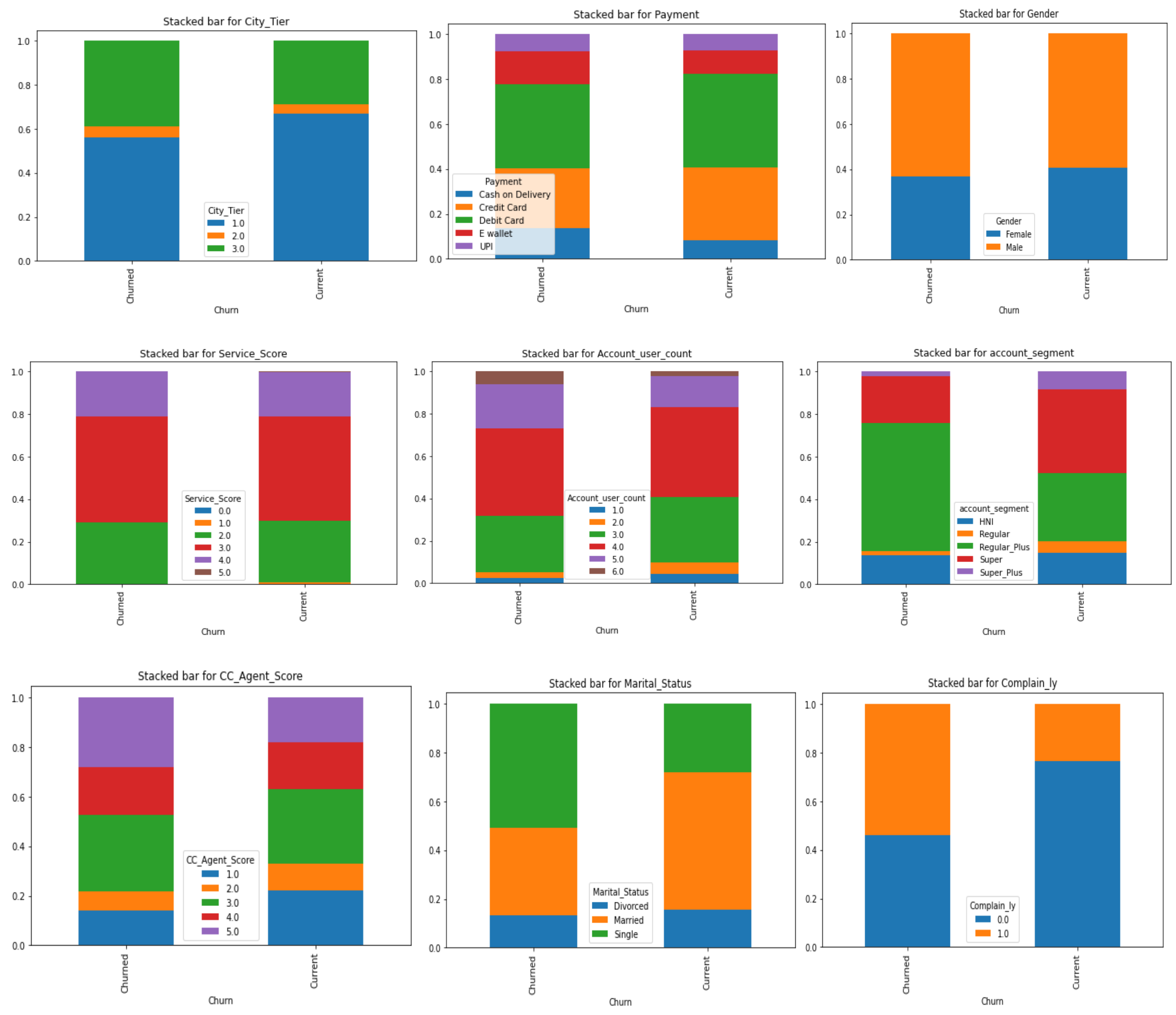


Figure 3-10 Stacked bar chart for categorical variables

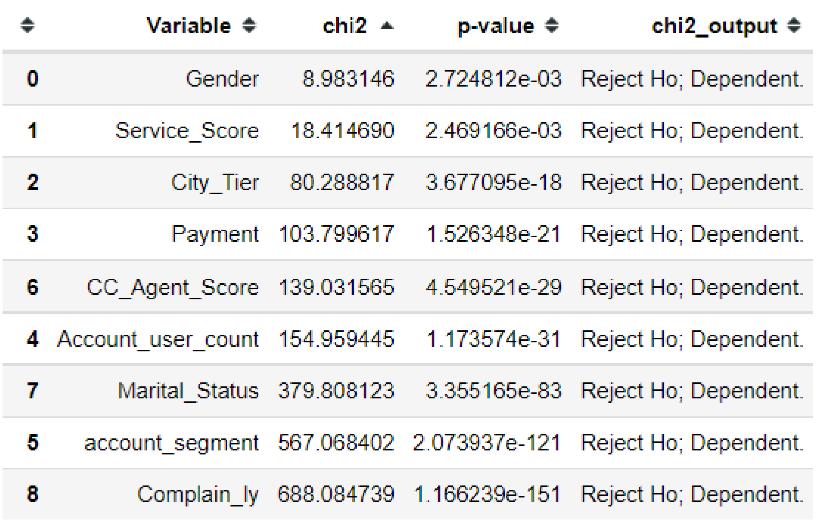
Observations: In the above stacked bar charts, active customers are called current customers. The first bar shows distribution of the categorical predictor variable being analysed within churned customer and the second bar shows distribution within active/current customers. Absolute comparison cannot be done as this is a stacked bar with the height of the bar representing 100% of churned and non-churned/current customers respectively. The respective absolute counts differ. Only interpretation from these plots is the % distribution between various categories of the categorical variable being plotted.

- From the above charts, some of the variables like city_tier, account_segment, Complain_ly, Marital_Status, CC_Agent_score seem to show a difference in distribution when churned vs current customers are considered. These variables may have an influence over the target variable churn and may contribute to the model.
- Other variables such as Gender and Service_Score have more or less similar distributions within Churned and Current/active customer bars. These variables may not have significant contribution towards the model.

Categorical variables: Chi-squared test of independence at significance level 0.05

These variables were subjected to Chi square test of independence to decide if they are statistically significant enough to be included in the model or not. The Chi-Square test of independence is used to determine if there is a significant relationship between two categorical variables. The frequency of each category for one categorical variable is compared across the categories of the second categorical variable. The data can be displayed in a contingency table where each row represents a category for one variable and each column represents a category for the other variable.

Null hypothesis	: There is no relationship between the two categorical variables.
Alternative hypothesis	: There is a relationship between the two categorical variables.



	Variable	chi2	p-value	chi2_output
0	Gender	8.983146	2.724812e-03	Reject Ho; Dependent.
1	Service_Score	18.414690	2.469166e-03	Reject Ho; Dependent.
2	City_Tier	80.288817	3.677095e-18	Reject Ho; Dependent.
3	Payment	103.799617	1.526348e-21	Reject Ho; Dependent.
6	CC_Agent_Score	139.031565	4.549521e-29	Reject Ho; Dependent.
4	Account_user_count	154.959445	1.173574e-31	Reject Ho; Dependent.
7	Marital_Status	379.808123	3.355165e-83	Reject Ho; Dependent.
5	account_segment	567.068402	2.073937e-121	Reject Ho; Dependent.
8	Complain_ly	688.084739	1.166239e-151	Reject Ho; Dependent.

Table 3-3 Results of chi-squared test

A p-value less than 0.05 (typically ≤ 0.05) is statistically significant. It indicates strong evidence against the null hypothesis, as there is less than a 5% probability the null is correct. Since the p-value returned for all the categorical variables is less than 0.05, the null hypothesis can be rejected. Hence at 5% level of significance, it may be concluded that churn is not independent of these categorical variables. Hence, we will proceed to retain all these categorical predictor variables at this point.

Pair plot for the numeric variables with hue set as target variable

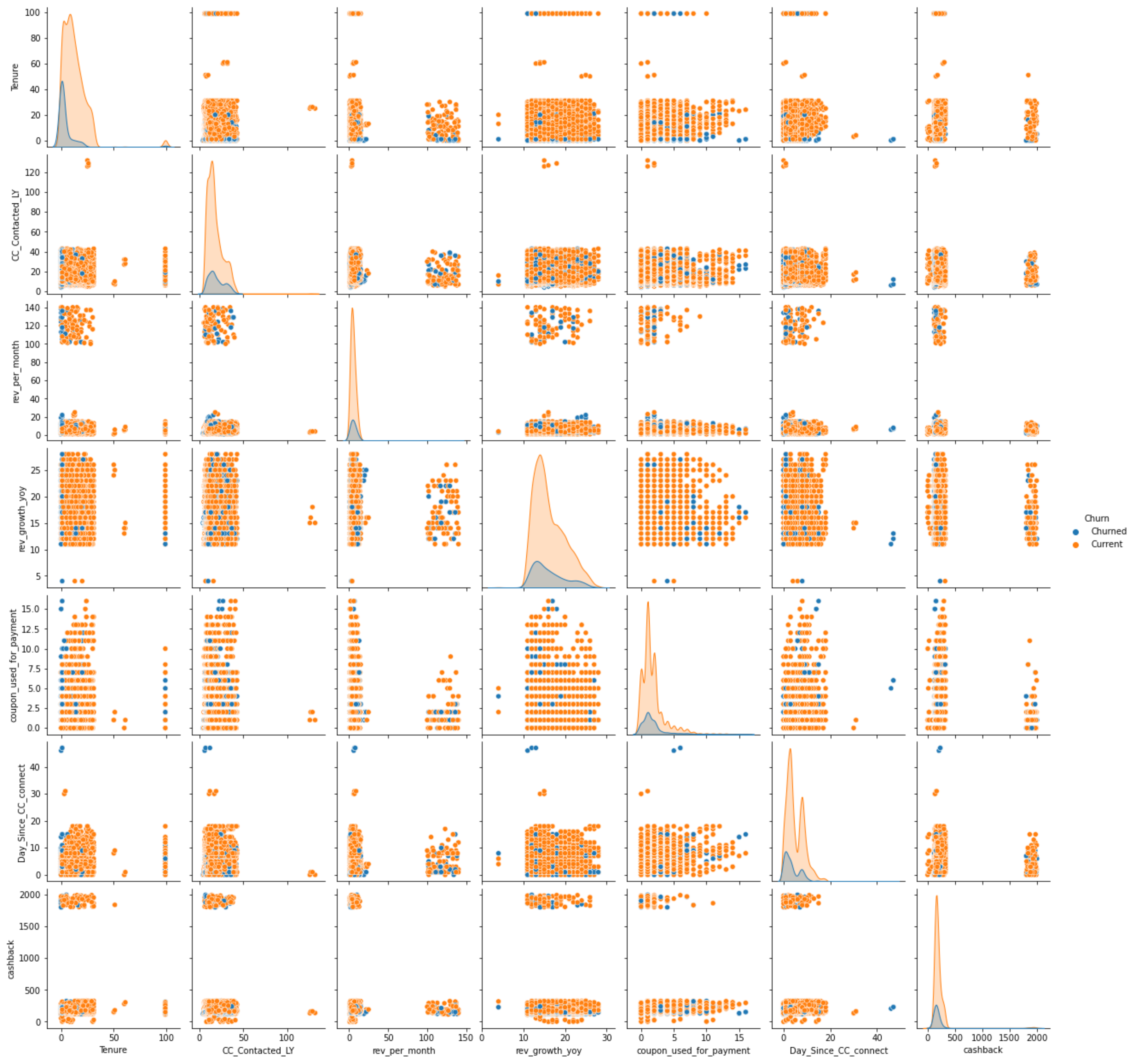


Figure 3-11 Pair plot for numeric predictor variables

Observations:

- Some of the variables clearly show presence of some clusters for e.g., rev_per_month and Day_Since_CC_connect
- The diagonal kde plot for Tenure shows a slight separation with customers who churned falling on the lower side of Tenure
- There is no linear relationship between any two continuous variables

Correlation heat map

The following heat map shows the correlation (pearson's) between various numeric predictors in the dataset. The purpose of doing a correlation matrix is to check if any of the variables have a strong correlation that can contribute to multi-collinearity. If there is a strong correlation > 0.80 between two variables, one of them can be dropped as it would not add much to the model's performance.

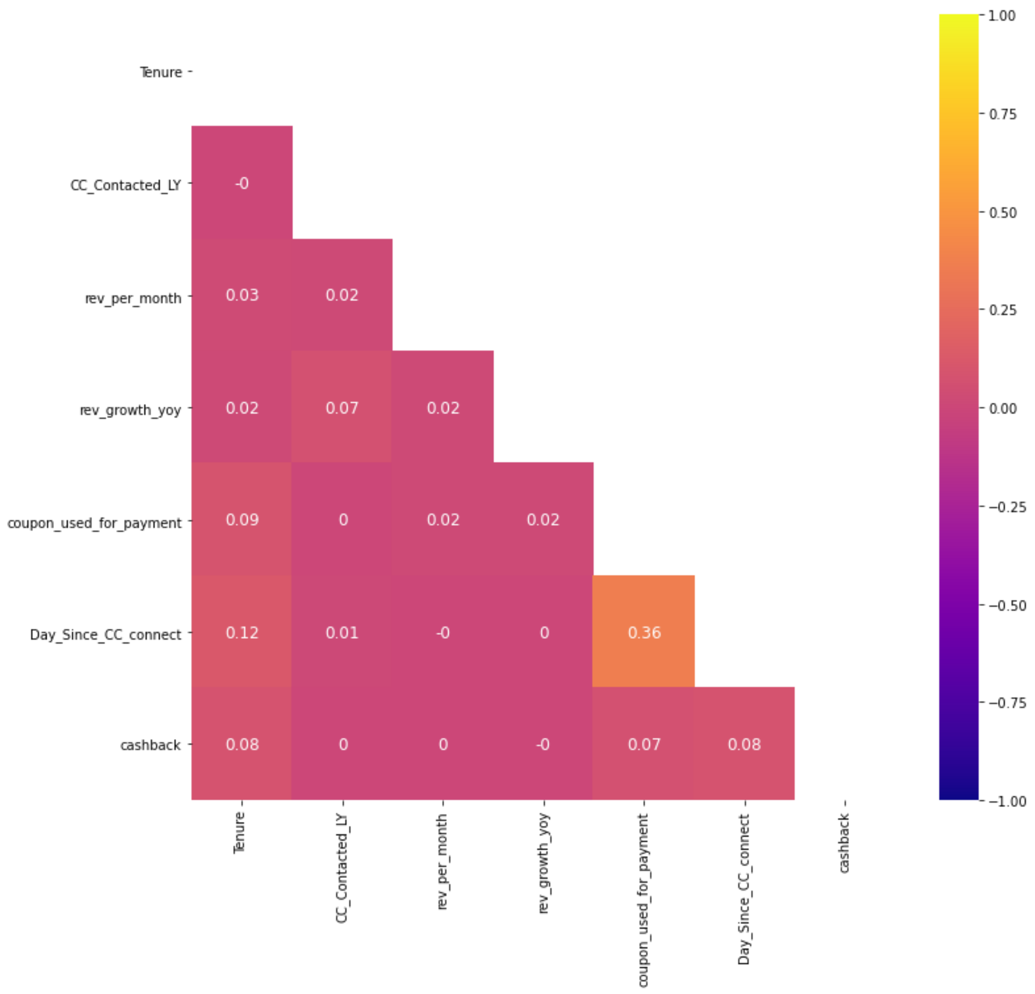


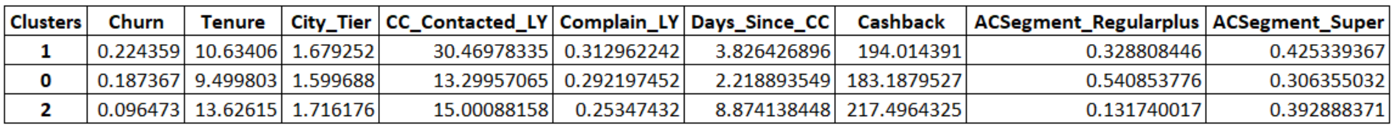
Figure 3-12 Correlation heatmap

Observation: None of the numeric variables show a strong correlation. Hence there is no need to drop any variable.

3.4. Clustering

- **Cluster analysis** or **clustering** is the task of grouping a set of objects in such a way that objects in the same group (called a **cluster**) are more similar to each other than to those in other groups (clusters).
- Clustering is an unsupervised task and given all the features in the dataset, the clustering algorithm is allowed to group customers such that each group has similar customers and customers of different groups are dissimilar.
- For this purpose, the processed dataset (cleaned, scaled, nulls imputed, outlier treated, categorical features encoded) without the target variable was used.
- As the dataset contained both continuous and categorical predictor variables, kprototypes function from Kprototypes library was used.
- The algorithm was run for 2,3 and 4 clusters and 3 clusters seemed to have the best separation. The cluster profile was formed by grouping the observations by clusters and finding the mean for all the features.

- Although churn was not part of the features for clustering, it was added part of the cluster profile so that it is possible to appreciate how churn varies for each cluster.



Clusters	Churn	Tenure	City_Tier	CC_Contacted_LY	Complain_LY	Days_Since_CC	Cashback	ACSegment_Regularplus	ACSegment_Super
1	0.224359	10.63406	1.679252	30.46978335	0.312962242	3.826426896	194.014391	0.328808446	0.425339367
0	0.187367	9.499803	1.599688	13.29957065	0.292197452	2.218893549	183.1879527	0.540853776	0.306355032
2	0.096473	13.62615	1.716176	15.00088158	0.25347432	8.874138448	217.4964325	0.131740017	0.392888371

Table 3-4 Cluster profile

The insights from clustering have been provided in the next sub-section.

3.5. Business Insights from EDA

- **Customer feedback:** 78% of customers have rated service as 3 or less than 3 (out of a scale of 5). Likewise, 61% of customers have rated customer care agents a score of 3 or less than 3 (out of a scale of 5). This indicates that the customer feedback is pointing to dissatisfaction or bare satisfaction in service and also in customer care engagement.
- **Relationship between Tenure and Churn:** In the bivariate histogram for Tenure vs Churn, it can be seen that the churn is very high for low tenures. For tenure between 0 and 1, around 51.85% customers have churned. The reason for this churn needs to be drilled down and addressed. Once the tenure increases, the histogram shows that proportion of churn decreases.
- **Relationship between Account segment and Churn:**
  More customers in Regular_Plus plan seem to churn. A comparison of this plan and competitors plan with same features and for same pricing range could be done to ascertain if the plan or pricing needs to be changed. Also, customer feedback for customers on this plan could be obtained and analyzed to find out why there is more churn in this account segment.
- **Relationship between Monthly revenue and Churn:**
  The % churn in high revenue customers is slightly more than the churn in lower revenue customers. This is would a cause for concern for the DTH provider that not only is there more churn but more high revenue customers are churning.
- **Relationship between Days since customer care connect and Churn**
  Days since customer care connect for churned customers is lesser than for active customers. This shows that churn has happened shortly after the customers have contacted customer care.
- **Relationship between Customer care contacted last year and Churn**
  The number of times customer care was contacted previous year was more in churned customers compared to active customers.
- **Relationship between Complaints made last year and Churn**
  Proportion of customers who complained is significantly more in churned customers compared to active customers.
- **Relationship between User count and Churn**
  Proportion of accounts with user counts 5 and 6 is more in churned customers compared to active customers.
- **Relationship between Payment type and Churn**
  Proportion of customers who have paid through E-wallet and Cash on delivery is more within churned customers compared to active customers.
- **Relationship between City tier and Churn**
  Proportion of customers who reside in Tier 3 cities is more within churned customers compared to active customers. Whether there is more competition in those cities compared to Tier-1 cities needs to be explored by the DTH provider.
- **Relationship between Marital status and Churn**
  Single customers have churned more compared to married or divorced customers.

Business Insights from cluster profiling

- Even though this is an unsupervised algorithm without the use of target variable in clustering, the profiling came up with clear separation of groups only for those features that also showed a high F-statistic and high Chi square value in the bivariate analysis.
- Higher the tenure, lesser the churn. But the cluster profile also shows that for low tenures (cluster 1 and 0), this is not holding good. To further explore this, it would be good to bin Tenure and check the relationship with Churn through a stacked bar.
- More Regular plus customers have a greater churn compared to least churn clusters
- High churn clusters had contacted customer care almost twice as the least churn customers
- The cashback was lower for high churn cluster compared to low churn cluster
- High churn cluster contacted customer care less times compared to low churn customers
- The cluster with maximum complaints last year also has the maximum churn. The cluster with minimum complaints last year has the minimum churn. It could be suggested that if any customer files a complaint, the complaint be followed up and resolved until the customer is satisfied. This process needs to be looked at if it can be strengthened.

