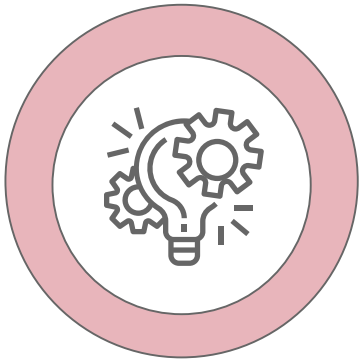
Telecom Churn Case Study

-

UPGRAD

BY: Shruti Koshti
Shreya Singh



Problem Statement

Business problem overview

In the telecom industry, customers are able to choose from multiple service providers and actively switch from one operator to another. In this highly competitive market, the telecommunications industry experiences an average of 15-25% annual churn rate. Given the fact that it costs 5-10 times more to acquire a new customer than to retain an existing one, **customer retention** has now become even more important than customer acquisition.

For many incumbent operators, retaining high profitable customers is the number one business goal.

To reduce customer churn, telecom companies need to **predict which customers are at high risk of churn.**

In this project, you will analyse customer-level data of a leading telecom firm, build predictive models to identify customers at high risk of churn and identify the main indicators of churn.

Problem Approach

1) Data Reading and Data Understanding
2) Data Cleaning & analysis
3) Data visualization using correlation
4) Outlier analysis
5) Building the model
6) Performing Logistic regression, Decision Tree and Random forest
7) Conclusion
8) Business Insights

Data Reading and Data Understanding

	mobile_number	circle_id	loc_og_t2o_mou	std_og_t2o_mou	loc_ic_t2o_mou	last_date_of_month_6	last_date_of_month_7	l
0	7000842753	109	0.0	0.0	0.0	6/30/2014	7/31/2014	
1	7001865778	109	0.0	0.0	0.0	6/30/2014	7/31/2014	
2	7001625959	109	0.0	0.0	0.0	6/30/2014	7/31/2014	
3	7001204172	109	0.0	0.0	0.0	6/30/2014	7/31/2014	
4	7000142493	109	0.0	0.0	0.0	6/30/2014	7/31/2014	

	mobile_number	circle_id	loc_og_t2o_mou	std_og_t2o_mou	loc_ic_t2o_mou	last_date_of_month_6	last_date_of_month_7	l
0	7000842753	109	0.0	0.0	0.0	6/30/2014	7/31/2014	
1	7001865778	109	0.0	0.0	0.0	6/30/2014	7/31/2014	
2	7001625959	109	0.0	0.0	0.0	6/30/2014	7/31/2014	
3	7001204172	109	0.0	0.0	0.0	6/30/2014	7/31/2014	
4	7000142493	109	0.0	0.0	0.0	6/30/2014	7/31/2014	
5	7000286308	109	0.0	0.0	0.0	6/30/2014	7/31/2014	
6	7001051193	109	0.0	0.0	0.0	6/30/2014	7/31/2014	
7	7000701601	109	0.0	0.0	0.0	6/30/2014	7/31/2014	
8	7001524846	109	0.0	0.0	0.0	6/30/2014	7/31/2014	
9	7001864400	109	0.0	0.0	0.0	6/30/2014	7/31/2014	

Statistical Analysis of the Data

	count	mean	std	min	25%	50%	75%	max
mobile_number	99999.0	7.001207e+09	695669.386290	7.000000e+09	7.000606e+09	7.001205e+09	7.001812e+09	7.002411e+09
circle_id	99999.0	1.090000e+02	0.000000	1.090000e+02	1.090000e+02	1.090000e+02	1.090000e+02	1.090000e+02
loc_og_t2o_mou	98981.0	0.000000e+00	0.000000	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00
std_og_t2o_mou	98981.0	0.000000e+00	0.000000	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00
loc_ic_t2o_mou	98981.0	0.000000e+00	0.000000	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00
arpu_6	99999.0	2.829874e+02	328.439770	-2.258709e+03	9.341150e+01	1.977040e+02	3.710600e+02	2.773109e+04
arpu_7	99999.0	2.785366e+02	338.156291	-2.014045e+03	8.698050e+01	1.916400e+02	3.653445e+02	3.514583e+04
arpu_8	99999.0	2.791547e+02	344.474791	-9.458080e+02	8.412600e+01	1.920800e+02	3.693705e+02	3.354362e+04
arpu_9	99999.0	2.616451e+02	341.998630	-1.899505e+03	6.268500e+01	1.768490e+02	3.534665e+02	3.880562e+04
onnet_mou_6	96062.0	1.323959e+02	297.207406	0.000000e+00	7.380000e+00	3.431000e+01	1.187400e+02	7.376710e+03
onnet_mou_7	96140.0	1.336708e+02	308.794148	0.000000e+00	6.660000e+00	3.233000e+01	1.155950e+02	8.157780e+03

Data Cleaning

- Below columns were dropped as the percentage of missing values were more than 40%. Hence dropped columns i.e. - 'total_data_rech_amt_9', 'arpu_3g_8', 'total_data_rech_amt_8', 'fb_user_8', 'max_rech_data_8', 'count_rech_2g_8', 'count_rech_3g_8', 'night_pck_user_8', 'date_of_last_rech_data_8', 'arpu_2g_8', 'arpu_3g_6', 'max_rech_data_6', 'total_data_rech_amt_6', 'night_pck_user_6', 'fb_user_6', 'count_rech_3g_6', 'date_of_last_rech_data_6', 'count_rech_2g_6', 'arpu_2g_6', 'date_of_last_rech_data_7', 'total_data_rech_amt_7', 'max_rech_data_7', 'arpu_3g_7', 'count_rech_2g_7', 'arpu_2g_7', 'count_rech_3g_7', 'night_pck_user_7', 'fb_user_7' .

- After removing above columns, rechecking the missing values in the balance columns (columns exhibit missing values less than 5%). Hence, we can eliminate the rows having the missing values to enhance the result of our analysis.

- Dropping the various Mobile number, dates columns those doesn't add any value to our analysis, except for marking the last day of the period.

- Column such as 'circle_id,' contain only a single value. Features of this nature indicate a lack of variance and are unlikely to contribute meaningfully to our target variable.

Assessing Correlation between features

[24]:
```
corr = Churn_Data.corr()
corr.loc[:, :] = np.tril(corr, -1)
corr = corr.stack()
high_corr_value = corr[(corr > 0.60) | (corr < -0.60)]
high_corr_value
```

[24]:

arpu_7	arpu_6	0.728704
arpu_8	arpu_6	0.671437
	arpu_7	0.778413
onnet_mou_7	onnet_mou_6	0.770224
onnet_mou_8	onnet_mou_6	0.646114
	onnet_mou_7	0.811314
offnet_mou_7	offnet_mou_6	0.755880
offnet_mou_8	offnet_mou_6	0.605742
	offnet_mou_7	0.772001
roam_ic_mou_8	roam_ic_mou_7	0.618233
roam_og_mou_6	roam_ic_mou_6	0.647696
roam_og_mou_8	roam_og_mou_7	0.605246
loc_og_t2t_mou_7	loc_og_t2t_mou_6	0.801091
loc_og_t2t_mou_8	loc_og_t2t_mou_6	0.708473
	loc_og_t2t_mou_7	0.836495
loc_og_t2m_mou_7	loc_og_t2m_mou_6	0.790918
loc_og_t2m_mou_8	loc_og_t2m_mou_6	0.698190
	loc_og_t2m_mou_7	0.826281
loc_og_t2f_mou_7	loc_og_t2f_mou_6	0.812776
loc_og_t2f_mou_8	loc_og_t2f_mou_6	0.674290
	loc_og_t2f_mou_7	0.779074
loc_og_mou_6	loc_og_t2t_mou_6	0.756545
	loc_og_t2t_mou_7	0.613042
	loc_og_t2m_mou_6	0.799117
	loc_og_t2m_mou_7	0.633168
loc_og_mou_7	loc_og_t2t_mou_6	0.630307
	loc_og_t2t_mou_7	0.765275
	loc_og_t2t_mou_8	0.656782
	loc_og_t2m_mou_6	0.634951
	loc_og_t2m_mou_7	0.767839

Assessing Correlation between features

[25]:
```
# List of columns that are explained well by other columns
drop_column_corr = ['loc_og_t2m_mou_6', 'std_og_t2t_mou_6', 'std_og_t2t_mou_7', 'std_og_t2t_mou_8', 'std_og_t2m_mou_6', 'std_og_t2m_mou_7',
                    'std_og_t2m_mou_8', 'total_og_mou_6', 'total_og_mou_7', 'total_og_mou_8', 'loc_ic_t2t_mou_6', 'loc_ic_t2t_mou_7',
                    'loc_ic_t2t_mou_8', 'loc_ic_t2m_mou_6', 'loc_ic_t2m_mou_7', 'loc_ic_t2m_mou_8', 'std_ic_t2m_mou_6', 'std_ic_t2m_mou_7',
                    'std_ic_t2m_mou_8', 'total_ic_mou_6', 'total_ic_mou_7', 'total_ic_mou_8', 'total_rech_amt_6', 'total_rech_amt_7',
                    'total_rech_amt_8', 'vol_3g_mb_6', 'vol_3g_mb_7', 'vol_3g_mb_8', 'loc_og_t2t_mou_6', 'loc_og_t2t_mou_7', 'loc_og_t2t_mou_8',
                    'loc_og_t2f_mou_6', 'loc_og_t2f_mou_7', 'loc_og_t2f_mou_8', 'loc_og_t2m_mou_6', 'loc_og_t2m_mou_7', 'loc_og_t2m_mou_8',
                    'loc_ic_t2f_mou_6', 'loc_ic_t2f_mou_7', 'loc_ic_t2f_mou_8']

# Drop the high corr columns
Churn_Data.drop(drop_column_corr, axis=1, inplace=True)
Churn_Data.shape
```

[25]: (28163, 87)