4. Data cleaning and Pre-processing

4.1. Removal of unwanted variables

The following are the checks done to see if any columns can be dropped before modelling exercise:

- Any variable that has unique values for each observation – for e.g., AccountID field. This would not contribute to the model as it is just an ID field to tag each observation.
- Any variable that remains constant for all or most of the observations as this does not add any strength to prediction. As observed from the histogram (numeric) and count plots/value counts (categorical), there are no variables that have constant value for all observations.
- Any variable that has nulls in more than 25 to 30% of observations. The maximum nulls present are in cashback variable and that column contains 4% nulls. Hence no column will be dropped.
- Any predictor variable that has a strong correlation with another predictor variable. Then one of the variables can be dropped. As seen in the correlation heat map above, there are no strong correlations, hence no variable needs to be dropped.
- Any predictor variable that has a very weak correlation with the target variable. As seen from the Chi square test and Anova test in the bivariate analysis section above, two variables – rev_growth_yoy and coupon_used_for_payment were found to not be significant (at a significance level of 5%). Hence these two variables would be dropped from further processing.
- Three variables (Cashback, Service score, Cluster code) were dropped as part of VIF check in later step as explained in section 5.2

AccountID, rev_growth_yoy and coupon_used_for_payment have been removed

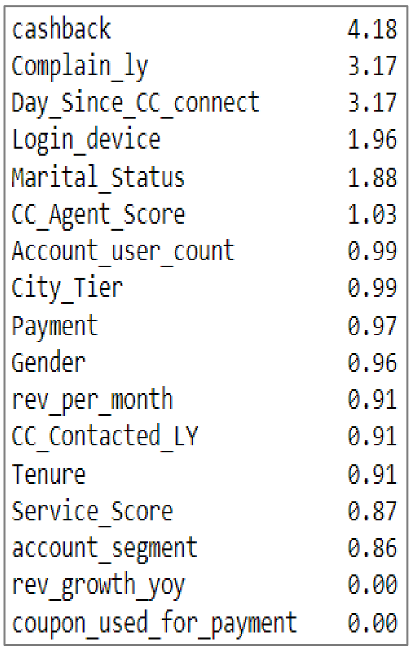
4.2. Addition of new variables

Cluster code has been added to the dataset (details about clustering is provided in section 3.5). It may be used when experimenting with model building.

4.3. Missing value treatment

Percentage of nulls

Percentage nulls or missing values present in the predictor variables of the dataset are as follows:



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

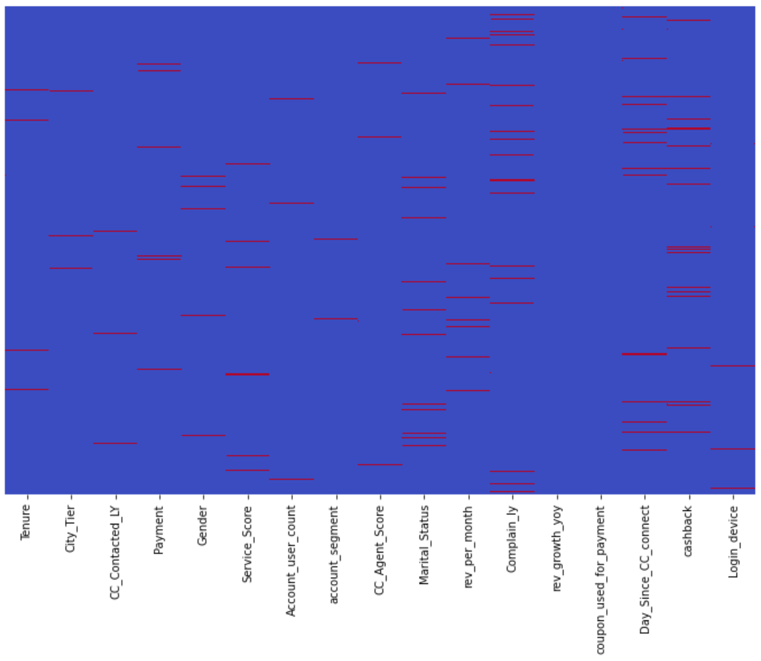


Table 4-1 Percent nulls in predictor variables and visualization of nulls

Missing value treatment

Missing value treatment was done using KNN imputation, a distance-based method. The following treatments were done as they are pre-requisites for missing value treatment using KNN.

- All variables need to be numeric. Any object-type categorical variables need to be encoded suitably (label/ one-hot encoding). The following columns were one hot encoded as it contained nominal categorical variables. The first encoded variable was dropped to avoid multi-collinearity

One hot encoded variables Payment','Gender','account_segment','Marital_Status','Login_device'

- After encoding, the column names were renamed in order to shorten, make it uniform and remove blank spaces that resulted out of some of the category values from one-hot encoded columns (e.g., Payment_Credit Card)
- All variables need to be scaled as KNN is a distance-based algorithm. Scaling was done using Standard Scaler function from SKLearn library for the predictor variables. The target variable was left as-is.
- Null imputing was done using Sklearn's KNNImputer function. This algorithm imputed missing values using K-nearest neighbors.

```
Tenure                   0
City_Tier                0
CC_Contacted_LY          0
Service_Score            0
User_Count               0
CC_Score                 0
Rev_Permonth             0
Complain_LY              0
Days_Since_CC            0
Cashback                 0
Payment_Creditcard       0
Payment_Debitcard        0
Payment_Ewallet          0
Payment_UPI              0
Gender_Male              0
ACSegment_Regular        0
ACSegment_Regularplus    0
ACSegment_Super          0
ACSegment_Superplus      0
Maritalstatus_Married    0
Maritalstatus_Single     0
Logindevice_Mobile       0
```

Table 4-2 Nulls in predictor variables after KNN imputing

4.4. Variable transformations

- **<u>Encoding:</u>** As mentioned in the above section, the variables **Payment, Gender, Account_Segment, Marital_Status and Login_device** are all categorical object type. They need to be converted to numeric variables. The categories in these variables do not have an order. Hence, they were one-hot encoded. When performing one-hot encoding, the variable is split into multiple columns with each column taking binary values corresponding to each category in the original attribute. Additionally, one of the columns in the one-hot encoded variables is dropped in order to avoid multi-collinearity which may cause performance degradation and interpretability issues in some of the models.
- **<u>Scaling:</u>** The dataset had been scaled as it is a pre-requisite for any distance-based algorithm like KNN imputer, K-means clustering, KNN and ANN. The scaled data can also be used by all models irrespective of whether they expect scaled input or not.
- No other transformation is expected for modeling as of now.

4.5. Outlier treatment

The following continuous variables have outliers.

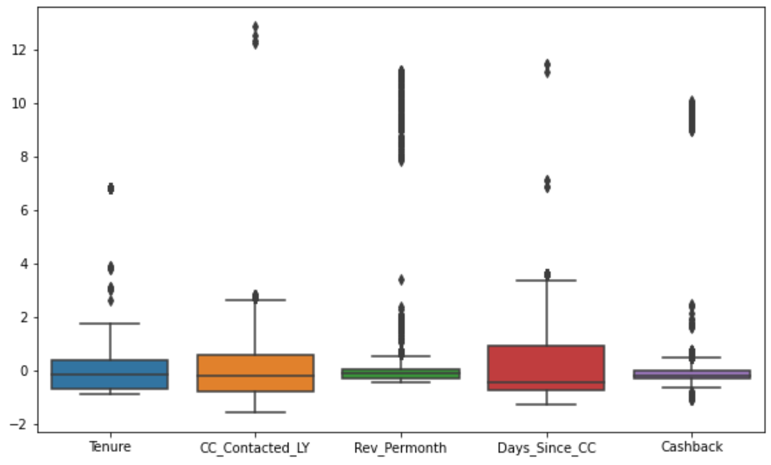


Figure 4-1 Box plot of variables containing outliers

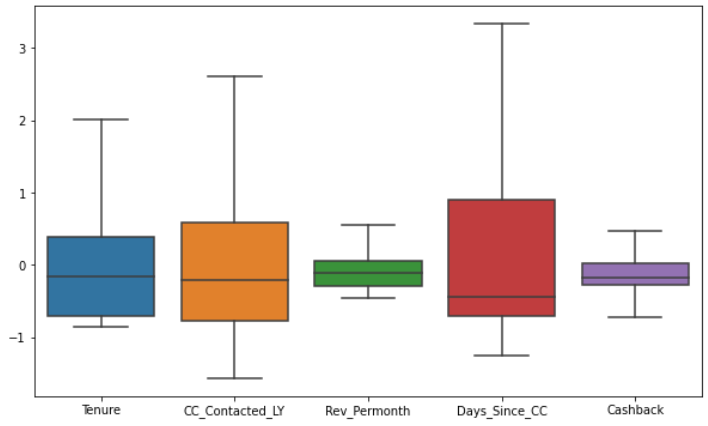
- Some outliers for certain variables are closer to the whisker, whereas there are a group of outliers that are far beyond the whisker with no in between values. For instance, rev_per_month has a huge space between 30 and 100 indicating absence of values in that range. Those outliers in the extreme values do not correlate with corresponding outliers in cashback field. We cannot rule out these outliers as incorrect values, they may belong to hotels with many rooms. But models like logistic regression are sensitive to outliers and may not give good performance if outliers are left untreated.

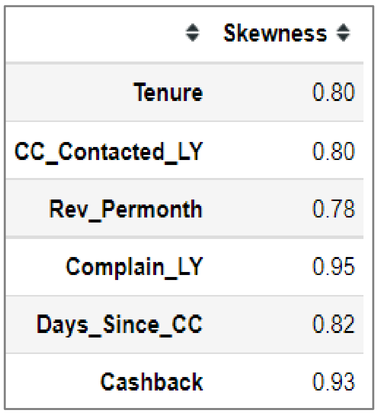
- Hence, two approaches to modelling were performed – one set of data with outliers treated for outlier sensitive models and other set of data with outliers not treated (left as-is) for outlier resistant algorithms such as Random-forest and during tuning/trials of other algorithms.
- Coupon_used_for_payment has a very limited range 0 to 16. Hence, for the purpose of this analysis, the outliers will not be treated (similar to categorical variables).

For the outlier treated dataset, outliers beyond upper and lower whiskers were treated by capping to the lower and upper range where

- lower_range= 1st quartile - (1.5 * IQR) and
- upper_range= 3rd quartile + (1.5 * IQR)

Where, IQR = 3rd quartile – 1st quartile value





	Skewness
Tenure	0.80
CC_Contacted_LY	0.80
Rev_Permonth	0.78
Complain_LY	0.95
Days_Since_CC	0.82
Cashback	0.93

Figure 4-2 Boxplot of variables post outlier treatment

5. Modelling Approach

5.1. Algorithms applicable for the given problem

- In this business case, the need is to predict whether a given customer would churn or not. This is a binary classification problem with only two prediction outcomes '0' – will not churn and '1' – will churn
- Since there is a target variable 'Churn' to be predicted, this is a supervised learning problem
- There are several algorithms that can be used for classification problems such as these
  - **Linear classification:** Logistic Regression, Linear Discriminant Analysis, Naïve Bayes
  - **Non-linear classification algorithms:** SVMs non-linear adaptations, Decision tree, K-nearest neighbor, Artificial Neural Network
  - **Ensemble models:** Random Forest, Adaboost, Gradient Boost
- These algorithms have certain assumptions about the data on which they are fit. Depending on the nature of the data, algorithms would give good or poor performance.

5.2. Methodology

- Different treatments of pre-processed data were prepared - with and without outliers, with and without SMOTE resampling, with and without scaling.
- VIF (Variance Inflation Factor) was calculated for all the predictor variables. One by one predictor variables whose VIF was more than 5 were identified. 4 variables – Cashback, Service Score, Clusters and User count had VIF greater than 5. It is to be noted that even though User count had a high VIF value, it was retained as it showed significant Chi square value as part of EDA. The remaining 3 variables were dropped before processing the datasets; hence, none of the models have used these variables as predictors.

- It is to be noted that rev_growth_yoy and coupon_used_for_payment were dropped after Anova and Chi-square test were conducted and double checked against EDA plots. Hence none of the models have used these variables as predictors.
- Scaled data was used for distance-based algorithms such as KNN and ANN. Smote resampled data was also tried for few algorithms.
- Data was split into train and test set in the ratio 70:30. 7882 records were assigned to train dataset and 3378 records were assigned to test dataset. The selection was done in such a manner that both sets had similar distribution of target variable as in the original dataset (i.e., 16.8% churn=1s).
- 8 algorithms were chosen and for each algorithm
    - Base model (with default hyperparameters) was constructed and evaluation on train and test datasets performed
    - Different data was used and performance measured and recorded
    - Hyperparameters for the algorithm were tuned using SKlearn library's GridSearchCV and also manually if required.
    - Performance was again measured for the tuned algorithm
    - Feature importance was extracted from the model through in-built attributes for certain models. Amongst black box models, Sklearn's Permutation feature importance was used as a wrapper function on the model to obtain feature importance.

5.3. Evaluation metrics for model comparison

- The final best model was decided based on comparison of evaluation metrics for train and test data of all the models. The following criteria was used:
    - The train and test performances should be comparable – i.e., no overfit or underfit. The model should be robust and reliable for use with unknown data.
    - Maximum Precision score for 1s. The problem statement states that Revenue assurance team do not want to give unnecessary freebies. If precision for 1s (churn customers) is high, then the actual churns out of the model's predicted churns would be high and hence the revenue assurance team's criteria would be satisfied.

      Precision = True positive/ (True positive + False positive)
    - High F1-score (to ensure Recall is also good) for Churns. High precision should not come at the cost of recall. The model should be able to get a good part of the actual churns as predicted churns so that this prediction exercise is meaningful and the DTH provider can actually address their churn problem. Hence combination of Precision and Recall to get the F1-score is important.
    - Model should be interpretable.
    - Model should not be computationally expensive (like KNN).

6. Model Building and Tuning

8 algorithms were tried for this dataset – Logistic Regression, Linear Discriminant Analysis, Support Vector Machine, Artificial Neural Network, K-Nearest Neighbours, Random Forest, Adaboost, Gradient Boost. Algorithm implementations from Sklearn package was used for all algorithms. In addition to SKlearn, Logistic regression algorithm was also executed using Statsmodel package. The Appendix contains detailed information for all the 8 models including details on the base and the tuned model performance along with their interpretations.

For sake of brevity, this section contains a tabulated form of individual models (for each of the 8 algorithms) and the results from tuning exercise. The feature importances for the top 3 best performing models are also shown. Only for Gradient Boost, the selected model, all details are provided. For complete details on all the algorithms, base model and best model performance metrics, confusion matrix, classification report for train and test data, feature importances and model interpretation, please refer the **Appendix – Annexure B**

6.1. Effort for model tuning

Model performance improvement was accomplished using the following methods:

- Around 8 different algorithms were tried across linear, non-linear and ensemble methods.
- The first model for each algorithm was the base model with the default hyperparameters. Model tuning to achieve better performance was done by tuning the hyperparameters using Grid Search CV, a function offered by Sklearn to automatically try out multiple parameter options for each hyperparameter.
- The underlying data was changed (Smote resampled/ non-resampled, outlier treated/not treated) and the effect of model performance observed for some of the models.
- Ensemble methods were used (Random Forest, Adaboost, Gradient Boost) and their hyperparameters were also tuned.

The following tables show all the algorithms tried, various model tuning trials done and the performance for each one of them in train and test dataset. The data treatment done is also shown as part of the table.

Hyperparameters are part of the code and hence not included here. The best model for each algorithm has been highlighted in green. The model reference number can be used to locate the model in the python code.

6.1.1. Logistic Regression

Logistic regression model performs well when the data is linearly separable. It assumes that there is linearity between target and predictor variables. It is a parametric model hence it can be fast compared to KNN. It also provides coefficients that helps with model interpretability. Hence the first model tried was Logistic regression. Both SKLearn and Statsmodel (python libraries) implementations were tried for Logistic regression.

Model reference	Data treatment		Hyper parameters	Train data					Test data				
	Smote	Scaling		Accuracy	Precision	Recall	F1	AUC	Accuracy	Precision	Recall	F1	AUC
LR_model1	**No**	**No**	**Default base model**	**0.89**	**0.77**	**0.51**	**0.61**	**0.88**	**0.89**	**0.78**	**0.5**	**0.61**	**0.87**
LR_model2	No	No	Default base model, RFE variables=8	AUC = 0.77					AUC = 0.77				
LR_model3	No	No	Default base model, RFE variables =12	AUC = 0.78					AUC = 0.78				
LR_model4	No	No	Default base model, RFE vars=16	0.89	0.78	0.49	0.6	0.88	0.89	0.8	0.47	0.59	0.87
LR_model5	No	No	Default base model, RFE vars=18	0.89	0.77	0.49	0.6	0.88	0.89	0.79	0.48	0.6	0.87
LR_model6	No	No	Default base model, RFE vars=19	0.89	0.77	0.49	0.6	0.88	0.89	0.8	0.48	0.6	0.87
LR_model7	No	No	Gridsearch CV, best model for f1 score	0.89	0.77	0.5	0.61	0.88	0.89	0.78	0.49	0.6	0.87
LR_model8	Yes	No	Default base model	0.81	0.8	0.82	0.81	0.89	0.79	0.44	0.79	0.56	0.87
LR_model9	No	Yes	Default base model	0.89	0.77	0.5	0.61	0.88	0.89	0.78	0.49	0.6	0.87
LR_model10 Statsmodel	No	No	Iterated 4 times to remove 4 variables	0.89	0.78	0.51	0.61	0.88	0.89	0.79	0.48	0.6	0.87

Table 6-1 Model tuning and Performances - Logistic regression

LR_model1 is the best model out of the Logistic regression models.

The detailed information regarding base model, best performing model, their train-test metrics, model interpretability are provided in section 9.2.1

6.1.2. Linear Discriminant Analysis

Linear Discriminant Analysis (LDA) is also another machine learning classifier. It works well when there are linearly separable classes in data. It assumes that the underlying data has a gaussian distribution but can perform well even if assumptions are violated.

Data Used			Hyper	Train data					Test data				
Outliers	Smote	Scaling	parameters	Accuracy	Precision	Recall	F1	AUC	Accuracy	Precision	Recall	F1	AUC
Treated	No	No	Default base model	0.89	0.77	0.47	0.58	0.88	0.88	0.77	0.45	0.57	0.86
Treated	**No**	**No**	**Gridsearch CV, best model for f1 score**	**0.89**	**0.77**	**0.47**	**0.59**	**0.88**	**0.88**	**0.77**	**0.45**	**0.57**	**0.86**

Table 6-2 Model tuning – LDA models performance

The detailed information regarding base model, best performing model, their train-test metrics, model interpretability are provided in section 9.2.2

6.1.3. Support Vector machine

Support vector machine (SVM) is a popular machine learning algorithm that can be used for classification as well as regression. It works well when there are higher dimensions as well. It is very versatile as there are different kernel functions that can be specified to work well with the given data.

Model		Data Used			Hyper	Train data					Test data				
reference	Algorithm	Outliers	Smote	Scaling	parameters	Accuracy	Precision	Recall	F1	AUC	Accuracy	Precision	Recall	F1	AUC
SVM_model1	SVM	Treated	No	Yes	Default base model	0.94	0.93	0.71	0.8	0.97	0.93	0.9	0.64	0.75	0.94
SVM_model2	**SVM**	**Treated**	**No**	**Yes**	**Gridsearch CV, best model for f1 score**	**0.99**	**0.93**	**1**	**0.96**	**1**	**0.97**	**0.87**	**0.93**	**0.9**	**0.98**
SVM_model3	SVM	Not treated	No	Yes	Gridsearch CV, best model for f1 score	0.98	0.89	0.99	0.94	1	0.95	0.83	0.92	0.87	0.97

Table 6-3 Model tuning – SVM models performance

The detailed information regarding base model, best performing model, their train-test metrics, model interpretability are provided in section 9.2.3

6.1.4. Artificial Neural Network

Artificial Neural Network (ANN) is a powerful machine learning algorithm that can be used for classification as well as regression. This algorithm can learn the complex patterns in underlying data. There is a tendency to overfit, but that can be controlled in the tuning exercise using the hyperparameters.

Model	Data Used			Hyper	Train data					Test data				
reference	Outliers	Smote	Scaling	parameters	Accuracy	Precision	Recall	F1	AUC	Accuracy	Precision	Recall	F1	AUC
ANN_model1	**Treated**	**No**	**Yes**	**Default base model**	**1**	**1**	**1**	**1**	**1**	**0.98**	**0.94**	**0.91**	**0.93**	**0.99**
ANN_model2	Treated	No	Yes	Gridsearch CV, best model for f1 score	0.99	0.98	0.97	0.97	1	0.97	0.92	0.89	0.91	0.99
ANN_model3 alpha = 0.2	Treated	No	Yes	Default base model with alpha = 0.2	0.97	0.95	0.89	0.92	0.99	0.95	0.92	0.8	0.85	0.97
ANN_model3 alpha = 0.1	Treated	No	Yes	Default base model with alpha = 0.1	0.99	0.98	0.95	0.97	1	0.97	0.94	0.85	0.9	0.98
ANN_model3 alpha = 0.05	**Treated**	**No**	**Yes**	**Default base model with alpha = 0.05**	**0.99**	**0.98**	**0.98**	**0.98**	**1**	**0.97**	**0.94**	**0.9**	**0.92**	**0.99**
ANN_model3 alpha = 0.04	Treated	No	Yes	Default base model with alpha = 0.04	1	0.99	0.98	0.99	1	0.97	0.94	0.9	0.92	0.99
ANN_model3 alpha = 0.03	Treated	No	Yes	Default base model with alpha = 0.03	1	0.99	0.98	0.99	1	0.97	0.95	0.86	0.91	0.99

Table 6-4 Model tuning – ANN models performance

The detailed information regarding base model, best performing model, their train-test metrics, model interpretability are provided in section 9.2.4

6.1.5. K-Nearest Neighbour

KNN classifier works by looking at K-Nearest Neighbours to the given datapoint. It decides the target value based on its neighbours. KNN works on a principle assuming every data point falling near to each other is falling in the same class. It is also a black box model and lacks interpretability. Since it is non-parametric, it may be computationally expensive and require more memory to store training data. It also has a tendency to overfit. Although this model was tried on the given data and tuned extensively, due to the above said reasons, it has been decided not to select this as best model even if model performance is good.