[26]:
```
Churn_Data.head()
```

[26]:

	arpu_6	arpu_7	arpu_8	onnet_mou_6	onnet_mou_7	onnet_mou_8	offnet_mou_6	offnet_mou_7	offnet_mou_8	roam_ic_mou_6	roam_ic_mou_7	roam_ic_mou_
7	1069.180	1349.850	3171.480	57.84	54.68	52.29	453.43	567.16	325.91	16.23	33.49	31.6
8	378.721	492.223	137.362	413.69	351.03	35.08	94.66	80.63	136.48	0.00	0.00	0.0
21	514.453	597.753	637.760	102.41	132.11	85.14	757.93	896.68	983.39	0.00	0.00	0.0
23	74.350	193.897	366.966	48.96	50.66	33.58	85.41	89.36	205.89	0.00	0.00	0.0
33	977.020	2362.833	409.230	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.0

Driving New Features

[27]:
```
# Create a total mou instead of offnet and onnet

Churn_Data['total_mou_6'] = Churn_Data['onnet_mou_6'] + Churn_Data['offnet_mou_6']
Churn_Data['total_mou_7'] = Churn_Data['onnet_mou_7'] + Churn_Data['offnet_mou_7']
Churn_Data['total_mou_8'] = Churn_Data['onnet_mou_8'] + Churn_Data['offnet_mou_8']

# Drop the redundant columns
Churn_Data.drop(['onnet_mou_6', 'onnet_mou_7', 'onnet_mou_8', 'offnet_mou_6', 'offnet_mou_7', 'offnet_mou_8'], axis=1, inplace=True)

Churn_Data.head()
```

[27]:

	arpu_6	arpu_7	arpu_8	roam_ic_mou_6	roam_ic_mou_7	roam_ic_mou_8	roam_og_mou_6	roam_og_mou_7	roam_og_mou_8	loc_og_t2c_mou_6	loc_og_t2c_mou
7	1069.180	1349.850	3171.480	16.23	33.49	31.64	23.74	12.59	38.06	0.0	
8	378.721	492.223	137.362	0.00	0.00	0.00	0.00	0.00	0.00	0.0	
21	514.453	597.753	637.760	0.00	0.00	0.00	0.00	0.00	0.00	0.0	
23	74.350	193.897	366.966	0.00	0.00	0.00	0.00	0.00	0.00	0.0	
33	977.020	2362.833	409.230	0.00	0.00	0.00	0.00	0.00	0.00	0.0	

- As the remaining columns provide a breakdown of totals into more detailed information, we won't consolidate them into one. Instead, we plan to generate new features by averaging the values for the 6th and 7th months to capture the details of the 'good' phase.

[28]:
```
# Seperate columns for 6th and 7th month
column_for_6_7 = [col[:-2] for col in Churn_Data.columns if '6' in col or '7' in col]

# Create new feature and drop the redundant columns
for col in set(column_for_6_7):
    Churn_Data[f'gd_ph_{col}'] = ( Churn_Data[f'{col}_6'] + Churn_Data[f'{col}_7'] ) / 2
    Churn_Data.drop([f'{col}_6', f'{col}_7'], axis=1, inplace=True)

Churn_Data.head()
```

[28]:

	arpu_8	roam_ic_mou_8	roam_og_mou_8	loc_og_t2c_mou_8	loc_og_mou_8	std_og_t2f_mou_8	std_og_mou_8	isd_og_mou_8	spl_og_mou_8	og_others_8	loc_ic_mo
7	3171.480	31.64	38.06	0.00	255.79	16.68	77.84	10.01	6.50	0.0	188
8	137.362	0.00	0.00	7.15	63.04	0.00	98.28	0.00	10.23	0.0	15
21	637.760	0.00	0.00	0.00	129.74	0.00	938.79	0.00	0.00	0.0	159
23	366.966	0.00	0.00	17.71	182.14	0.00	39.61	0.00	17.71	0.0	224
33	409.230	0.00	0.00	0.56	0.00	0.00	0.00	0.00	0.56	0.0	4(

The 'vbc' columns lack a numeric month suffix, leading to oversight. We will address this by averaging out the columns for this feature as well.

Checking the Outliers :-

[32]:
```
Churn_Data.describe()
```

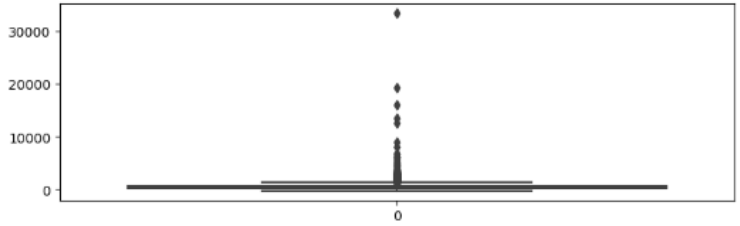
[32]:

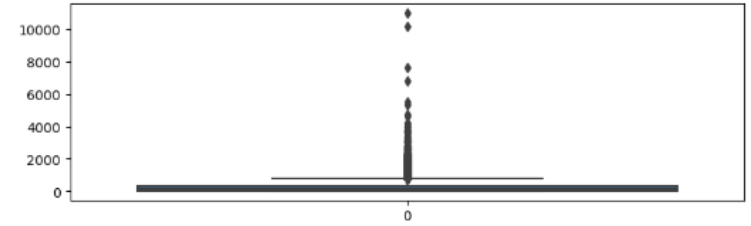
	arpu_8	roam_ic_mou_8	roam_og_mou_8	loc_og_t2c_mou_8	loc_og_mou_8	std_og_t2f_mou_8	std_og_mou_8	isd_og_mou_8	spl_og_mou_8	og_others_8
count	28163.000000	28163.000000	28163.000000	28163.000000	28163.000000	28163.000000	28163.000000	28163.000000	28163.000000	28163.000000
mean	528.992500	13.145865	20.926258	1.789363	247.845569	1.646062	322.121339	1.989213	6.889342	0.061151
std	500.479643	76.125433	107.722393	7.390599	367.353912	11.126142	615.275135	45.888765	20.813297	3.421503
min	-345.129000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	258.075500	0.000000	0.000000	0.000000	31.410000	0.000000	0.000000	0.000000	0.000000	0.000000
50%	445.338000	0.000000	0.000000	0.000000	124.060000	0.000000	34.990000	0.000000	0.700000	0.000000
75%	675.208500	0.000000	0.000000	0.130000	325.160000	0.000000	392.840000	0.000000	6.640000	0.000000
max	33543.624000	4169.810000	5337.040000	351.830000	11039.910000	516.910000	13980.060000	5681.540000	954.510000	394.930000

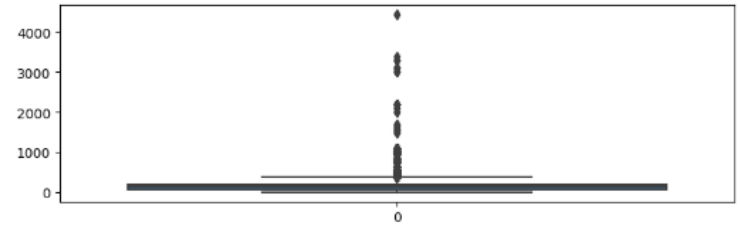
Observation

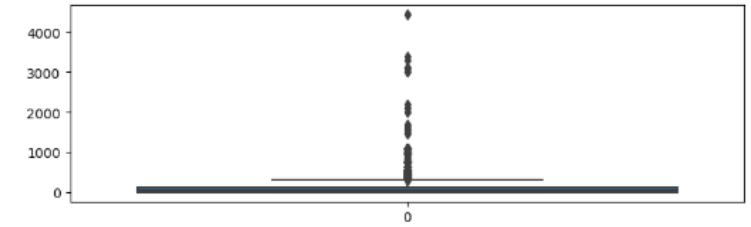
- Nearly every column exhibits outliers, primarily stemming from instances where the service was not utilized (indicated by 0.0). However, some outliers appear to be genuine.
- Given the absence of actual business personnel to verify the accuracy of the data, we intend to cap these features.

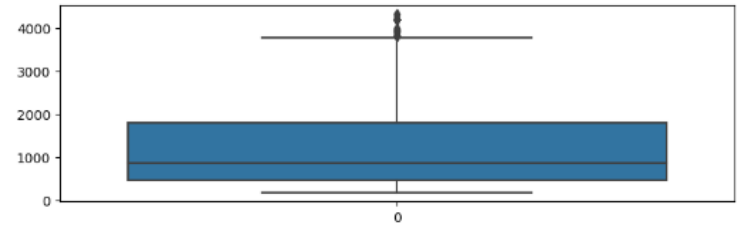
List of features to be analyzed for outlier :-