Model reference	Data Used			Hyper parameters	Train data					Test data				
	Outliers	Smote	Scaling		Accuracy	Precision	Recall	F1	AUC	Accuracy	Precision	Recall	F1	AUC
KNN_model1	Treated	No	Yes	Default base model	0.98	0.95	0.93	0.94	1	0.96	0.88	0.85	0.87	0.98
KNN_model2	**Treated**	**No**	**Yes**	**GridSearchCV, best model for f1 score**	**1**	**1**	**1**	**1**	**1**	**0.98**	**0.93**	**0.93**	**0.93**	**0.99**

Table 6-5 Model tuning – KNN models performance

As the train data seemed to overfit, a 5-fold cross validation was run on complete data to ensure that the test data performance measure f1-score is holding up. The 5-fold cross validation gave a mean F1 score of 0.90 across 5 folds. The minimum f1 score in one of the folds was 0.85. Also, KNN as a model is computationally expensive. Hence although KNN seemed to give good performance as far as metrics is concerned, this was not considered to be selected as final model.

The detailed information regarding base model, best performing model, their train-test metrics, model interpretability are provided in section 9.2.5

6.1.6. Ensemble method – Random Forest

Random forest is an ensemble machine learning algorithm that uses bootstrapping to reduce variance in the underlying decision trees. It also selects only a subset of features for each node split decision. Since it is an ensemble of trees with varying features for each node, each tree is different from another. Random forest is resistant to outliers and does not require the data to be scaled.

Model reference	Data Used			Hyper parameters	Train data					Test data				
	Outliers	Smote	Scaling		Accuracy	Precision	Recall	F1	AUC	Accuracy	Precision	Recall	F1	AUC
RF_model1	Treated	No	No	Default base model	1	1	1	1	1	0.97	0.97	0.86	0.91	0.99
RF_model2	**No**	**No**	**No**	**Default base model with outliers**	**1**	**1**	**1**	**1**	**1**	**0.97**	**0.98**	**0.86**	**0.92**	**0.99**
RF_model3	Treated	No	No	GridSearchCV, best model for f1 score	0.98	0.98	0.91	0.94	1	0.95	0.9	0.78	0.84	0.98

Table 6-6 Model tuning – Random forest models performance

The data with outliers gave a slightly better performance compared to outlier treated data. But the difference between Recall in train and test is more than 10%. The model has overfit on train data and hence this was not considered for selection as final model.

The detailed information regarding base model, best performing model, their train-test metrics, model interpretability are provided in section 9.2.6

6.1.7. Ensemble method – Adaboost

An AdaBoost classifier is a meta-estimator that begins by fitting a classifier on the original dataset and then fits additional copies of the classifier on the same dataset but where the weights of incorrectly classified instances are adjusted such that subsequent classifiers focus more on difficult cases.

Model	Data Used			Hyper	Train data					Test data				
reference	Outliers	Smote	Scaling	parameters	Accuracy	Precision	Recall	F1	AUC	Accuracy	Precision	Recall	F1	AUC
ADA_model1	Treated	No	No	Default base model	0.9	0.74	0.61	0.67	0.91	0.9	0.75	0.6	0.67	0.9
ADA_model2	**Treated**	**No**	**No**	**GridSearchCV, best model for f1 score**	**0.9**	**0.75**	**0.6**	**0.67**	**0.92**	**0.9**	**0.76**	**0.6**	**0.67**	**0.91**

Table 6-7 Model tuning – Adaboost models performance

The performance of this model is not as good as SVM or ANN. Hence this was not selected as best model.

The detailed information regarding base model, best performing model, their train-test metrics, model interpretability are provided in section 9.2.7

6.1.8. Ensemble method - Gradient Boost

Gradient boost is an ensemble machine learning algorithm that trains underlying models in a gradual, additive and sequential manner.

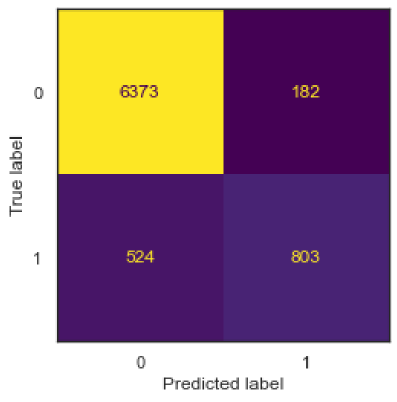
In this modelling exercise, SKlearn's Gradient Boost classifier function was used for modelling:

- The base model was run with default hyperparameters and the performance metrics noted. Tuning was later done using GridSearchCV.
- Model performance metrics for base model and best model have been provided in the below sections

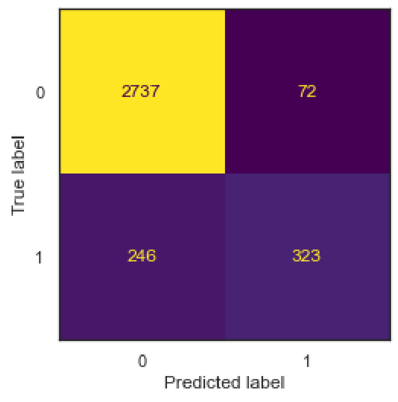
The details of this algorithm, its tuning and performance metrics have been provided in this section. The reasons why the tuned Gradient boost was selected as best model and model interpretation are provided in section 7.

6.1.8.1. Gradient boost base model with default hyperparameters

Train dataset Confusion Matrix



Test dataset Confusion Matrix

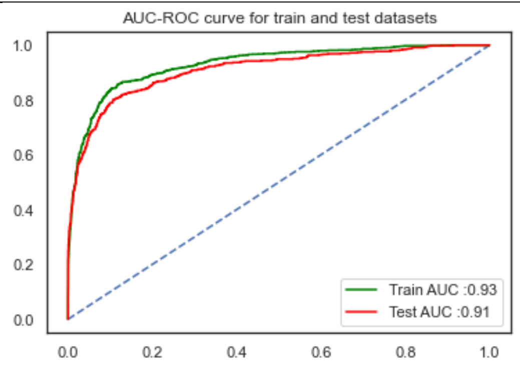


Train dataset Classification report

```
              precision    recall  f1-score   support

           0       0.92      0.97      0.95      6555
           1       0.82      0.61      0.69      1327

    accuracy                           0.91      7882
   macro avg       0.87      0.79      0.82      7882
weighted avg       0.91      0.91      0.90      7882
```

Test dataset Classification report

```
              precision    recall  f1-score   support

           0       0.92      0.97      0.95      2809
           1       0.82      0.57      0.67       569

    accuracy                           0.91      3378
   macro avg       0.87      0.77      0.81      3378
weighted avg       0.90      0.91      0.90      3378
```



The model is robust (no overfit or underfit) but the recall is quite poor in both train and test data. Hyperparameter tuning was done to see if performance can be improved on the model.

6.1.8.2. Gradient Boost - Model tuning

- The model's hyperparameters were tuned using GridSearchCV function from Sklearn library and model constructed using the best parameters selected. The tuning parameters for Gridsearch and individual scoring metrics for each parameter grid have been provided in Appendix – Annexure A.

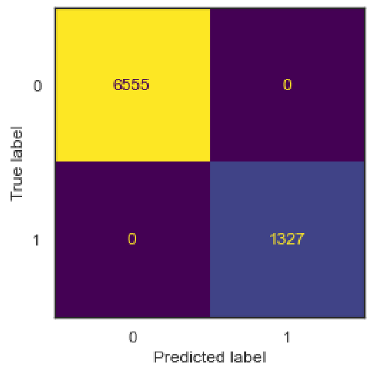
  - Tuning improved the model performance considerably

Model	Data Used			Hyper	Train data					Test data				
reference	Outliers	Smote	Scaling	parameters	Accuracy	Precision	Recall	F1	AUC	Accuracy	Precision	Recall	F1	AUC
GB_model1	Treated	No	No	Default base model	0.91	0.82	0.61	0.69	0.93	0.91	0.82	0.57	0.67	0.91
GB_model2	**Treated**	**No**	**No**	**GridSearchCV, best model for f1 score**	**1**	**1**	**1**	**1**	**1**	**0.99**	**0.99**	**0.94**	**0.96**	**1**

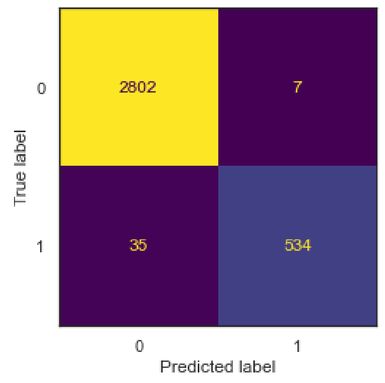
Table 6-8 Model tuning – Gradient Boost models performance

6.1.8.3. Gradient Boost best model with tuned hyperparameters

Train dataset Confusion Matrix



Test dataset Confusion Matrix

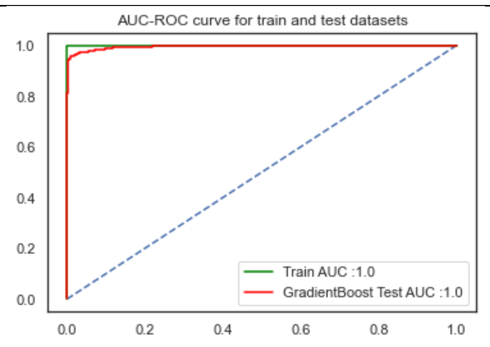


Train dataset Classification report

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00      6555
           1       1.00      1.00      1.00      1327

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

Test dataset Classification report

```
              precision    recall  f1-score   support

           0       0.99      1.00      0.99      2809
           1       0.99      0.94      0.96       569

    accuracy                           0.99      3378
   macro avg       0.99      0.97      0.98      3378
weighted avg       0.99      0.99      0.99      3378
```



Since the tuned model has shown 1s in all the train data performance parameters, it seems to have learnt all the noise in training dataset well. The test data shows a performance that is within 10% of train data performance. However, in order to ensure that it has not overfitted on train and the test performance can be achieved with unseen data, 5-fold and 10-fold cross validation on entire data was done. The mean f1-score for 5-fold (0.96) was the same as test f1-score (0.96) and 10-fold cross validation showed an increase in mean f1-score (0.98). All the folds also showed individual f1-scores that was comparable or better than test f1-score.

7. Model validation

- The given data was split into train and test dataset in the ratio 70:30. The test dataset was held out and kept for validation purpose only.
- All models were trained only on the train dataset. The trained model was used to predict train dataset target variable. The performance metrics such as Accuracy, F1-score, Precision, Recall, confusion matrix, ROC curve and AUC was observed and recorded on train dataset.
- The trained model was then used to predict target variable on test dataset. All the above said performance metrics were observed and recorded for the test data performance as well.
- If train data seemed to be giving all 1's and the difference between train and test performances are not off by more than 10%, a validation of test scores was done by doing 5-fold and 10-fold cross validation on entire dataset. The scores on this cross validation were compared to test dataset scores to ensure that model had not overfitted on train dataset.

7.1. Criteria for the best performing model

Primary criteria

- **Precision, Recall & F1-score for 1s:** This is the case of class imbalance as the dataset has 16.8% churns. In this case study 'Precision' and 'Recall' of class 1 or the minority class is most important. Combining these 2 metrics, F1-score for class 1 is also used in the comparison.
    - **Precision** is defined as True positive/ (True Positive + False Positive). It answers the question - Out of all customers that the model identifies as churning customers, how many are actual churners? This is the most important factor in this case as budget for offers to retain customers would be limited and more false positives would mean spending that budget on customers who would not churn anyway. The problem statement states that the revenue assurance team is very stringent about providing freebies where it is not required. Translated into metric, this would mean that the precision for 1's/churns should be highest.
    - **Recall** is defined as True Positive / (True Positive + False Negative). It answers the question - Out of all actual churning customers, how many does the model correctly identify as churners? This is very important as the purpose of the project is to identify as many churners as possible in order to give special offers in order to retain them. For the DTH provider, customer acquisition cost is very high and hence retention is of utmost importance. This is the whole reason why the project exists and hence Recall for 1s is also important in this case.

- **F1-score** is harmonic mean of Precision and Recall. Where both Precision and Recall are important, this metric can be used as a single metric that needs to be optimized for.

Secondary criteria

- **Accuracy:** This is a classification problem and the dataset has class imbalance. That is, the proportion of churn and non-churn customers are not equal. With imbalanced classes, it's easy to get a high accuracy without actually making useful predictions. So, accuracy as an evaluation metric makes sense only if the class labels are uniformly distributed. We are concerned with correct prediction of churn customers (class 1). Hence 'Accuracy' is not a correct metric to compare various models but for the sake of completeness and to ensure that 0s (majority class) are not overlooked, it is still recorded in the comparison matrix.

- **AUROC:** In addition to the above metrics, the Area under curve of ROC curve is also used to evaluate model performance. An ROC curve (or receiver operating characteristic curve) is a plot that summarizes the performance of a binary classification model on the positive class. It is a curve that is constructed by evaluating true positives and false positives for different threshold values. As visualizing ROC curve is difficult for actual comparison, the Area Under Curve (AUC) metric helps with a numeric comparison. The closer the AUC is to 1, the better the model. However, like accuracy this also works well for balanced dataset[4]. For the sake of completeness, this is also recorded in comparison matrix.

The following table shows performance of the best model from each algorithm built so far. The metrics given in the below table are all for minority class/churners (1s) which is the class of interest. The best performer has been highlighted in green. The figure below that shows the ROC curve for all models and the best performer is the blue line.

Model reference	Algorithm used	Data Used			Hyper parameters	Train data					Test data				
		Outliers	Smote	Scaling		Accuracy	Precision	Recall	F1	AUC	Accuracy	Precision	Recall	F1	AUC
LR_model1	Logistic Regression	Treated	No	No	Default base model	0.89	0.77	0.51	0.61	0.88	0.89	0.78	0.5	0.61	0.87
LDA_model1	LDA	Treated	No	No	Gridsearch CV, best model for f1 score	0.89	0.77	0.47	0.59	0.88	0.88	0.77	0.45	0.57	0.86
SVM_model2	SVM	Treated	No	Yes	Gridsearch CV, best model for f1 score	0.99	0.93	1	0.96	1	0.97	0.87	0.93	0.9	0.98
ANN_model3 alpha = 0.05	ANN	Treated	No	Yes	Default base model with alpha = 0.05	0.99	0.98	0.98	0.98	1	0.97	0.94	0.9	0.92	0.99
RF_model2	RandomForest	No	No	No	Default base model with outliers	1	1	1	1	1	0.97	0.98	0.86	0.92	0.99
ADA_model2	Adaboost	Treated	No	No	GridSearchCV, best model for f1 score	0.9	0.75	0.6	0.67	0.92	0.9	0.76	0.6	0.67	0.91
GB_model2	**Gradient Boost**	**Treated**	**No**	**No**	**GridSearchCV, best model for f1 score**	**1**	**1**	**1**	**1**	**1**	**0.99**	**0.99**	**0.94**	**1**	**1**

Table 7-1 A comparison of best model from all algorithms

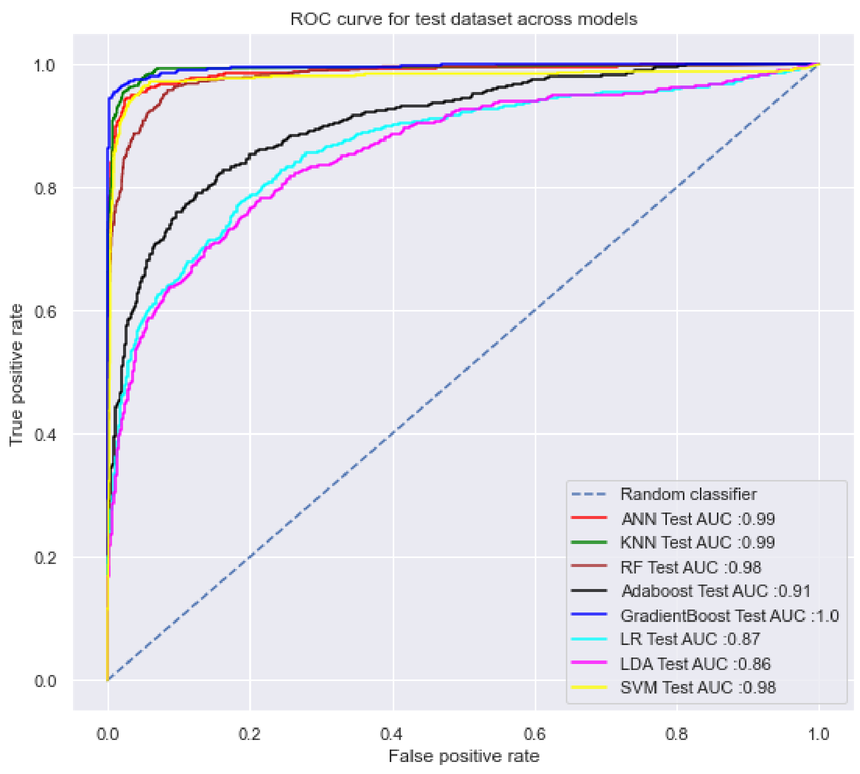


Figure 7-1 ROC curve for test dataset across all models

7.2. Why Gradient boost is the best model?

- The hyperparameter tuned Gradient Boost model (GB_model2) has given the best performance in terms of test data precision, recall and f1-score which are the primary evaluation metrics. The Accuracy and AUC are also highest amongst all the models. As the model looked like it overfit on train dataset, a further 5-fold and 10-fold cross-validation was done on complete dataset in which the F1-score on all folds was either comparable or greater than test data f1-score (the test data performance held true or was better for all folds)
- The difference between train data metrics and test data metrics is within 10%
- The model is interpretable. Sklearn provided feature importance for the model

7.3. How can business use these metrics?

- A precision of 0.99 implies that out of 100 customers that the model has identified as churned, 99 would actually churn and 1 would not. Any marketing budget allocated for a targeted campaign for retention of these customers would be most optimally utilized as only 1/100 customers would be incorrectly identified as churn.
- A recall of 0.94 implies that for 100 customers who actually churn, the model would have identified 94 as churn and 6 as not-churn. This would mean that the campaign would target these 94 customers and there is scope for retaining these customers and missing out on 6 customers.
- Based on the customer base for which prediction needs to be done, business can use the above to project the numbers of customers who would churn and how many the model would identify and miss.
- That could be further used to come up with per customer budget (if total budget for retention campaign is known) so that appropriate offers can be designed for each customer.
- If per customer budget is known, cost projections for retention campaign can be calculated.
- If limited budget is available and not all customers can be covered, the model can also provide the probability of churning so that high probability customers can be targeted first. Also, a different perspective to this problem would be to do a segmentation and target high value customer segment.

7.4. Model interpretation from best models

Gradient boost model has given very good performance after tuning. The model is robust (no overfit or underfit).

The figures below show the feature importance given by the Gradient Boost model and two other top scoring models (F1-score) – Random Forest and ANN. A comparison of top 6 features that contributed to these models is provided in a table.

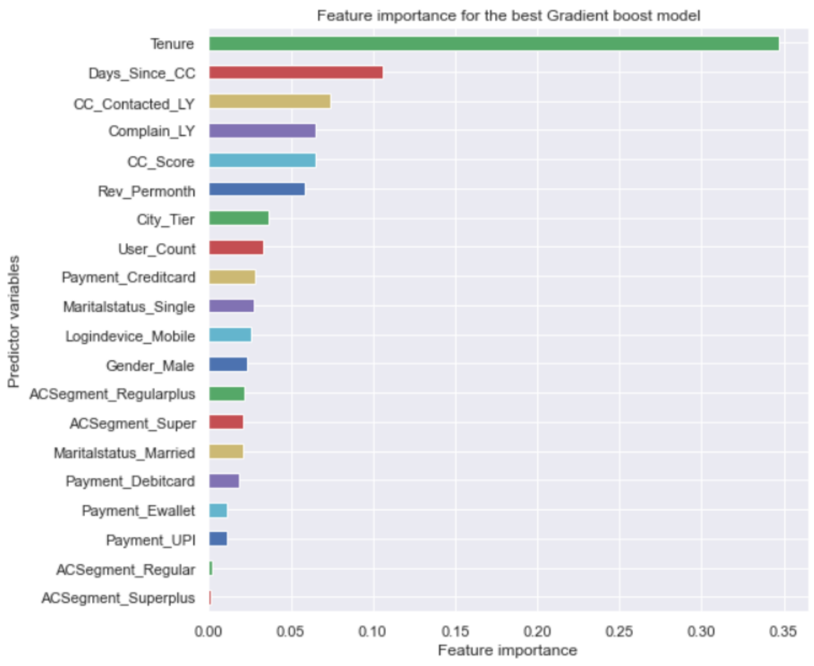


Figure 7-2 Feature importance for Gradient Boost model

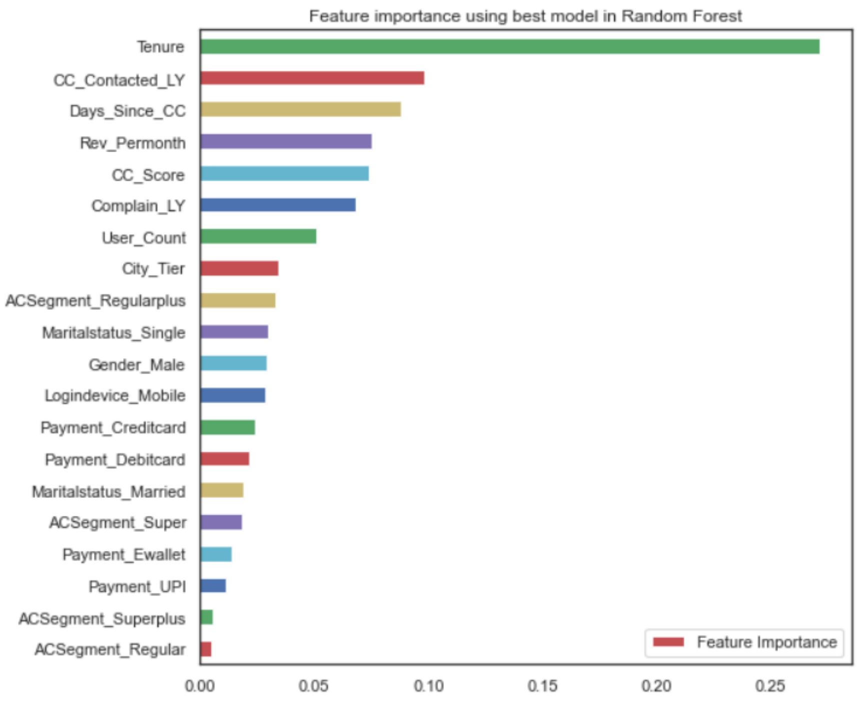


Figure 7-3 Feature importance for Random Forest model

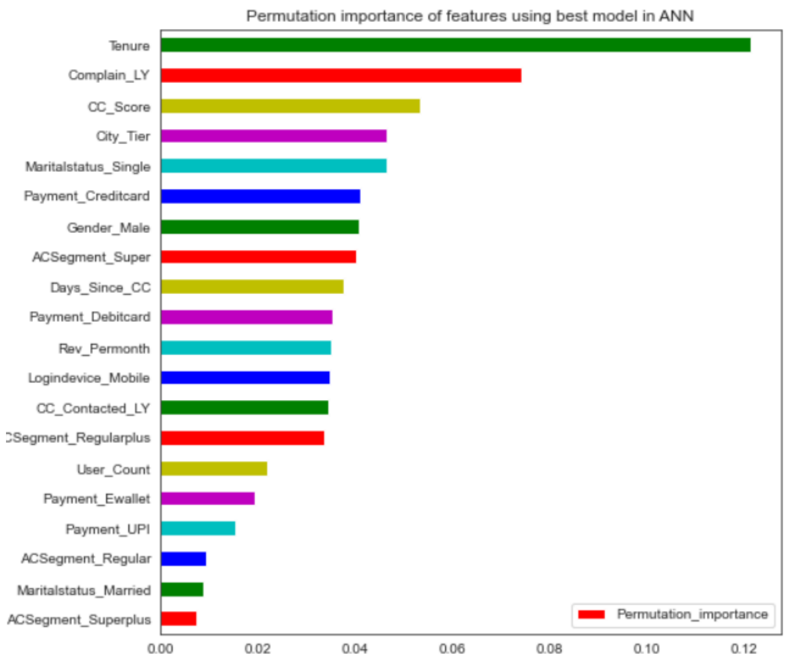


*Figure 7-4 Feature importance for ANN model**

* ANN is a black box model and feature importance is not directly available. Hence Sklearn's permutation feature importance function was used with this model. The permutation feature importance is defined to be the decrease in a model score when a single feature value is randomly shuffled. This procedure breaks the relationship between the feature and the target, thus the drop in the model score is indicative of how much the model depends on the feature.

	Gradient Boost	**Random Forest**	**ANN**
Feature1	Tenure	Tenure	Tenure
Feature2	Days since customer care last contacted	Number of times customer care was contacted last year	Was complaint made last year
Feature3	Number of times customer care was contacted last year	Days since customer care last contacted	Customer care agent score
Feature4	Was complaint made last year	Revenue per month	City tier
Feature5	Customer care agent score	Customer care agent score	Marital status single
Feature6	Revenue per month	Was complaint made last year	Payment credit card

Table 7-2 Comparison of feature importance of top 3 models

The top 5 features that have influenced Gradient boost model are Tenure, Days since last customer connect, number of times customer contacted last year, complaint last year and customer care score. Together, they add up to almost 66% of the total feature importance.

If we look at the top3 models, the top most contributor remains the same. Was complaint made last year and customer care agent score also figure in Top6 features across these 3 models.

Top 5 features from Gradient Boost

- From EDA, we can observe that for lower tenures especially within the first month, the churn is higher. Hence, once a customer has been acquired, the first two months is very important to keep the customer satisfied.
- Churned customers had contacted customer care more recently before churning than active customers. Per EDA, median days since last customer connect is higher for active customers. Churned customers had contacted customer care recently before churning.

- The next important parameter to predict customer churn per this model is number of times customer care was contacted by the customer. Per EDA, the median and third quantile of number of times customer care was contacted previous year is higher for churned customers compared to active/current customers.
- According to ranking of important features, except tenure, the other features in top 5 are related to customer care or complaints. This may be indicative of need to monitor and improve the customer care processes.

8. Business recommendations

8.1. High churn rate in low tenure customers

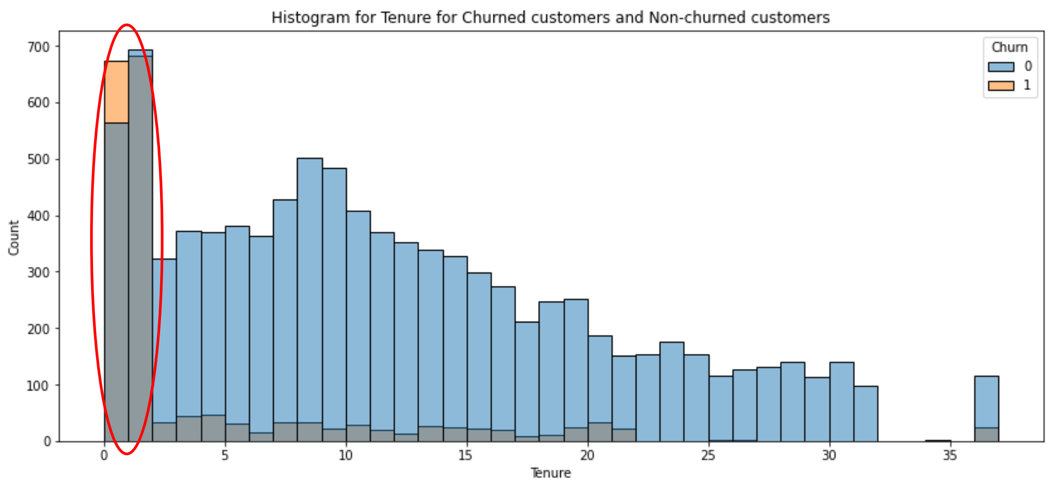


Figure 8-1 Tenure and Churn

<u>Insight:</u> Churn is highest between the tenure period 0 to 2

Possible Reasons: Bad first experience or Trial periods/prepaid accounts that expire automatically if no top-up is done within a predefined period. Important to determine between the above two reasons. Based on high customer care calls, complaints registered and low cashback and coupons for low tenure customers, it points to the first reason.

<u>Business recommendations:</u>

- **<u>Activation</u>**/Onboarding team could **<u>extend support</u>** beyond the initial setup until customers settle down with the service
- Activation team **proactively engages customers** for the first month or two
- Customer care take a **<u>feedback survey</u>** about the process so that any hiccups can be understood and sorted out
- To increase response rates for feedback, gift cards/coupons can be given

8.2. Existing retention programs – Cashback and Coupons

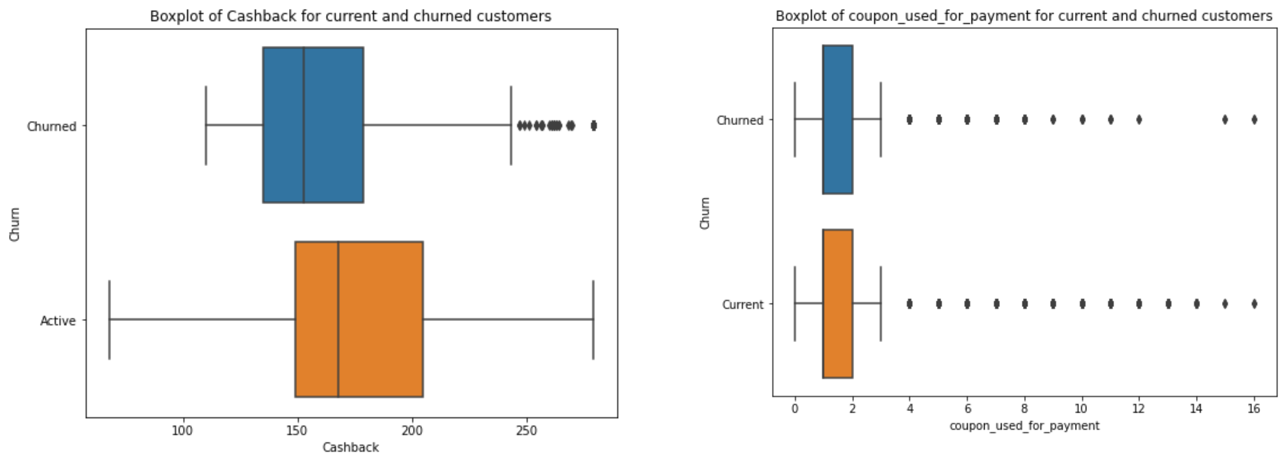


Figure 8-2 Churn and existing retention programs

- The churned customers as shown in first boxplot have lesser cashback

- The churned and active customers have almost used the same number of coupons for payment

Insight: The current retention programs do not seem to be focusing on the customers with higher risk of churn

Recommendation: Review whether existing cashback and coupon programs are still relevant given the current churn model. If they are not relevant, design new retention programs to address current high risk customer group.

8.3. Churn and Customer care service

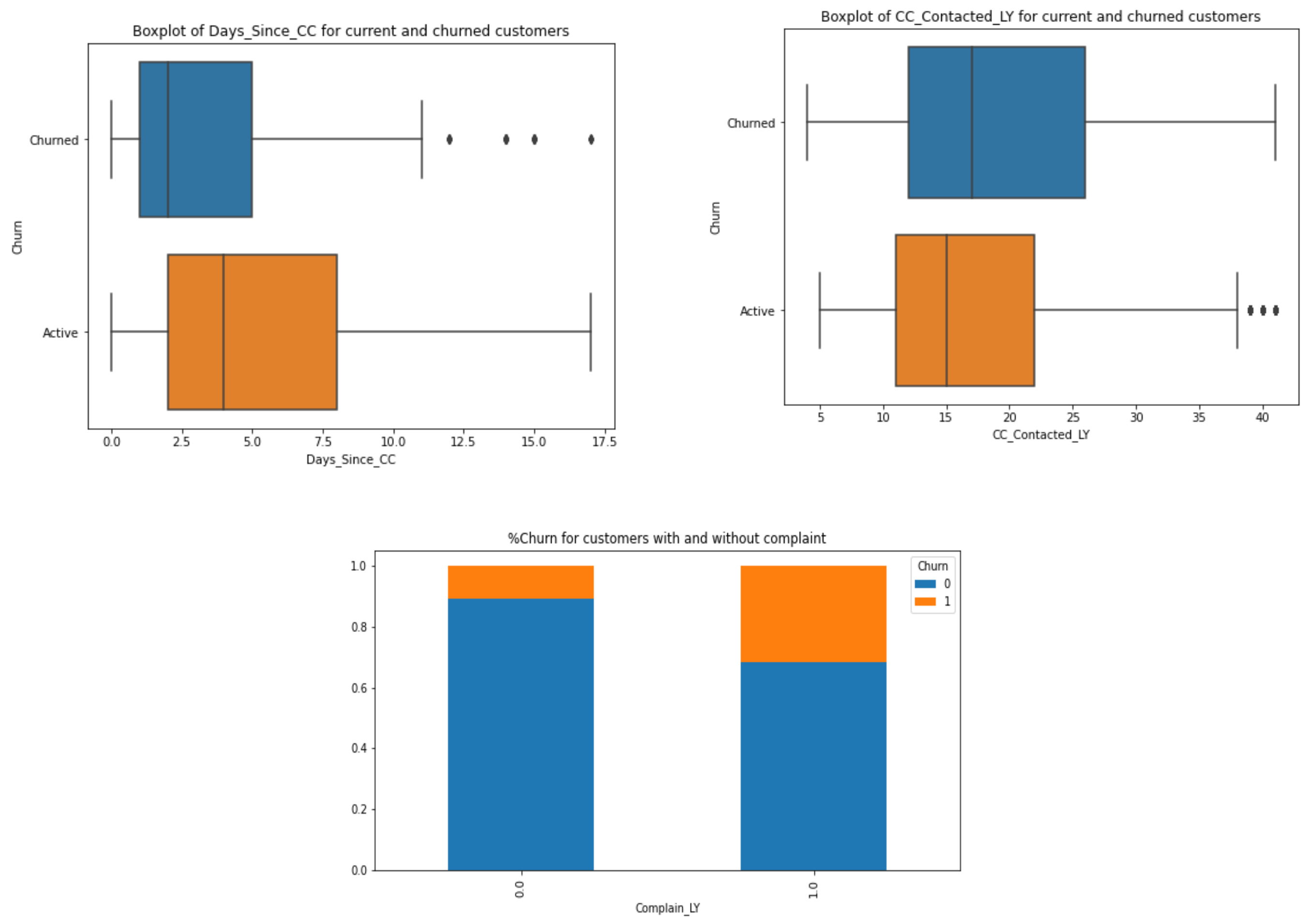


Figure 8-3 Churn and Customer care service

Churned customers seem to have contacted customer care more recently before churning

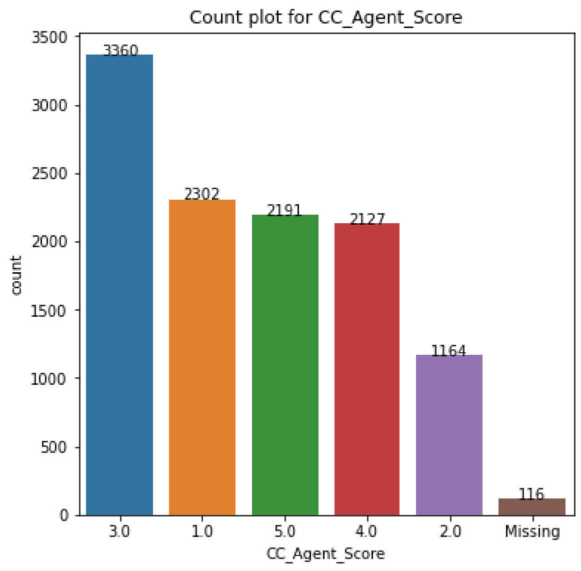
- The number of times churned customers contacted customer care in the year is higher than number of times active customers contacted customer care
- 31% of customers who registered complaint churned Vs 11% of customers who have not registered complaint in last year

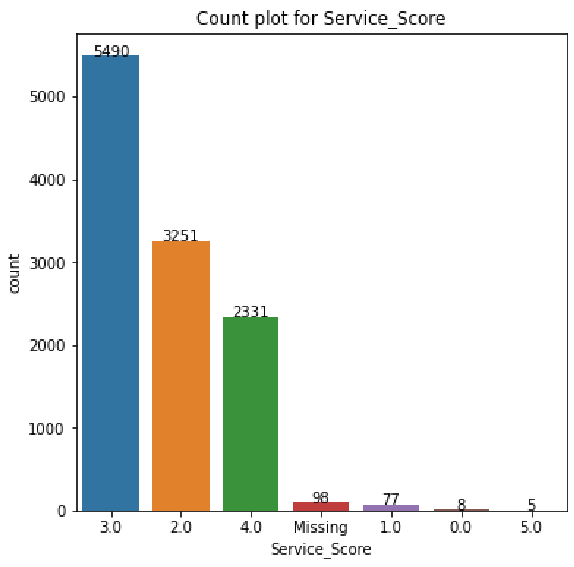
<u>Insight:</u> These indicate behavioral changes in customer before churn happens

<u>Recommendation:</u> Analyze Complaints & Customer care contact reasons

- Perform Root cause analysis, identify and fix top reasons
- Establish Service level agreements (if not already present)

8.4. Customer care & Service – Customer perspective





Insight: 78% of customers have rated service as 3 or less than 3

Insight: 61% of customers have rated customer care agents a score of 3 or less than 3

Recommendation: Analyze customer feedback. Perform Sentiment analysis of the feedback (if any) that went along with scores. Identify top reasons that have resulted in low scores; if subjective feedback not captured, capture that as well

8.5. Revenue per month and Churn

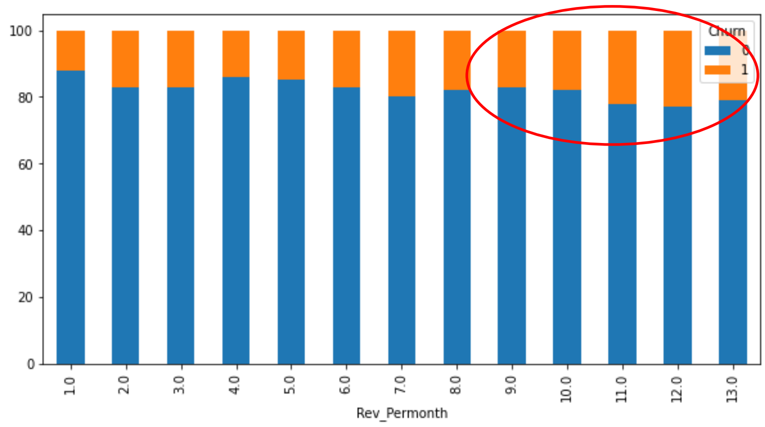
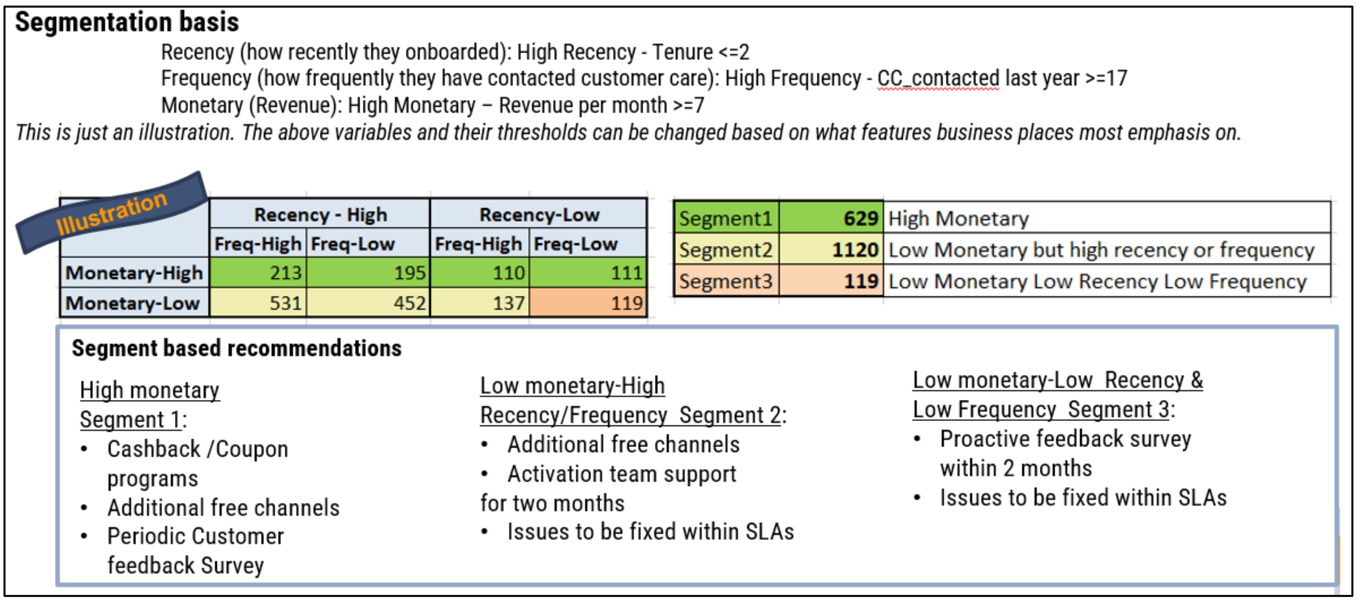


Figure 8-4 High churn in high revenue customers

The % churn of customers in higher revenue group, for revenue >=7 per month is higher than low revenue group customers

***Insight:* More proportion of high revenue customers are leaving compared to less revenue customers**

Recommendation: Create segmented offers for high revenue customers. An illustrative segmentation based on RFM (Recency – Frequency – Monetary) has been shown below along with sample targeted recommendations for each segment. This illustration is based on the test data. A similar segmentation can be done in discussion with client based on what metrics they feel are important to segment based on.