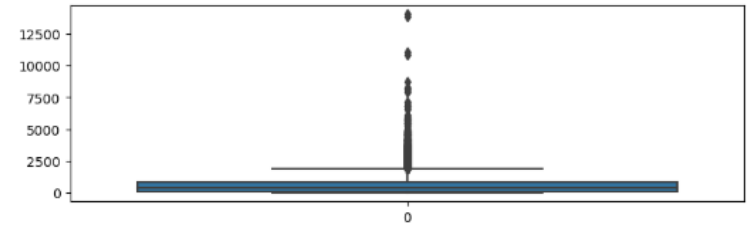


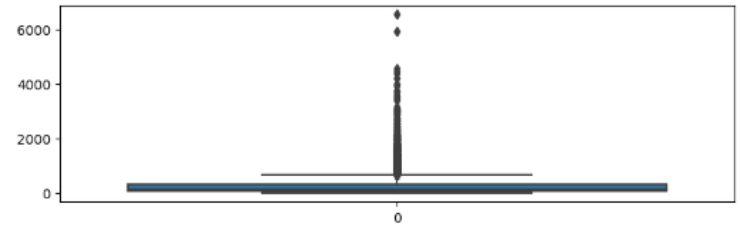


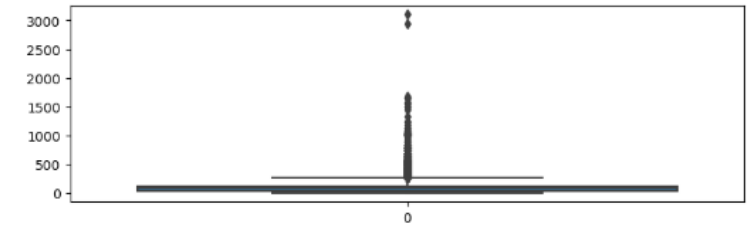










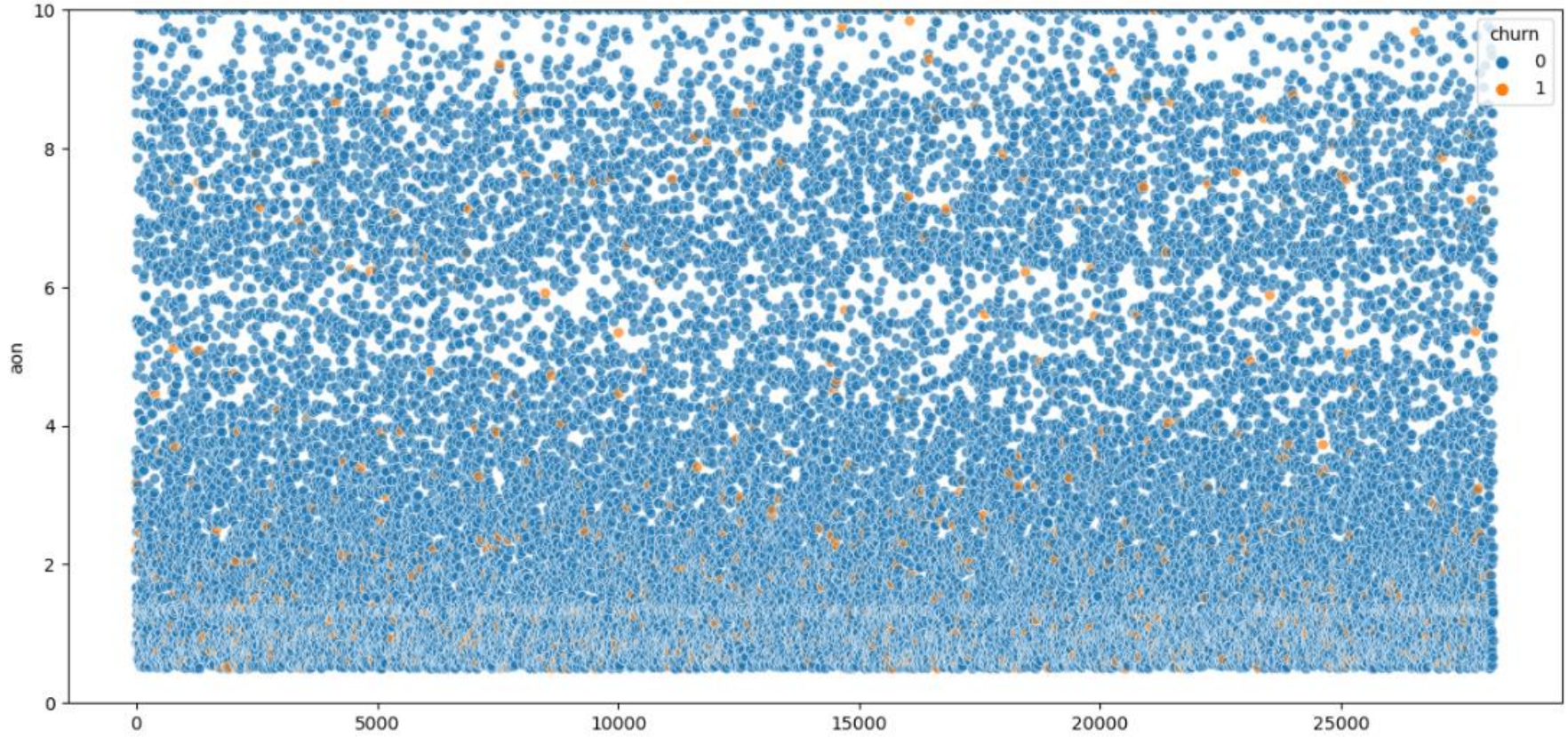


Observations for outlier :-

- From the above plots we can define following upper limits to the sepected variables