Segmentation basis

Recency (how recently they onboarded): High Recency - Tenure <=2

Frequency (how frequently they have contacted customer care): High Frequency - CC_contacted last year >=17

Monetary (Revenue): High Monetary – Revenue per month >=7

This is just an illustration. The above variables and their thresholds can be changed based on what features business places most emphasis on.

	Recency - High		Recency-Low	
	Freq-High	Freq-Low	Freq-High	Freq-Low
Monetary-High	213	195	110	111
Monetary-Low	531	452	137	119

Segment1	629	High Monetary
Segment2	1120	Low Monetary but high recency or frequency
Segment3	119	Low Monetary Low Recency Low Frequency

Segment based recommendations

High monetary Segment 1:
- Cashback /Coupon programs
- Additional free channels
- Periodic Customer feedback Survey

Low monetary-High Recency/Frequency Segment 2:
- Additional free channels
- Activation team support for two months
- Issues to be fixed within SLAs

Low monetary-Low Recency & Low Frequency Segment 3:
- Proactive feedback survey within 2 months
- Issues to be fixed within SLAs

Figure 8-5 Segmented offers illustration

9. Appendix

9.1. Annexure A: Tuning done for Gradient Boost algorithm

```
param_grid1 = {
   'loss': ['deviance', 'exponential'],
   'learning_rate': [0.1,0.5],
#   'n_estimators': [51,101,151],
#   'criterion': ['friedman_mse', 'mse', 'mae'],
   'min_samples_split': [20,60,100],
   'min_samples_leaf': [2,6,10],
   'max_depth':[3,6,9],
   'max_features':[7,10]
}
## Best parameters for above grid is learning_rate=0.5, max_depth=9, max_features=10,min_samples_leaf=6, min_samples_split=20
## loss = deviance.  Above is for 100 estimators and friedman_mse which are default. Best f1_score = 0.917

param_grid2 = {
   'loss': ['deviance'],  # 'exponential'
   'learning_rate': [0.5],
   'n_estimators': [101],
   'criterion': ['mse'], #
   'min_samples_split': [20],
   'min_samples_leaf': [6],
   'max_depth':[9],
   'max_features':[10]
}  # Trying different criterion ('mse') for same best grid settings as grid1
## For mse, score has slightly improved to 0.9124. Now let us try for mae as well.

param_grid3 = {
   'loss': ['deviance'],
   'learning_rate': [0.5],
   'n_estimators': [101],
   'criterion': ['mae'], #
   'min_samples_split': [20],
   'min_samples_leaf': [6],
   'max_depth':[9],
   'max_features':[10]
}  # Trying different criterion ('mse') for same best grid settings as grid1
## For mse, score has slightly improved to 0.9124. Now let us try for mae as well.
## mae keeps running forever. So, we'll stick to mse and vary other parameters of the grid to improve f1-score


param_grid4 = {
   'loss': ['deviance'],
   'learning_rate': [0.5],
   'n_estimators': [101,151,201],
   'criterion': ['mse'],
   'min_samples_split': [20],
   'min_samples_leaf': [6],
   'max_depth':[9],
   'max_features':[10]
}  # increasing just estimators for same best grid settings as grid1 with mse
## f1_score = 0.92. Increases with increase in estimators. Let's try for higher number of estimators in next round.

param_grid5 = {
   'loss': ['deviance'],
   'learning_rate': [0.5],
   'n_estimators': [201,401,601],
   'criterion': ['mse'],
   'min_samples_split': [20],
   'min_samples_leaf': [6],
   'max_depth':[9],
   'max_features':[10]
}  # estimators settled at 201 with almost same f1_score of 0.9205. Fixing it at 201 and varying other features in next round
```

```
param_grid6 = {
    'loss': ['deviance'],
    'learning_rate': [0.5],
    'n_estimators': [201],
    'criterion': ['mse'],
    'min_samples_split': [20],
    'min_samples_leaf': [6],
    'max_depth':[9,12,15],
    'max_features':[10,11,12]
}  # estimators settled at 201 with almost same f1_score of 0.92. Fixing it at 201 and varying other parameters in next round
# Result: GradientBoostingClassifier(criterion='mse', learning_rate=0.5, max_depth=9,max_features=10, min_samples_leaf=6,
#                          min_samples_split=20, n_estimators=201,random_state=0)
# f1_Score: Not improved - same at 0.9205
# max_depth and max_features have settled at 9 and 10 as in previous parameters
# We'll slightly reduce min_samples_split and min_samples_leaf in next round


param_grid7 = {
    'loss': ['deviance'],
    'learning_rate': [0.5],
    'n_estimators': [201],
    'criterion': ['mse'],
    'min_samples_split': [20,15],
    'min_samples_leaf': [4,6],
    'max_depth':[9,11],
    'max_features':[10,11]
}
# Result: GradientBoostingClassifier(criterion='mse', learning_rate=0.5, max_depth=9,max_features=11, min_samples_leaf=6,
#                          min_samples_split=15, n_estimators=201,random_state=0)

# f1_Score: 0.9215
# In the next iteration, we will fix all values as per the best parameters above and only tune learning rate

param_grid8 = {
    'loss': ['deviance'],
    'learning_rate': [0.1, 0.5, 1],
    'n_estimators': [201],
    'criterion': ['mse'],
    'min_samples_split': [15],
    'min_samples_leaf': [6],
    'max_depth':[9],
    'max_features':[11]
}
# Result: GradientBoostingClassifier(criterion='mse', learning_rate=0.5, max_depth=9,max_features=11, min_samples_leaf=6,
#                          min_samples_split=15, n_estimators=201,random_state=0)

# f1_Score: 0.9215
# Learning rate settled at middle value of 0.5. We'll finalize this parameter set and evaluate on train and test
```

9.2. Annexure B: Model building and tuning for all 8 models

9.2.1. Logistic Regression

Logistic regression model performs well when the data is linearly separable. It assumes that there is linearity between target and predictor variables. It is a parametric model hence it can be fast compared to KNN. It also provides coefficients that helps with model interpretability. Hence the first model tried was Logistic regression. Both SKLearn and Statsmodel (python libraries) implementations were tried for Logistic regression.

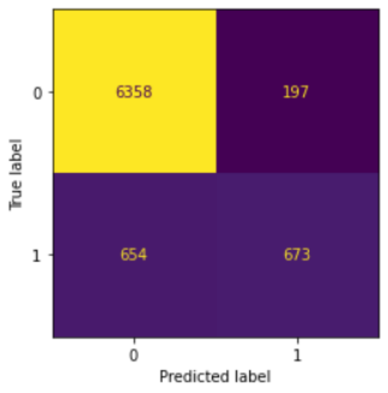
Data used: The features that were skipped prior to trying out Logistic regression algorithm are listed in Methodology section (Section 6.2). Other features were dropped during the model tuning phase and corresponding performances have been noted in the performance table in model tuning section.

Logistic regression is outlier sensitive; hence only outlier treated data has been used for all the models.

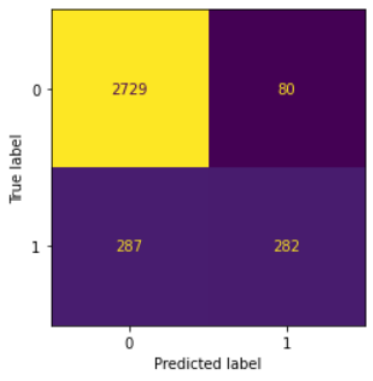
- SKlearn's Logistic Regression was used alongside RFE (Recursive feature elimination) to determine number of features that can give best performance as well as the ranking of features that Logistic regression provides.
- Statsmodel implementation was tried to obtain p-values to determine what features need to be retained in the model and to use the coefficients to understand the relationship of predictor and target variables.

9.2.1.1. SKLearn Base model with default hyperparameters (Also the best model)

Train dataset Confusion Matrix



Test dataset Confusion Matrix

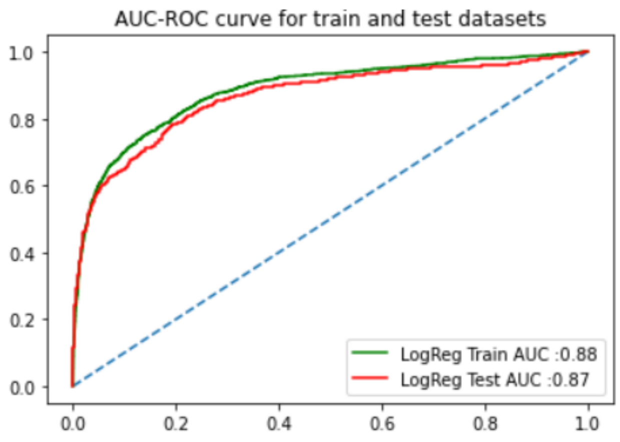


Train dataset Classification report

```
              precision    recall  f1-score   support

           0       0.91      0.97      0.94      6555
           1       0.77      0.51      0.61      1327

    accuracy                           0.89      7882
   macro avg       0.84      0.74      0.77      7882
weighted avg       0.88      0.89      0.88      7882
```

Test dataset Classification report

```
              precision    recall  f1-score   support

           0       0.90      0.97      0.94      2809
           1       0.78      0.50      0.61       569

    accuracy                           0.89      3378
   macro avg       0.84      0.73      0.77      3378
weighted avg       0.88      0.89      0.88      3378
```



It is to be noted that this is also the best model out of all Logistic Regression models

Table 7-9-1 Logistic regression : Base model and best model performance

9.2.1.2. Statsmodel Logistic regression - model tuning

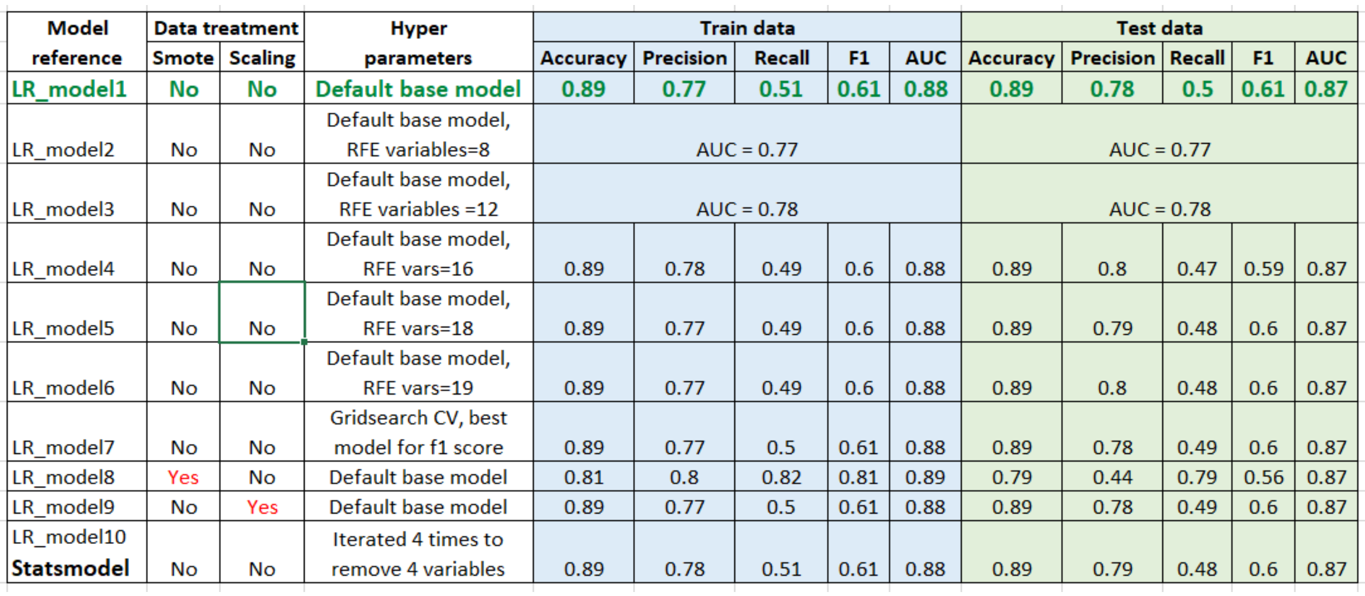
- Statsmodel provides a model summary when a model is fit on the train data. The p-value of the predictor variables in the summary was used for deciding the significance of each predictor variable to the model.
- One by one the predictor variables whose p-value was > 0.05 were removed from the model and model rebuilt. Over multiple iterations (as shown below), the variables were eliminated one by one such that only variables with p-value < 0.05 remained in the model. 16 variables were significant at a level of 0.05.

  ## Iteration 1: ACSegment_Superplus removed
  ## Iteration 2: Payment_Ewallet removed
  ## Iteration 3: ACSegment_Regularplus removed
  ## Iteration 4: Maritalstatus_Married removed

The F1-score of the final model built was close but not greater than the SkLearn's best model (described in 7.1.1). Hence this was not selected as the best model in logistic regression.

9.2.1.3. SKLearn Logistic regression - model tuning

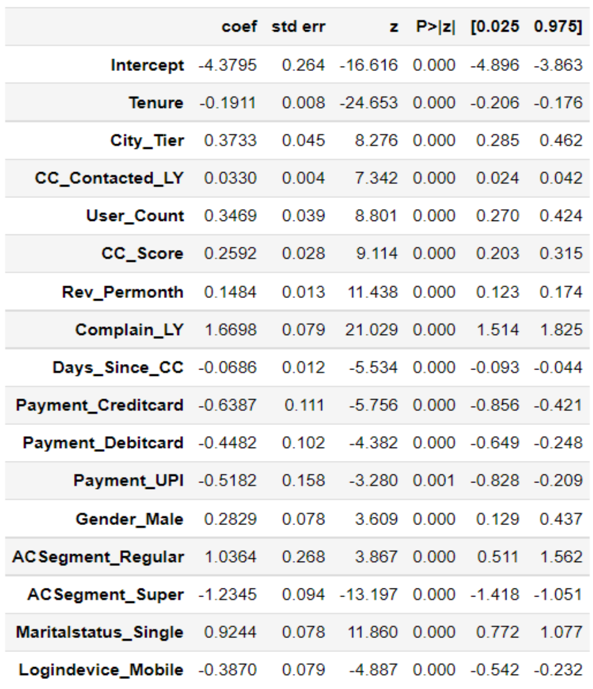
- The base model (Section 7.1.1) was further tuned by using RFE to change the number of predictors used. It was found that reducing variables did not yield better results. Hence all 20 predictors were retained.
- GridSearchCV function from Sklearn library was used to tune the hyperparameters. It was found that tuned hyperparameters did not perform better than the base model.
- Further, Scaled data and Smote resampled data was used with the base model. It was found that Smote resulted in overfitting precision of class 1. Scaling also did not improve performance compared to non-scaled data.
- The results of all the trials have been given in the table below
- The best model is the same as the base model mentioned in 7.1.1 which is LR_model1 (reference to the python code) highlighted in green in the table below



Model reference	Data treatment		Hyper parameters	Train data					Test data				
	Smote	Scaling		Accuracy	Precision	Recall	F1	AUC	Accuracy	Precision	Recall	F1	AUC
LR_model1	**No**	**No**	**Default base model**	**0.89**	**0.77**	**0.51**	**0.61**	**0.88**	**0.89**	**0.78**	**0.5**	**0.61**	**0.87**
LR_model2	No	No	Default base model, RFE variables=8	AUC = 0.77					AUC = 0.77				
LR_model3	No	No	Default base model, RFE variables =12	AUC = 0.78					AUC = 0.78				
LR_model4	No	No	Default base model, RFE vars=16	0.89	0.78	0.49	0.6	0.88	0.89	0.8	0.47	0.59	0.87
LR_model5	No	No	Default base model, RFE vars=18	0.89	0.77	0.49	0.6	0.88	0.89	0.79	0.48	0.6	0.87
LR_model6	No	No	Default base model, RFE vars=19	0.89	0.77	0.49	0.6	0.88	0.89	0.8	0.48	0.6	0.87
LR_model7	No	No	Gridsearch CV, best model for f1 score	0.89	0.77	0.5	0.61	0.88	0.89	0.78	0.49	0.6	0.87
LR_model8	Yes	No	Default base model	0.81	0.8	0.82	0.81	0.89	0.79	0.44	0.79	0.56	0.87
LR_model9	No	Yes	Default base model	0.89	0.77	0.5	0.61	0.88	0.89	0.78	0.49	0.6	0.87
LR_model10 **Statsmodel**	No	No	Iterated 4 times to remove 4 variables	0.89	0.78	0.51	0.61	0.88	0.89	0.79	0.48	0.6	0.87

Table 7-9-2 Model tuning - Logistic regression models performance

9.2.1.4. Model interpretation



	coef	std err	z	P>|z|	[0.025	0.975]
Intercept	-4.3795	0.264	-16.616	0.000	-4.896	-3.863
Tenure	-0.1911	0.008	-24.653	0.000	-0.206	-0.176
City_Tier	0.3733	0.045	8.276	0.000	0.285	0.462
CC_Contacted_LY	0.0330	0.004	7.342	0.000	0.024	0.042
User_Count	0.3469	0.039	8.801	0.000	0.270	0.424
CC_Score	0.2592	0.028	9.114	0.000	0.203	0.315
Rev_Permonth	0.1484	0.013	11.438	0.000	0.123	0.174
Complain_LY	1.6698	0.079	21.029	0.000	1.514	1.825
Days_Since_CC	-0.0686	0.012	-5.534	0.000	-0.093	-0.044
Payment_Creditcard	-0.6387	0.111	-5.756	0.000	-0.856	-0.421
Payment_Debitcard	-0.4482	0.102	-4.382	0.000	-0.649	-0.248
Payment_UPI	-0.5182	0.158	-3.280	0.001	-0.828	-0.209
Gender_Male	0.2829	0.078	3.609	0.000	0.129	0.437
ACSegment_Regular	1.0364	0.268	3.867	0.000	0.511	1.562
ACSegment_Super	-1.2345	0.094	-13.197	0.000	-1.418	-1.051
Maritalstatus_Single	0.9244	0.078	11.860	0.000	0.772	1.077
Logindevice_Mobile	-0.3870	0.079	-4.887	0.000	-0.542	-0.232

Table 7-9-3 Statsmodel Logistic Regression coefficients

This model's performance with respect to precision and recall of minority class is not a good one. A recall of 0.50 means out of 100 churning customers, model can only identify 50 correctly. The precision is not very high either. So other models may have to be looked at for this problem. Since the model has not performed well, the coefficients are only an academic exercise and business cannot rely on that for insights.

The following variables have a positive correlation with Churn: i.e., as the variable increases, churn increases: City tier, User count, Customer care score, Revenue per month, Complaint last year, Regular customer segment and Single marital status.

The following variables have a negative correlation with Churn, i.e, as the variable increases, churn decreases: Tenure, Credit/debit/UPI payment, Super segment, Mobile login device

Unlike linear regression models, the coefficients become non-intuitive to interpret in logistic regression models. For instance, the logistic regression coefficient for City_Tier means: as City_tier goes up by 1, the log odds of churning go up by 0.37. It has been broken down for the top 8 influential variables below:

From the table above, the logistic regression equation for this model is as follows:

logit(p) or log(p/1-p) or log(P{Y=1}/P{Y=0}) or log-odds ratio =
*-4.38 – 0.19*Tenure + 0.37*City_tier + 0.03*CC_contacted_LY + 0.35*User_Count + 0.26*CC_Score + 0.15*Rev_Permonth + 1.67*Complain_LY – 0.07*Days_Since_CC – 0.64*Payment_Creditcard – 0.45*Payment_Debitcard – 0.52*Payment_UPI + 0.28*Gender_Male + 1.04*ACSegment_Regular – 1.23*ACSegment_Super + 0.92*Maritalstatus_Single – 0.39*Logindevice_Mobile*

Coefficient	Predictor	Exponential of coefficient/ Odds of customer churning	Interpretation
1.67	Complain_LY	5.31	If complaint last year increases by 1, the odds of customer churning increases by 531%
-1.23	ACSegment_Super	3.42	If Account segment is Super (increases by 1), the odds of customer churning decreases by 342%
1.04	ACSegment_Regular	2.83	If Account segment is Regular (increases by 1),

			the odds of customer churning increases by 283%
0.92	Maritalstatus_Single	2.51	If Marital status is single (increases by 1), the odds of customer churning increases by 251%
-0.64	Payment_Creditcard	1.90	If payment is done by credit card (increases by 1), the odds of customer churning decreases by 190%
-0.52	Payment_UPI	1.68	If payment is done by UPI (increases by 1), the odds of customer churning decreases by 168%
-0.45	Payment_Debitcard	1.57	If payment is done by debit card (increases by 1), the odds of customer churning decreases by 157%
-0.39	Logindevice_Mobile	1.48	If login device is mobile (increases by 1), the odds of customer churning decreases by 148%

Table 9-4 Interpretation of logistic regression model coefficients for top 8 variables

As we can see from above, Tenure which from EDA seemed one of the significant separators of churn and non-churn customers does not figure in the top 8 influential variables. The model performance in terms of precision, recall and F1 score are not good compared to other models (non-linear and ensemble).

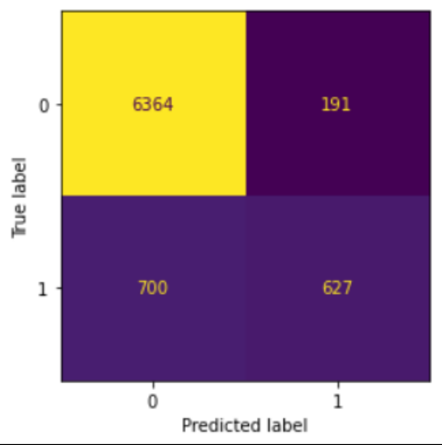
9.2.2. Linear Discriminant Analysis

Linear Discriminant Analysis (LDA) is also another machine learning classifier. It works well when there are linearly separable classes in data. It assumes that the underlying data has a gaussian distribution but can perform well even if assumptions are violated.

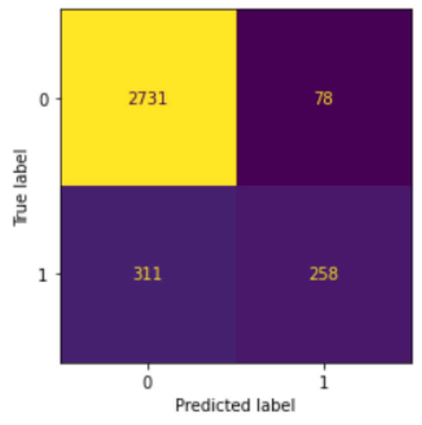
- SKlearn's Linear Discriminant Analysis function was used for modelling
- The base model was run with default hyperparameters and the performance metrics noted
- The model's hyperparameters were tuned using GridSearchCV function and model constructed using the best parameters selected by GridSearchCV. This did not provide much improvement over the base model except that f1-score improved by 0.01
- Data used was outlier treated unscaled dataset
- Model performance metrics for base model and tuned model have been provided below

9.2.2.1. Linear Discriminant Analysis model with default hyperparameters

Train dataset Confusion Matrix



Test dataset Confusion Matrix

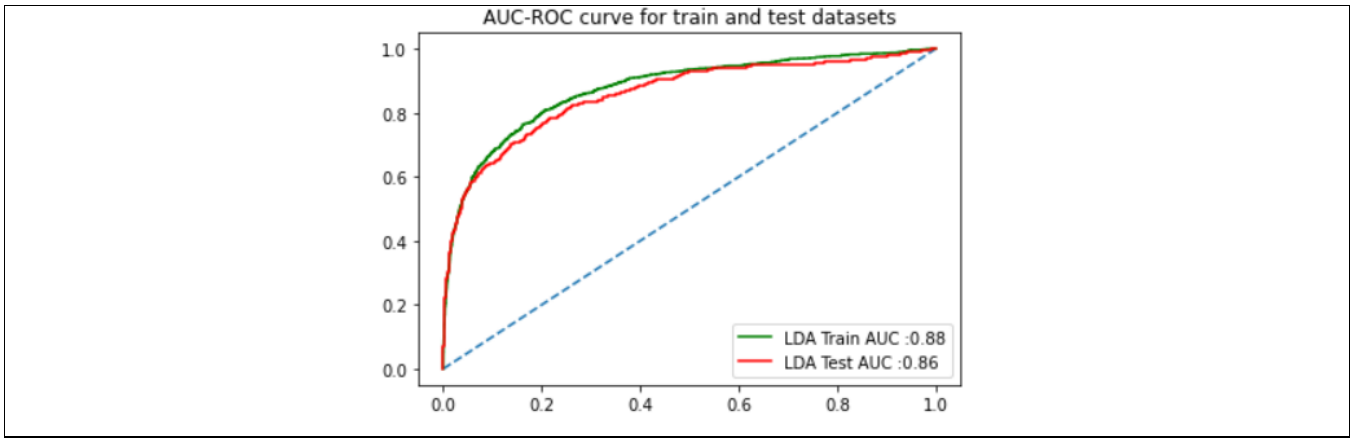


Train dataset Classification report

```
              precision    recall  f1-score   support

           0       0.90      0.97      0.93      6555
           1       0.77      0.47      0.58      1327

    accuracy                           0.89      7882
   macro avg       0.83      0.72      0.76      7882
weighted avg       0.88      0.89      0.88      7882
```

Test dataset Classification report

```
              precision    recall  f1-score   support

           0       0.90      0.97      0.93      2809
           1       0.77      0.45      0.57       569

    accuracy                           0.88      3378
   macro avg       0.83      0.71      0.75      3378
weighted avg       0.88      0.88      0.87      3378
```



9.2.2.2. SKLearn LDA - model tuning

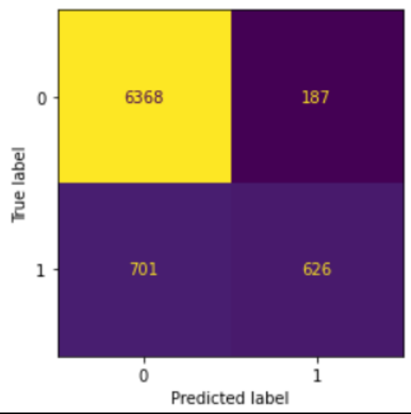
- GridSearchCV function from Sklearn library was used to tune the hyperparameters on the base model described in 7.2.1. It was found that tuned hyperparameters performed almost similar to the base model.
- The results of all the trials have been given in the table below
- The best model is highlighted in green in the table below and the metrics for this model is given in section 7.2.3 (hyperparameter tuned model)

Data Used			Hyper parameters	Train data					Test data				
Outliers	Smote	Scaling		Accuracy	Precision	Recall	F1	AUC	Accuracy	Precision	Recall	F1	AUC
Treated	No	No	Default base model	0.89	0.77	0.47	0.58	0.88	0.88	0.77	0.45	0.57	0.86
Treated	**No**	**No**	**Gridsearch CV, best model for f1 score**	**0.89**	**0.77**	**0.47**	**0.59**	**0.88**	**0.88**	**0.77**	**0.45**	**0.57**	**0.86**

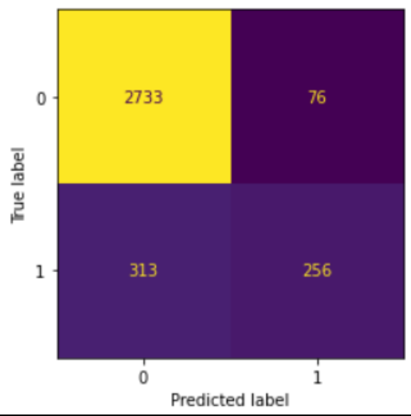
Table 9-5 Model tuning - Linear Discriminant Analysis models performance

9.2.2.3. Linear Discriminant Analysis model with Gridsearch tuned hyperparameters

Train dataset Confusion Matrix



Test dataset Confusion Matrix

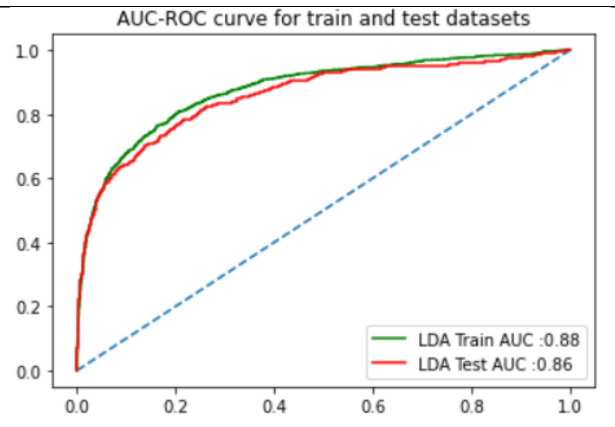


Train dataset Classification report

```
              precision    recall  f1-score   support

           0       0.90      0.97      0.93      6555
           1       0.77      0.47      0.59      1327

    accuracy                           0.89      7882
   macro avg       0.84      0.72      0.76      7882
weighted avg       0.88      0.89      0.88      7882
```

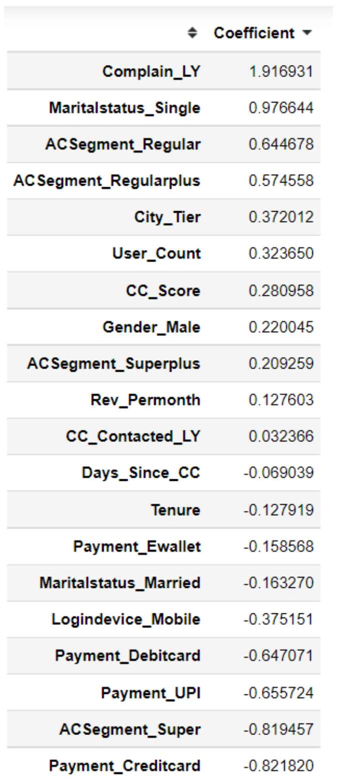
Test dataset Classification report

```
              precision    recall  f1-score   support

           0       0.90      0.97      0.93      2809
           1       0.77      0.45      0.57       569

    accuracy                           0.88      3378
   macro avg       0.83      0.71      0.75      3378
weighted avg       0.88      0.88      0.87      3378
```



9.2.2.4. Model interpretation

This model's performance with respect to precision and recall of minority class is not a good one. A recall of 0.45 means out of 100 churning customers, model can only identify 45 correctly. The precision is not very high either. So other models may have to be looked at for this exercise. Model coefficients for this model have been provided below:



	Coefficient
Complain_LY	1.916931
Maritalstatus_Single	0.976644
ACSegment_Regular	0.644678
ACSegment_Regularplus	0.574558
City_Tier	0.372012
User_Count	0.323650
CC_Score	0.280958
Gender_Male	0.220045
ACSegment_Superplus	0.209259
Rev_Permonth	0.127603
CC_Contacted_LY	0.032366
Days_Since_CC	-0.069039
Tenure	-0.127919
Payment_Ewallet	-0.158568
Maritalstatus_Married	-0.163270
Logindevice_Mobile	-0.375151
Payment_Debitcard	-0.647071
Payment_UPI	-0.655724
ACSegment_Super	-0.819457
Payment_Creditcard	-0.821820

Observations:

The best model out of all LDA models has given the following coefficients.

Compared to overall best model provided by Gradient boost, we can see that the features are not ranked in the same way. In fact, tenure seems to have a low-ranking coefficient although the model has established the inverse relationship it has with churn.

This is probably why LDA's performance is not comparable to other model's performance.

Table 9-6 Model coefficients for Linear Discriminant Analysis model

Complaints made by customers the previous year and Single customers are the top two positive coefficients. If there was a complaint the previous year and if the customer is single, that increases the possibility of churn. Similarly, Credit card payment and Super segment customers have the highest negative coefficients. This implies that customers paying using credit card and customers falling under super segment accounts churn lesser. This model has not performed well compared to other models. Hence the interpretation may not be very useful.

9.2.3. Support vector machines

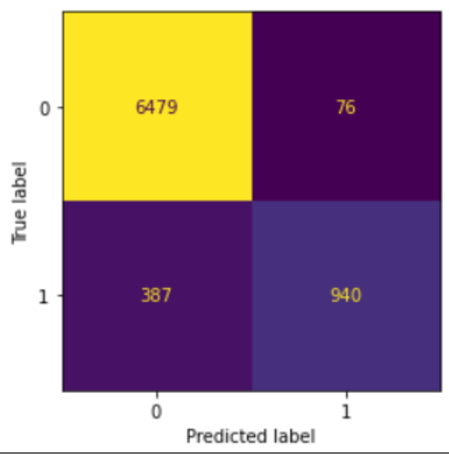
Support vector machine (SVM) is a popular machine learning algorithm that can be used for classification as well as regression. It works well when there are higher dimensions as well. It is very versatile as there are different kernel functions that can be specified to work well with the given data.

In this modelling exercise, SKlearn's Support Vector machine function was used for modelling:

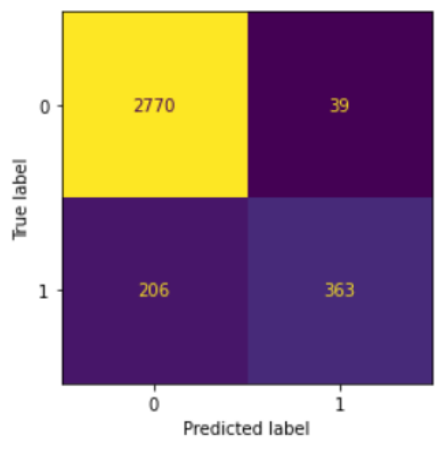
- The model requires scaled data so scaling was done using Sklearn's Standard Scaler.
- The base model was run with default hyperparameters with outlier treated unscaled dataset and the performance metrics noted. Tuning was later done using GridSearchCV
- Model performance metrics for base model and best model have been provided in the below sections

9.2.3.1. SVM base model with default hyperparameters

Train dataset Confusion Matrix



Test dataset Confusion Matrix

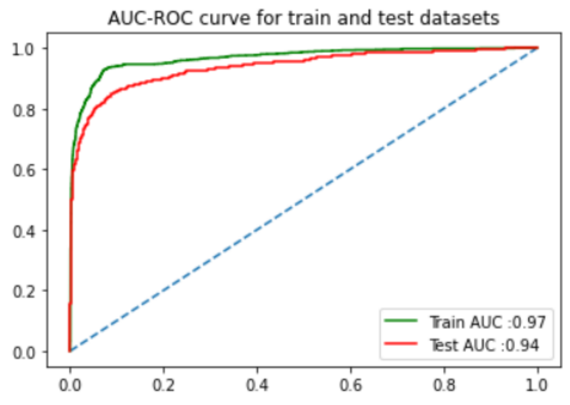


Train dataset Classification report

```
              precision    recall  f1-score   support

           0       0.94      0.99      0.97      6555
           1       0.93      0.71      0.80      1327

    accuracy                           0.94      7882
   macro avg       0.93      0.85      0.88      7882
weighted avg       0.94      0.94      0.94      7882
```

Test dataset Classification report

```
              precision    recall  f1-score   support

           0       0.93      0.99      0.96      2809
           1       0.90      0.64      0.75       569

    accuracy                           0.93      3378
   macro avg       0.92      0.81      0.85      3378
weighted avg       0.93      0.93      0.92      3378
```



9.2.3.2. SKLearn SVM - model tuning

- The model's hyperparameters were tuned using GridSearchCV function and model constructed using the best parameters selected by GridSearchCV. This provided significant improvement over the base model's performance simply by changing the kernel to 'poly'. A penalty parameter ('C') had to be included as the model was overfitting on training data.

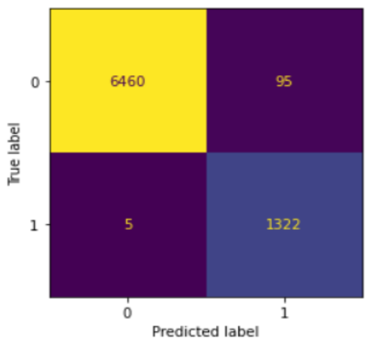
- The best grid model was also run with data that was not outlier treated. Although this was better than base model with default parameters, the previous model using outlier treated data had the best performance in this algorithm
  - The best model is highlighted in green in the table below and the metrics for this model is given in section 7.3.3 (hyperparameter tuned model)

Model		Data Used			Hyper	Train data					Test data				
reference	Algorithm	Outliers	Smote	Scaling	parameters	Accuracy	Precision	Recall	F1	AUC	Accuracy	Precision	Recall	F1	AUC
SVM_model1	SVM	Treated	No	Yes	Default base model	0.94	0.93	0.71	0.8	0.97	0.93	0.9	0.64	0.75	0.94
SVM_model2	**SVM**	**Treated**	**No**	**Yes**	**Gridsearch CV, best model for f1 score**	**0.99**	**0.93**	**1**	**0.96**	**1**	**0.97**	**0.87**	**0.93**	**0.9**	**0.98**
SVM_model3	SVM	Not treated	No	Yes	Gridsearch CV, best model for f1 score	0.98	0.89	0.99	0.94	1	0.95	0.83	0.92	0.87	0.97

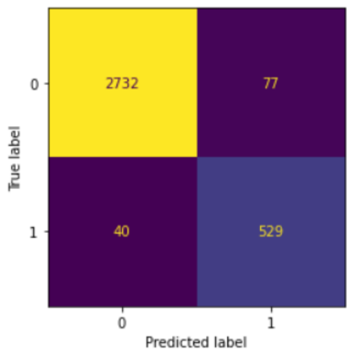
Table 9-7 Model tuning - SVM models performance

9.2.3.3. Best SVM model – with Gridsearch tuned hyperparameters

Train dataset Confusion Matrix



Test dataset Confusion Matrix

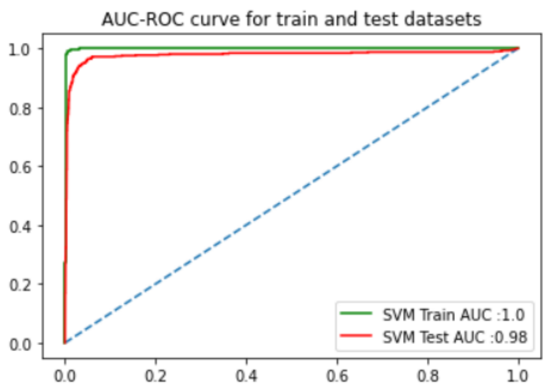


Train dataset Classification report

```
              precision    recall  f1-score   support

           0       1.00      0.99      0.99      6555
           1       0.93      1.00      0.96      1327

    accuracy                           0.99      7882
   macro avg       0.97      0.99      0.98      7882
weighted avg       0.99      0.99      0.99      7882
```

Test dataset Classification report

```
              precision    recall  f1-score   support

           0       0.99      0.97      0.98      2809
           1       0.87      0.93      0.90       569

    accuracy                           0.97      3378
   macro avg       0.93      0.95      0.94      3378
weighted avg       0.97      0.97      0.97      3378
```



9.2.3.4. Model interpretation

SVM has given a reasonably good performance on train and test datasets. There has been no overfitting or underfitting of the model as the train and test dataset performances have been comparable.

SVM's model coefficients attribute can be used only if linear kernel is used. Since 'poly' was used, SKlearn's permutation importance function was used to determine the features that are important to the model.

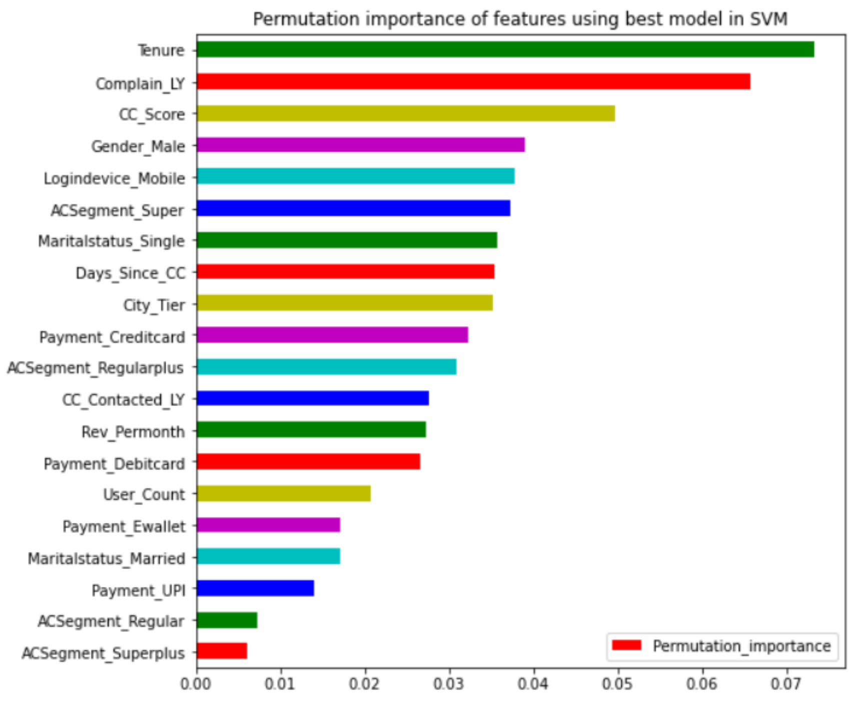


Figure 9-1 Feature importance from the best SVM model

In contrast to Logistic Regression and Linear Discriminant Analysis, this model has picked Tenure to be the most important feature followed by Complaints last year and Customer care score. From EDA, we can observe that for lower tenures especially within the first year, the churn is higher. Hence, once a customer has been acquired, the first year is very important to keep the customer satisfied. The next risk indicator of customer churn per this model is if a complaint has been registered in the previous year. From EDA, if a complaint was registered, the risk of churn is high. The customer satisfaction score is also important per the SVM model. This is consistent with the patterns and insights that emerged out of EDA. This is also consistent with the best performing model of this exercise, Gradient Boost.