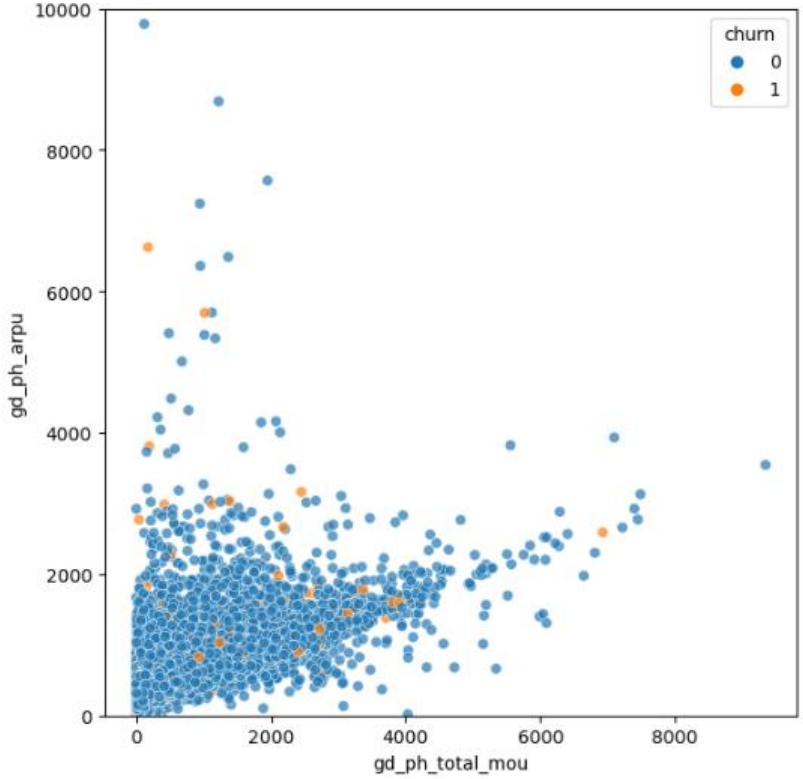
Feature	Value
arpu_8	7000
loc_og_mou_8	4000
max_rech_amt_8	1000
last_day_rch_amt_8	1000
aon	3000
total_mou_8	4000
gd_ph_loc_ic_mou	3000
gd_ph_last_day_rch_amt	1000
gd_ph_std_og_mou	4000
gd_ph_max_rech_amt	1500
gd_ph_loc_og_mou	3000
gd_ph_arpu	7000

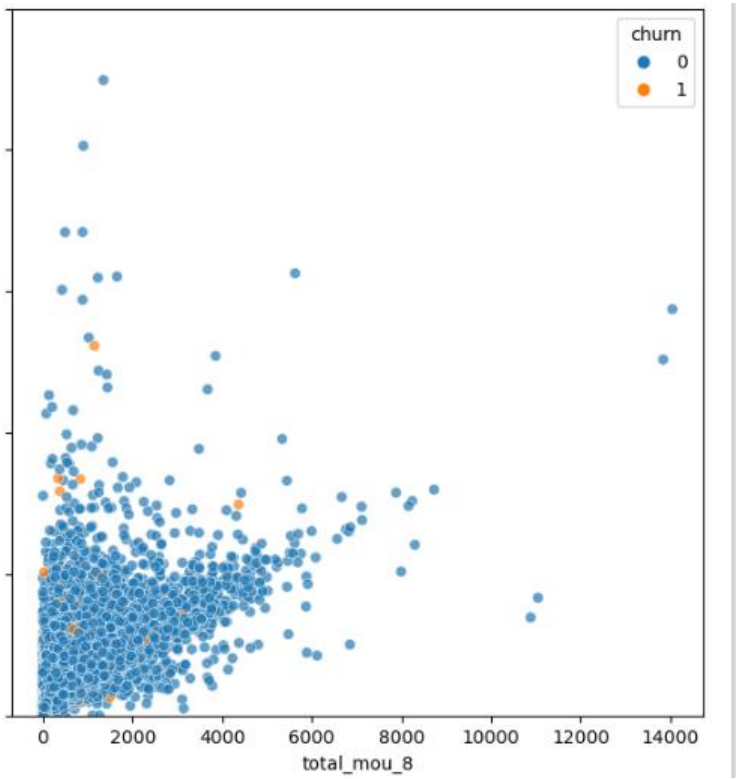
Observations regarding churn (based on tenure):-