9.2.4. Artificial Neural Network (ANN)

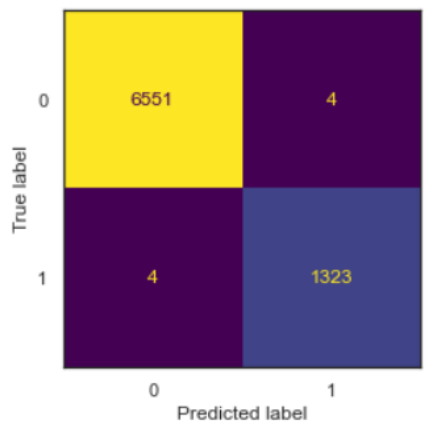
Artificial Neural Network (ANN) is a powerful machine learning algorithm that can be used for classification as well as regression. This algorithm can learn the complex patterns in underlying data. There is a tendency to overfit, but that can be controlled in the tuning exercise using the hyperparameters.

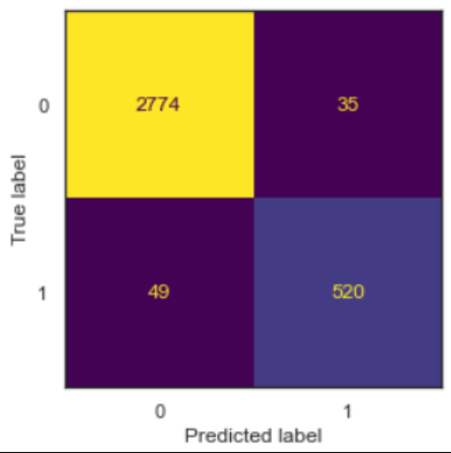
In this modelling exercise, SKlearn's Multilayer perceptron function was used for modelling:

- The model requires scaled data so scaling was done using Sklearn's Standard Scaler.
- The base model was run with default hyperparameters with outlier treated scaled dataset and the performance metrics noted. Tuning was later done using GridSearchCV
- Model performance metrics for base model and best model have been provided in the below sections

9.2.4.1. ANN base model with default hyperparameters

Train dataset Confusion Matrix	**Test dataset Confusion Matrix**



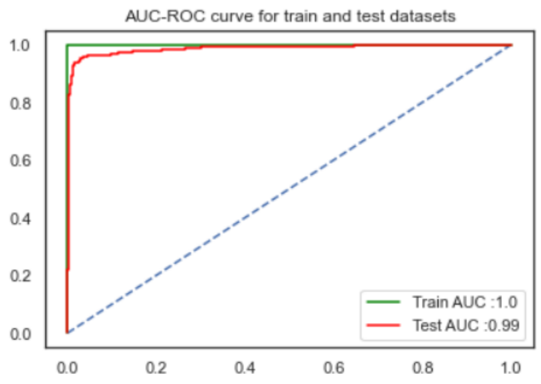


Train dataset Classification report

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00      6555
           1       1.00      1.00      1.00      1327

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

Test dataset Classification report

```
              precision    recall  f1-score   support

           0       0.98      0.99      0.99      2809
           1       0.94      0.91      0.93       569

    accuracy                           0.98      3378
   macro avg       0.96      0.95      0.96      3378
weighted avg       0.97      0.98      0.98      3378
```



9.2.4.2. SKLearn ANN - model tuning

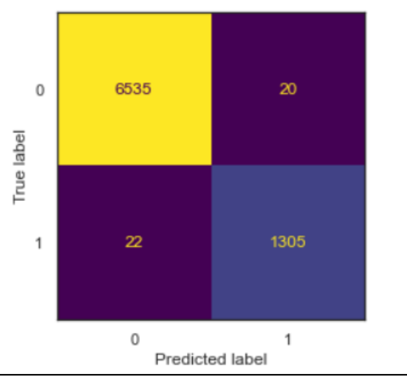
- The model's hyperparameters were tuned using GridSearchCV function and model constructed using the best parameters selected by GridSearchCV.
- As can be seen from the base model, the model had overfit on train dataset (all 1s). During tuning, higher alpha was provided in order to add a penalty that will regularize the model by reducing weights.
- Lower hidden layer sizes were also provided to GridsearchCV in order to reduce overfitting.
    - The best model is highlighted in green in the table below and the metrics for this model is given in section 7.4.3 (hyperparameter tuned model).
    - The base model has a 0.01 better recall than ANN_model3, but ANN_model3 has been chosen as the best model as the train and test performances are more comparable whereas in base model, the difference is more. Also, the train dataset in base model has overfit and completely learnt even the noise in the train dataset. Hence ANN_model3 which introduces a constraint through penalty is chosen to the best model here.

Model	Data Used			Hyper	Train data					Test data				
reference	Outliers	Smote	Scaling	parameters	Accuracy	Precision	Recall	F1	AUC	Accuracy	Precision	Recall	F1	AUC
ANN_model1	**Treated**	**No**	**Yes**	**Default base model**	**1**	**1**	**1**	**1**	**1**	**0.98**	**0.94**	**0.91**	**0.93**	**0.99**
ANN_model2	Treated	No	Yes	Gridsearch CV, best model for f1 score	0.99	0.98	0.97	0.97	1	0.97	0.92	0.89	0.91	0.99
ANN_model3 alpha = 0.2	Treated	No	Yes	Default base model with alpha = 0.2	0.97	0.95	0.89	0.92	0.99	0.95	0.92	0.8	0.85	0.97
ANN_model3 alpha = 0.1	Treated	No	Yes	Default base model with alpha = 0.1	0.99	0.98	0.95	0.97	1	0.97	0.94	0.85	0.9	0.98
ANN_model3 alpha = 0.05	**Treated**	**No**	**Yes**	**Default base model with alpha = 0.05**	**0.99**	**0.98**	**0.98**	**0.98**	**1**	**0.97**	**0.94**	**0.9**	**0.92**	**0.99**
ANN_model3 alpha = 0.04	Treated	No	Yes	Default base model with alpha = 0.04	1	0.99	0.98	0.99	1	0.97	0.94	0.9	0.92	0.99
ANN_model3 alpha = 0.03	Treated	No	Yes	Default base model with alpha = 0.03	1	0.99	0.98	0.99	1	0.97	0.95	0.86	0.91	0.99

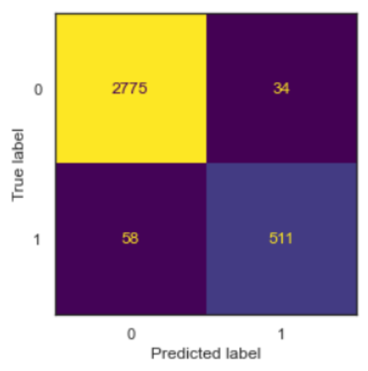
Table 9-8 Model tuning – ANN models performance

9.2.4.3. Best ANN model – with penalty term

Train dataset Confusion Matrix



Test dataset Confusion Matrix

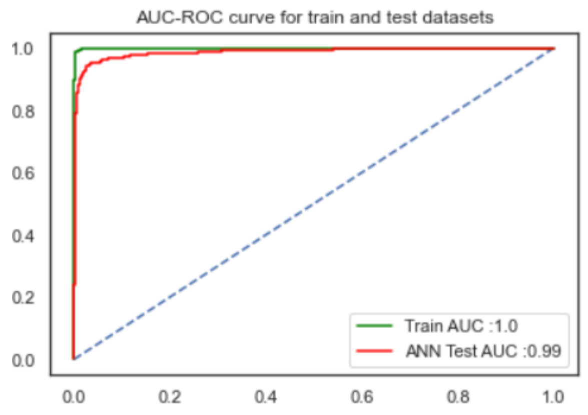


Train dataset Classification report

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00      6555
           1       0.98      0.98      0.98      1327

    accuracy                           0.99      7882
   macro avg       0.99      0.99      0.99      7882
weighted avg       0.99      0.99      0.99      7882
```

Test dataset Classification report

```
              precision    recall  f1-score   support

           0       0.98      0.99      0.98      2809
           1       0.94      0.90      0.92       569

    accuracy                           0.97      3378
   macro avg       0.96      0.94      0.95      3378
weighted avg       0.97      0.97      0.97      3378
```



9.2.4.4. Model interpretation

ANN is a black box model and hence the underlying feature importance cannot be found directly. A penalty constrained ANN has given a good performance on train and test datasets. There has been no overfitting or underfitting of the model as the train and test dataset performances have been comparable.

SKlearn's permutation importance function was used to determine the features that are important to the model.

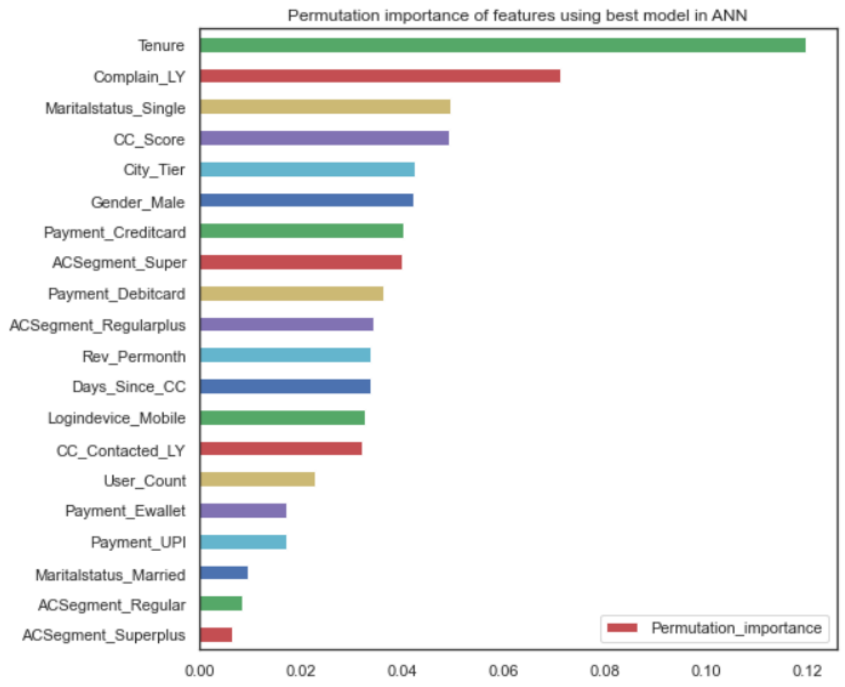


Figure 9-2 Feature importance from the best ANN model

Similar to SVM, this model has also picked Tenure to be the most important feature followed by Complaints last year. Out of the top 5 important features, SVM and ANN have 3 features in common with Tenure and Complain_LY being the same top 2 important parameters. This model has picked Tenure to be the most important feature followed by Complaints last year and Customer care score. From EDA, we can observe that for lower tenures especially within the first year, the churn is higher. Hence, once a customer has been acquired, the first year is very important to keep the customer satisfied. The next risk indicator of customer churn per this model is if a complaint has been registered in the previous year. From EDA, if a complaint was registered, the risk of churn is high. Also, single customers have more propensity to churn compared to married or divorced customers. ANN seems to have given more weightage to Tenure compared to SVM.

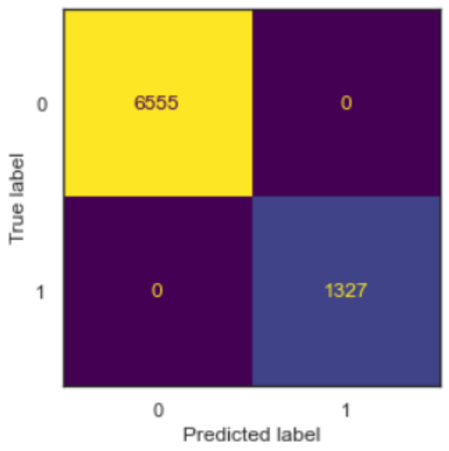
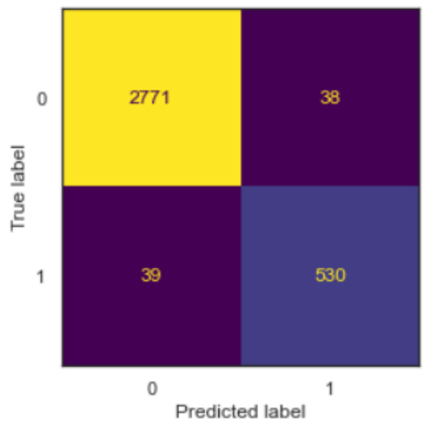
9.2.5. K-Nearest Neighbours (KNN)

KNN classifier works by looking at K-Nearest Neighbours to the given datapoint. It decides the target value based on its neighbours. KNN works on a principle assuming every data point falling near to each other is falling in the same class. It is also a black box model and lacks interpretability. Since it is non-parametric, it may be computationally expensive and require more memory to store training data. It also has a tendency to overfit. Although this model was tried on the given data and tuned extensively, due to the above said reasons, it has been decided not to select this as best model even if model performance is good. Hence, minimal details of this algorithm are given in this section.

9.2.5.1. KNN best model performance metrics

Hyper parameters used: KNeighborsClassifier(algorithm = 'auto', metric= 'minkowski', p= 1, weights= 'distance')

Train dataset Confusion Matrix	**Test dataset Confusion Matrix**

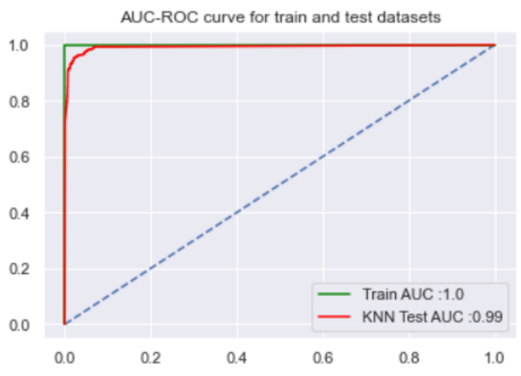
True label 0 1 6555 0 0 1327 0 1 Predicted label	True label 0 1 2771 38 39 530 0 1 Predicted label

Train dataset Classification report

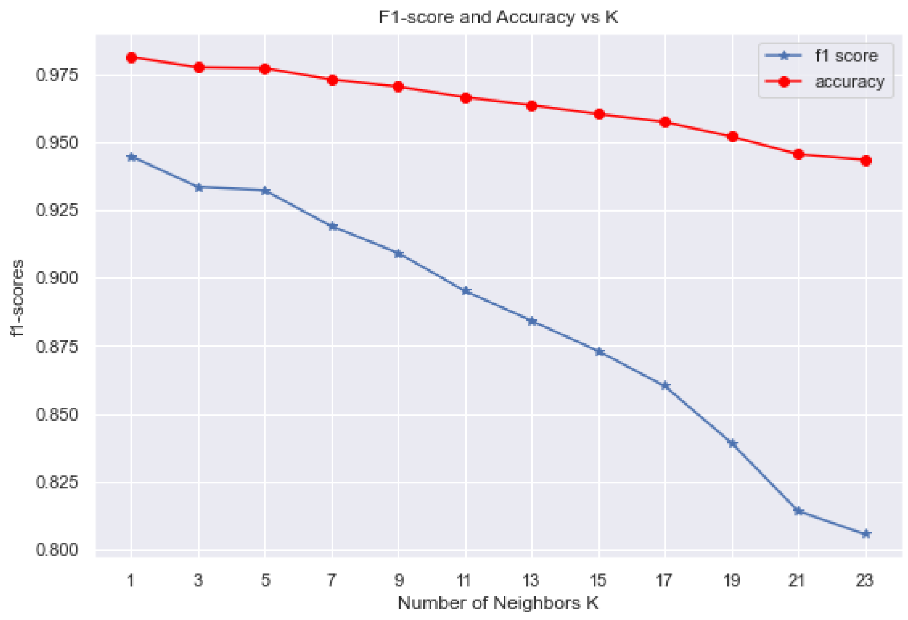
```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00      6555
           1       1.00      1.00      1.00      1327

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

Test dataset Classification report

```
              precision    recall  f1-score   support

           0       0.99      0.99      0.99      2809
           1       0.93      0.93      0.93       569

    accuracy                           0.98      3378
   macro avg       0.96      0.96      0.96      3378
weighted avg       0.98      0.98      0.98      3378
```



Observations: Train dataset has overfit because of a smaller number of neighbours selection. Let us try to observe the f1-score and accuracy for different values of K (neighbours) again using the above hyperparameters.



Observations: 5 seems to be the optimum value but for that, train dataset has fully grown.

KNN_model2 has the best grid with best neighbours. Even though the training data performance has fully grown, test data is not far behind and has a difference of 7%. Let's use cross validation on full data to see if the f1-score of 93% holds true. For 5-fold cross validation on full data, the following are the F1-scores

```
[0.90358127, 0.93530997, 0.93405114, 0.91005291, 0.84656085]
```

Observations: The cross-validation scores on train dataset and entire dataset for cv=5 are comparable to test dataset but within the folds, the differences are quite high and there are some inconsistencies within folds. Hence K-means may not always give predictable results as in testing. Increasing neighbours to 7 may reduce variance but the f1-score may drop considerably compared to other models.

9.2.6. Random Forest (RF)

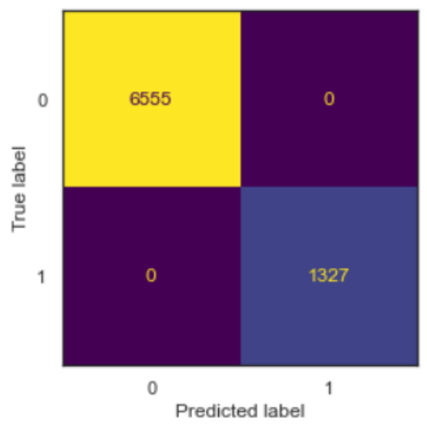
Random forest is an ensemble machine learning algorithm that uses bootstrapping to reduce variance in the underlying decision trees. It also selects only a subset of features for each node split decision. Since it is an ensemble of trees with varying features for each node, each tree is different from another. Random forest is resistant to outliers and does not require the data to be scaled.

In this modelling exercise, SKlearn's Random Forest classifier function was used for modelling:

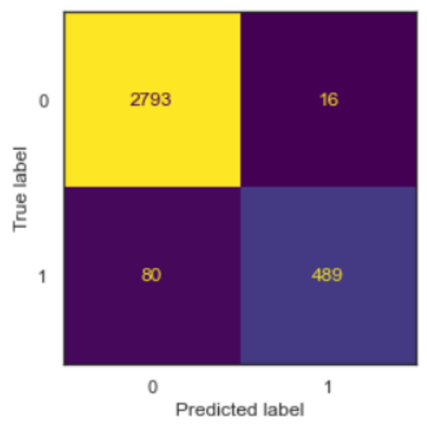
- The base model was run with default hyperparameters and the performance metrics noted. Tuning was later done using GridSearchCV.
- Model performance metrics for base model and best model have been provided in the below sections

9.2.6.1. Random Forest base model with default hyperparameters

Train dataset Confusion Matrix



Test dataset Confusion Matrix

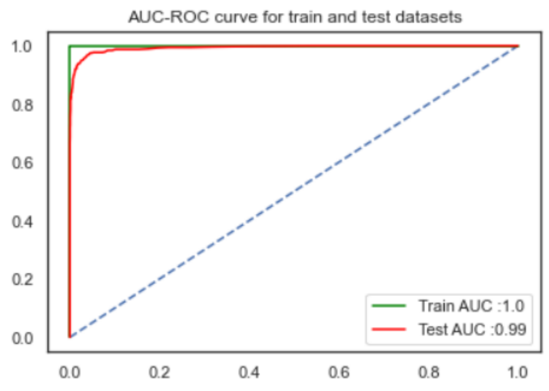


Train dataset Classification report

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00      6555
           1       1.00      1.00      1.00      1327

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

Test dataset Classification report

```
              precision    recall  f1-score   support

           0       0.97      0.99      0.98      2809
           1       0.97      0.86      0.91       569

    accuracy                           0.97      3378
   macro avg       0.97      0.93      0.95      3378
weighted avg       0.97      0.97      0.97      3378
```



Clearly, the model has overfit on the train dataset as the test dataset shows a recall of only 0.86. The model has to be tuned to reduce variance.

9.2.6.2. SKLearn Random Forest - model tuning

- The model's hyperparameters were tuned using GridSearchCV function and model constructed using the best parameters selected by GridSearchCV.

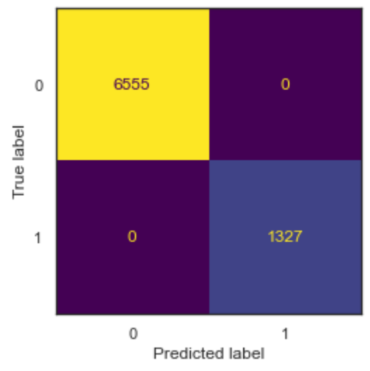
- As can be seen from the base model, the model had overfit on train dataset (all 1s). During tuning, the tree depth was contained, the minimum samples in each leaf increased to reduce overfit. The metrics for this model is given in section 7.6.3 (hyperparameter tuned model).
- One other model using data that was not outlier treated (i.e., outliers left as-is) was built using default hyper parameters. That model showed a 0.01 improvement over outlier treated base model for precision and f1-score, but recall was still an overfit.
    - The best model is highlighted in green in the table below

Model	Data Used			Hyper	Train data					Test data				
reference	Outliers	Smote	Scaling	parameters	Accuracy	Precision	Recall	F1	AUC	Accuracy	Precision	Recall	F1	AUC
RF_model1	Treated	No	No	Default base model	1	1	1	1	1	0.97	0.97	0.86	0.91	0.99
RF_model2	**No**	**No**	**No**	**Default base model with outliers**	**1**	**1**	**1**	**1**	**1**	**0.97**	**0.98**	**0.86**	**0.92**	**0.99**
RF_model3	Treated	No	No	GridSearchCV, best model for f1 score	0.98	0.98	0.91	0.94	1	0.95	0.9	0.78	0.84	0.98

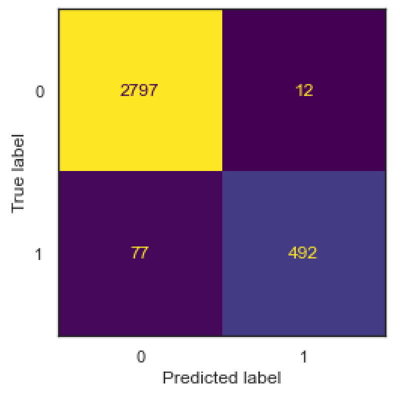
Table 9-9 Model tuning – Random Forest models performance

9.2.6.3. Random Forest best model – default hyperparameters with outliers

Train dataset Confusion Matrix



Test dataset Confusion Matrix

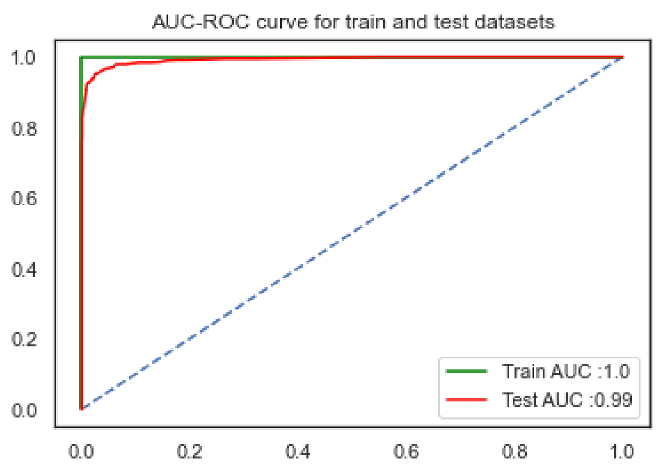


Train dataset Classification report

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00      6555
           1       1.00      1.00      1.00      1327

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

Test dataset Classification report

```
              precision    recall  f1-score   support

           0       0.97      1.00      0.98      2809
           1       0.98      0.86      0.92       569

    accuracy                           0.97      3378
   macro avg       0.97      0.93      0.95      3378
weighted avg       0.97      0.97      0.97      3378
```



9.2.6.4. Model interpretation

This model has overfitted both in the default base model as well as the hyper parameter tuned model. The tuned model in fact performed worser than the base model. Hence in the final comparison, this cannot get selected as best model.

SKlearn's permutation importance function was used to determine the features that are important to the model.

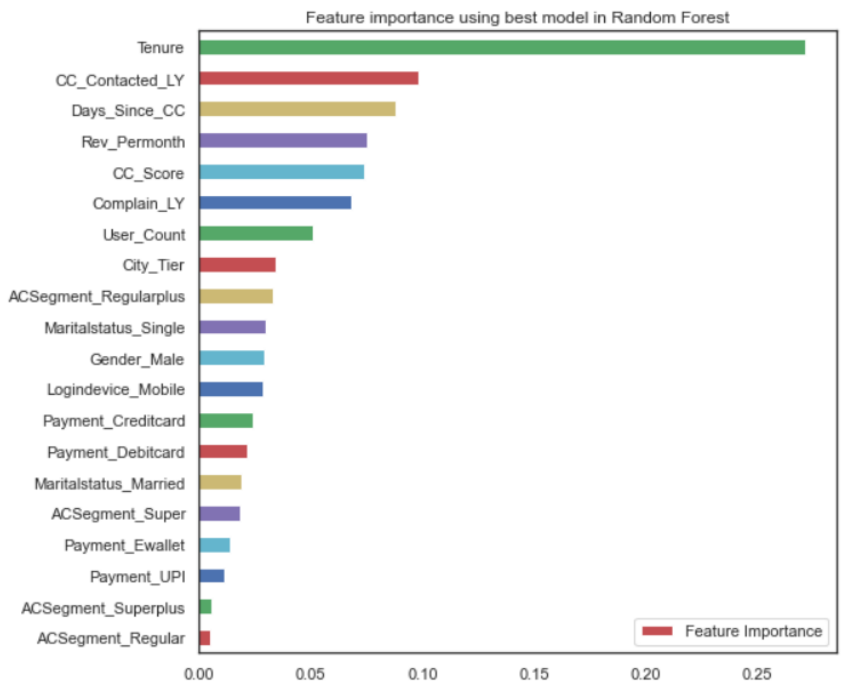


Figure 9-3 Feature importance from the best Random Forest model

Similar to SVM and ANN, this model has also picked Tenure to be the most important feature. However, the next two parameters in terms of importance are not the same amongst ANN, SVM and Random Forest. Since the performance for this model is not good the feature importance may not be reflective of actual data.

This model has picked Tenure to be the most important feature followed by Customer care contacted previous year and Days since customer care last contact.

- From EDA, we can observe that for lower tenures especially within the first year, the churn is higher. Hence, once a customer has been acquired, the first year is very important to keep the customer satisfied.
- The next important parameter to predict customer churn per this model is number of times customer care was contacted by the customer. Per EDA, the median and third quantile of number of times customer care was contacted previous year is higher for churned customers compared to active/current customers.
- Churned customers had contacted customer care more recently before churning than active customers. Per EDA, median days since last customer connect is higher for active customers. Churned customers had contacted customer care recently before churning.

9.2.7. Adaboost

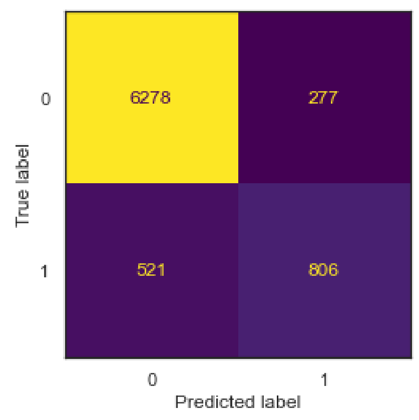
An AdaBoost classifier is a meta-estimator that begins by fitting a classifier on the original dataset and then fits additional copies of the classifier on the same dataset but where the weights of incorrectly classified instances are adjusted such that subsequent classifiers focus more on difficult cases.

In this modelling exercise, SKlearn's Adaboost classifier function was used for modelling:

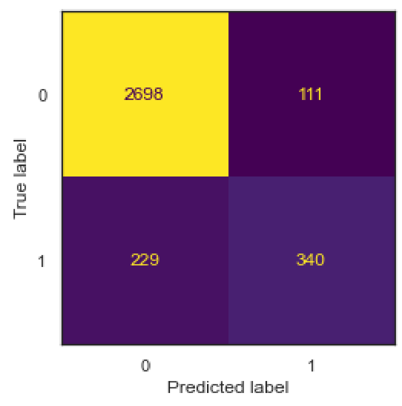
- The base model was run with default hyperparameters and the performance metrics noted. Tuning was later done using GridSearchCV.
- Model performance metrics for base model and tuned model have been provided in the below sections

9.2.7.1. Adaboost base model with default hyperparameters

Train dataset Confusion Matrix



Test dataset Confusion Matrix

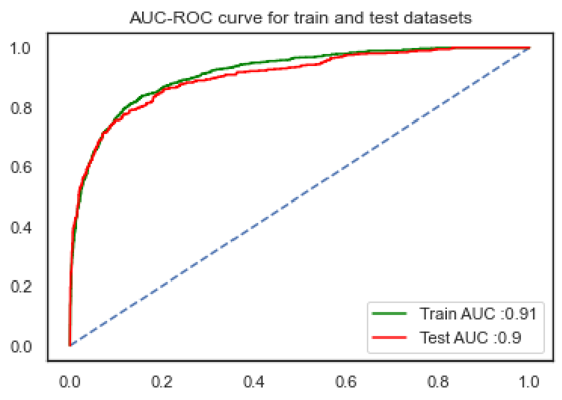


Train dataset Classification report

```
              precision    recall  f1-score   support

           0       0.92      0.96      0.94      6555
           1       0.74      0.61      0.67      1327

    accuracy                           0.90      7882
   macro avg       0.83      0.78      0.80      7882
weighted avg       0.89      0.90      0.89      7882
```

Test dataset Classification report

```
              precision    recall  f1-score   support

           0       0.92      0.96      0.94      2809
           1       0.75      0.60      0.67       569

    accuracy                           0.90      3378
   macro avg       0.84      0.78      0.80      3378
weighted avg       0.89      0.90      0.89      3378
```



There is no overfit or underfit issues with the model, but the model has not performed well on predicting minority class. The section below shows the results of tuning.

9.2.7.2. SKLearn Adaboost - model tuning

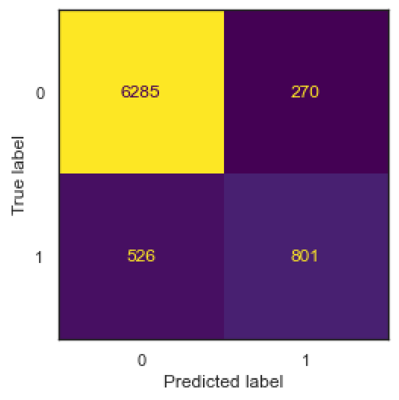
- The model's hyperparameters were tuned using GridSearchCV function and model constructed using the best parameters selected by GridSearchCV.
  - The best model is highlighted in green in the table below. Tuning has not resulted in noticeable improvement in the model. The details of the tuned model performance metrics are given in section 7.7.3

Model	Data Used			Hyper	Train data					Test data				
reference	Outliers	Smote	Scaling	parameters	Accuracy	Precision	Recall	F1	AUC	Accuracy	Precision	Recall	F1	AUC
ADA_model1	Treated	No	No	Default base model	0.9	0.74	0.61	0.67	0.91	0.9	0.75	0.6	0.67	0.9
ADA_model2	**Treated**	**No**	**No**	**GridSearchCV, best model for f1 score**	**0.9**	**0.75**	**0.6**	**0.67**	**0.92**	**0.9**	**0.76**	**0.6**	**0.67**	**0.91**

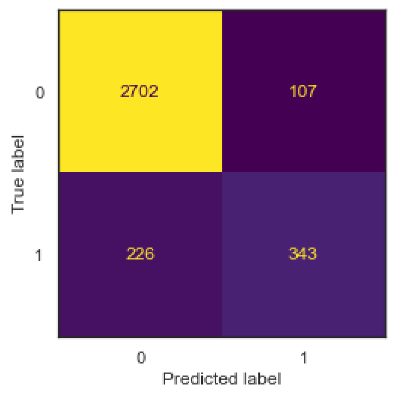
Table 9-10 Model tuning – Adaboost models performance

9.2.7.3. Adaboost best model

Train dataset Confusion Matrix



Test dataset Confusion Matrix

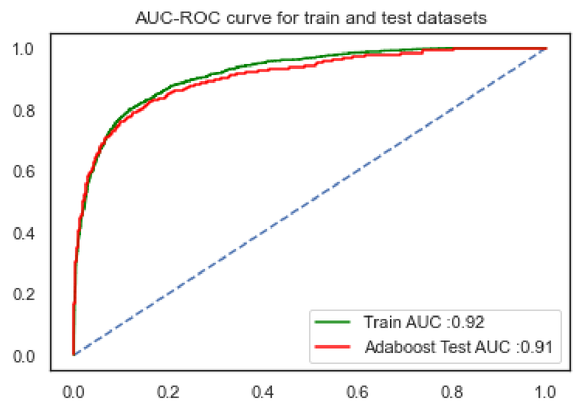


Train dataset Classification report

```
              precision    recall  f1-score   support

           0       0.92      0.96      0.94      6555
           1       0.75      0.60      0.67      1327

    accuracy                           0.90      7882
   macro avg       0.84      0.78      0.80      7882
weighted avg       0.89      0.90      0.89      7882
```

Test dataset Classification report

```
              precision    recall  f1-score   support

           0       0.92      0.96      0.94      2809
           1       0.76      0.60      0.67       569

    accuracy                           0.90      3378
   macro avg       0.84      0.78      0.81      3378
weighted avg       0.90      0.90      0.90      3378
```



9.2.7.4. Model interpretation

The performance of this model when compared with other models is low. But it has not shown any overfitting or underfitting.

Feature importance from Adaboost

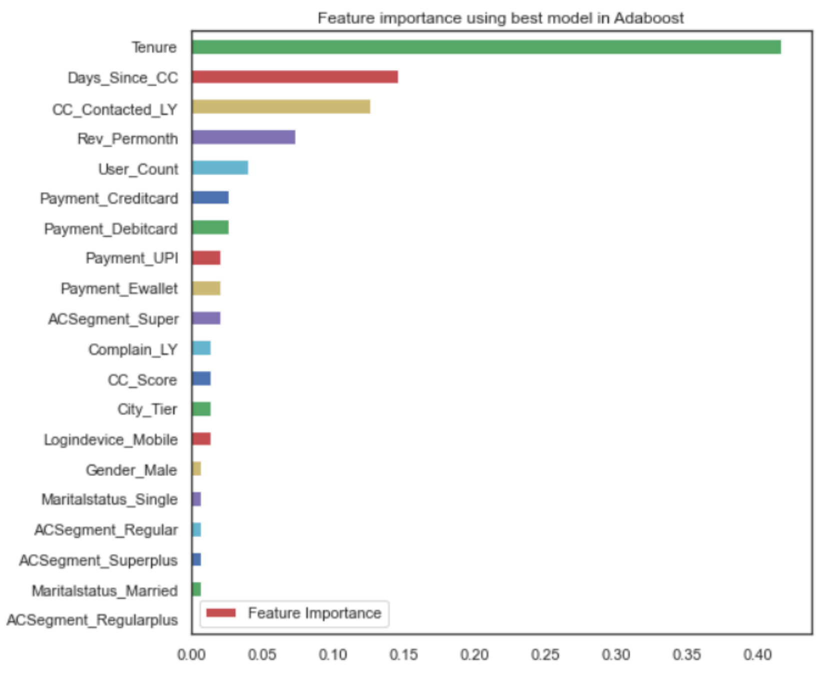


Figure 9-4 Feature importance from best Adaboost model

Similar to SVM and ANN, this model has also picked Tenure to be the most important feature. Since the performance for this model is not good the feature importance may not be reflective of actual data.

This model has picked Tenure to be the most important feature followed by Days since customer care last contact and Customer care contacted previous year.

- From EDA, we can observe that for lower tenures especially within the first year, the churn is higher. Hence, once a customer has been acquired, the first year is very important to keep the customer satisfied.
- Churned customers had contacted customer care more recently before churning than active customers. Per EDA, median days since last customer connect is higher for active customers. Churned customers had contacted customer care recently before churning.
- The next important parameter to predict customer churn per this model is number of times customer care was contacted by the customer. Per EDA, the median and third quantile of number of times customer care was contacted previous year is higher for churned customers compared to active/current customers.

9.2.8. Gradient Boost

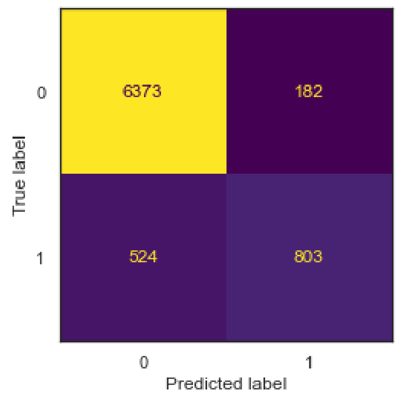
Gradient boost is an ensemble machine learning algorithm that trains underlying models in a gradual, additive and sequential manner.

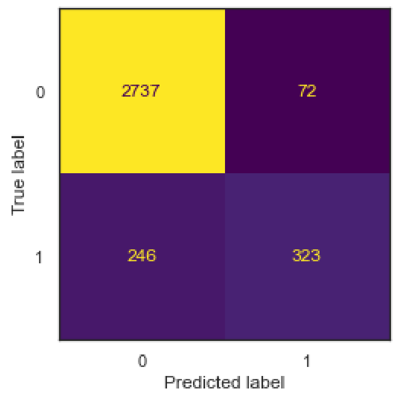
In this modelling exercise, SKlearn's Gradient Boost classifier function was used for modelling:

- The base model was run with default hyperparameters and the performance metrics noted. Tuning was later done using GridSearchCV.
- Model performance metrics for base model and best model have been provided in the below sections

9.2.8.1. Gradient boost base model with default hyperparameters

Train dataset Confusion Matrix	**Test dataset Confusion Matrix**



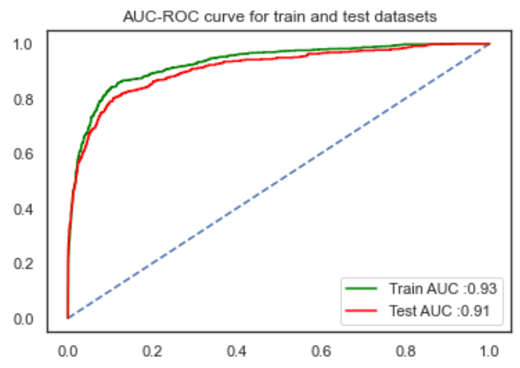


Train dataset Classification report

```
              precision    recall  f1-score   support

           0       0.92      0.97      0.95      6555
           1       0.82      0.61      0.69      1327

    accuracy                           0.91      7882
   macro avg       0.87      0.79      0.82      7882
weighted avg       0.91      0.91      0.90      7882
```

Test dataset Classification report

```
              precision    recall  f1-score   support

           0       0.92      0.97      0.95      2809
           1       0.82      0.57      0.67       569

    accuracy                           0.91      3378
   macro avg       0.87      0.77      0.81      3378
weighted avg       0.90      0.91      0.90      3378
```



The model is robust (no overfit or underfit) but the recall is quite poor in both train and test data. Tuning was done to see if performance can be improved on the model.

9.2.8.2. SKLearn GradientBoost - model tuning

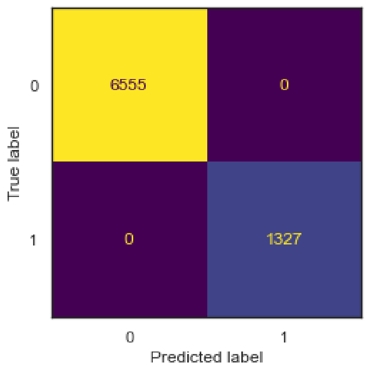
- The model's hyperparameters were tuned using GridSearchCV function and model constructed using the best parameters selected by GridSearchCV.
    - Tuning improved the model performance considerably

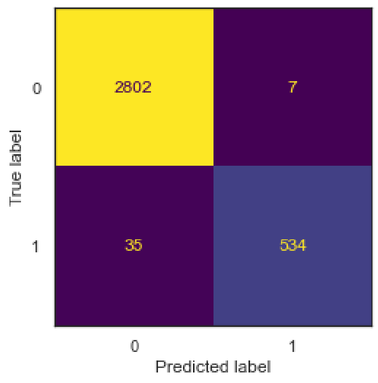
Model	Data Used			Hyper	Train data					Test data				
reference	Outliers	Smote	Scaling	parameters	Accuracy	Precision	Recall	F1	AUC	Accuracy	Precision	Recall	F1	AUC
GB_model1	Treated	No	No	Default base model	0.91	0.82	0.61	0.69	0.93	0.91	0.82	0.57	0.67	0.91
GB_model2	**Treated**	**No**	**No**	**GridSearchCV, best model for f1 score**	**1**	**1**	**1**	**1**	**1**	**0.99**	**0.99**	**0.94**	**0.96**	**1**

Table 9-11 Model tuning – Gradient Boost models performance

9.2.8.3. Gradient Boost best model with tuned hyperparameters

Train dataset Confusion Matrix	Test dataset Confusion Matrix



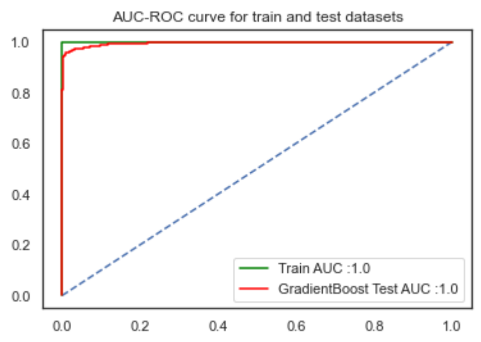


Train dataset Classification report

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00      6555
           1       1.00      1.00      1.00      1327

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

Test dataset Classification report

```
              precision    recall  f1-score   support

           0       0.99      1.00      0.99      2809
           1       0.99      0.94      0.96       569

    accuracy                           0.99      3378
   macro avg       0.99      0.97      0.98      3378
weighted avg       0.99      0.99      0.99      3378
```



9.3. Annexure C: References

1. DTH industry: A glimpse of profits at last! | Business Standard News (business-standard.com)
2. Forbes India - Direct To My Pocket: The DTH Tug Of War
3. Customer Retention Marketing vs. Customer Acquisition Marketing | OutboundEngine
4.The Precision-Recall Plot Is More Informative than the ROC Plot When Evaluating Binary Classifiers on Imbalanced Datasets (plos.org)