Majority of churners had a tenure of less than 4 years

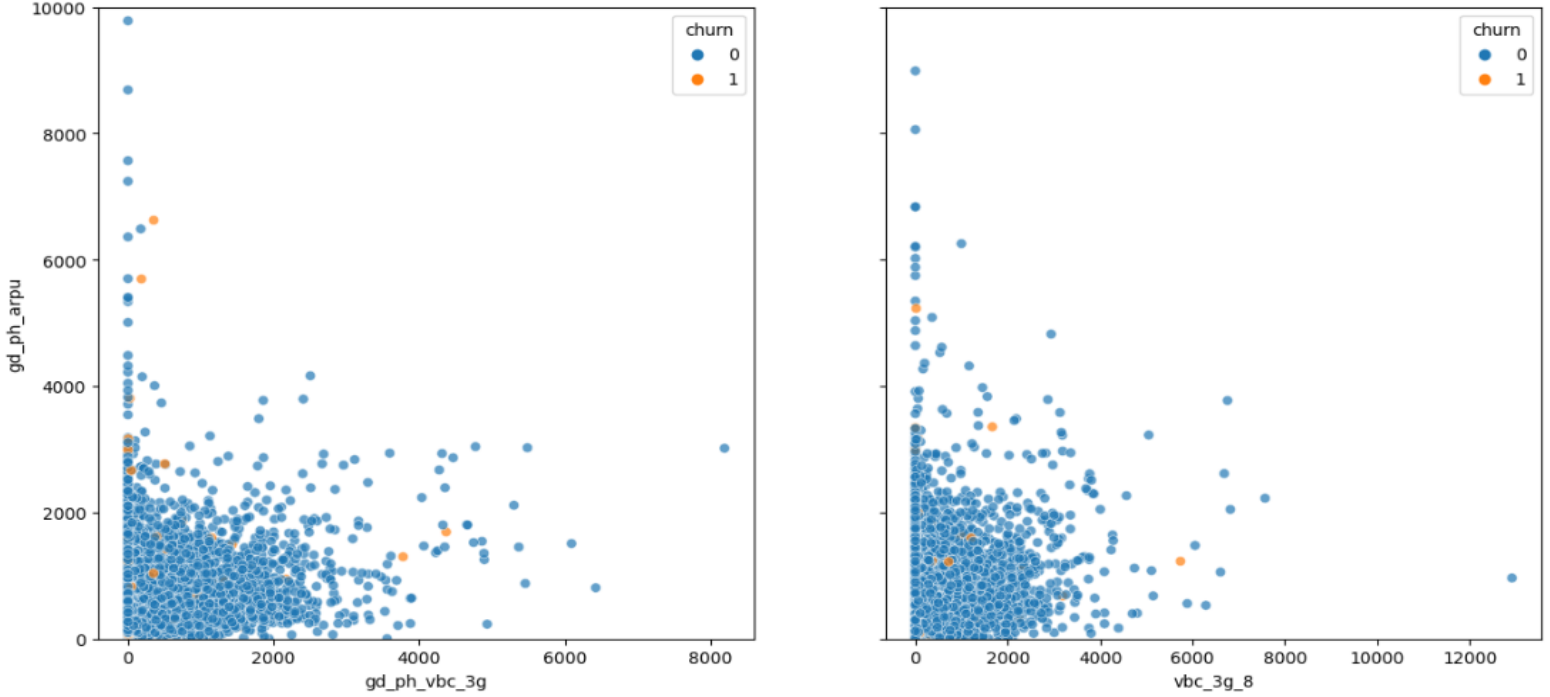
Effect on Revenue (based on churn):-





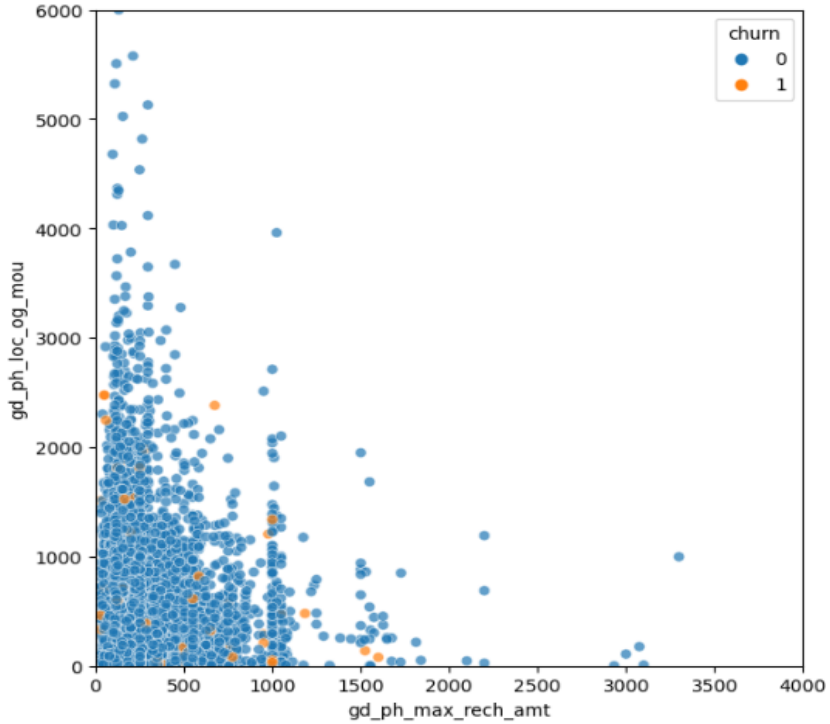
- The significant drop in MOU (Minutes of Usage) for churners during the action phase (8th month), consequently impacting the revenue generated from them. Additionally, it is intriguing that within the MOU range of 0-2000, the revenue is notably high, indicating that these users likely had other services contributing to the overall revenue.

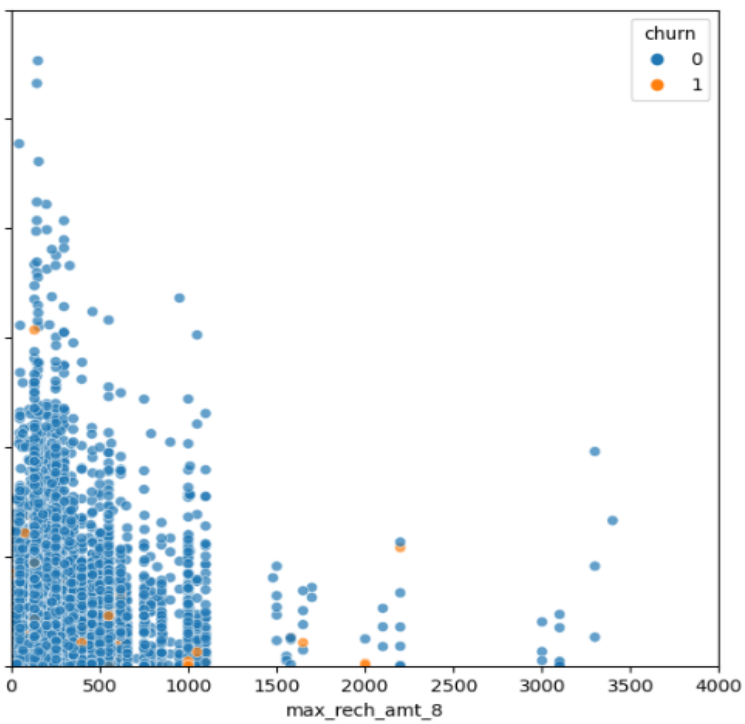
[36]:
```
# Lets check how the total_mou effects the revenue
fig, axes = plt.subplots(1, 2, sharey=True, figsize=(15, 7))
sns.scatterplot(y='gd_ph_arpu', x='gd_ph_vbc_3g', data=Churn_Data, ax=axes[0], hue='churn', alpha=0.7)
sns.scatterplot(y='arpu_8', x='vbc_3g_8', data=Churn_Data, ax=axes[1], hue='churn', alpha=0.7)


# Limiting the graph to more general upper bound
plt.ylim(0,10000)
plt.show()
```



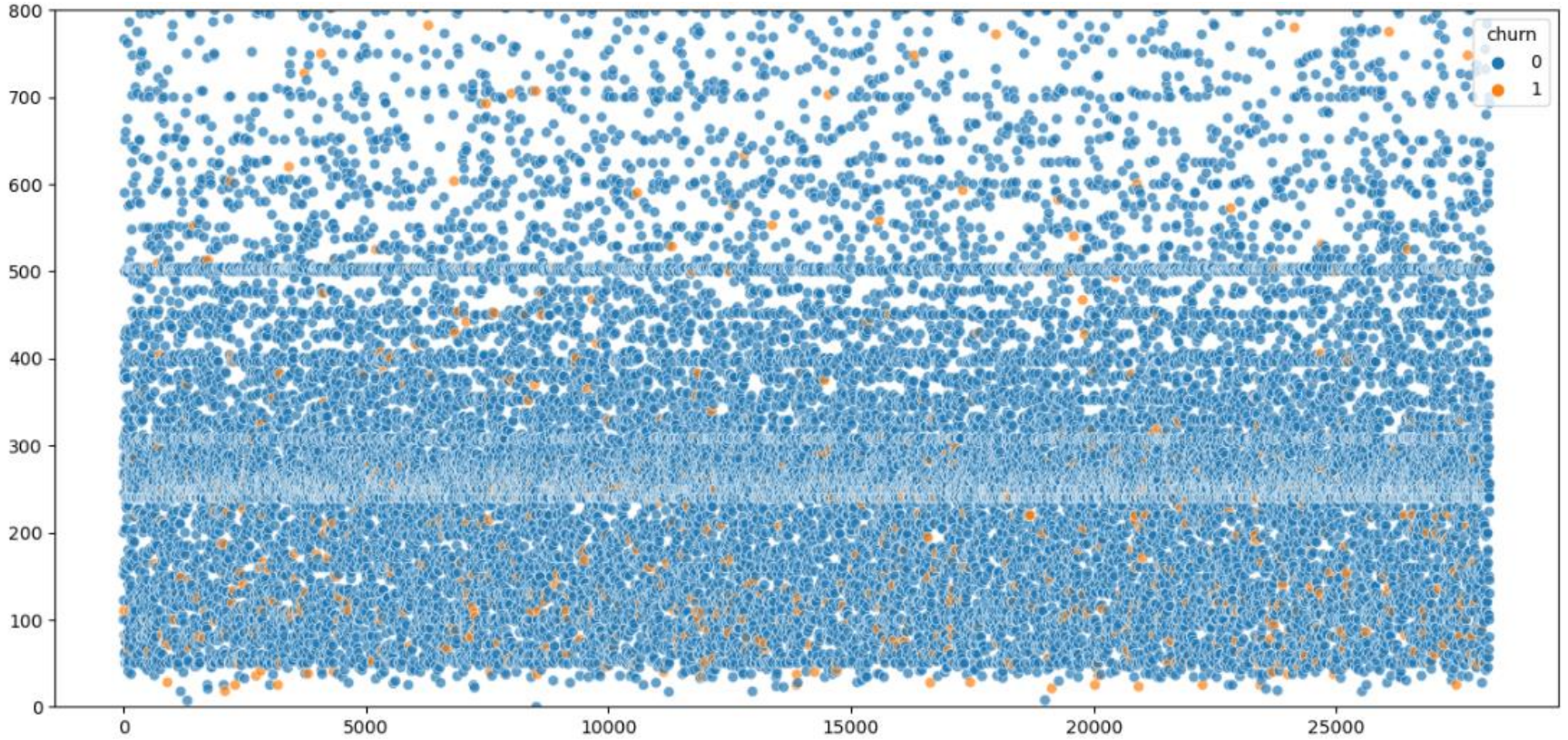
- We can see that the users who were using very less amount of VBC data and yet were generating high revenue churned
- Yet again we see that the revenue is higher towards the lesser consumption side

Observation based on Recharge amount :-





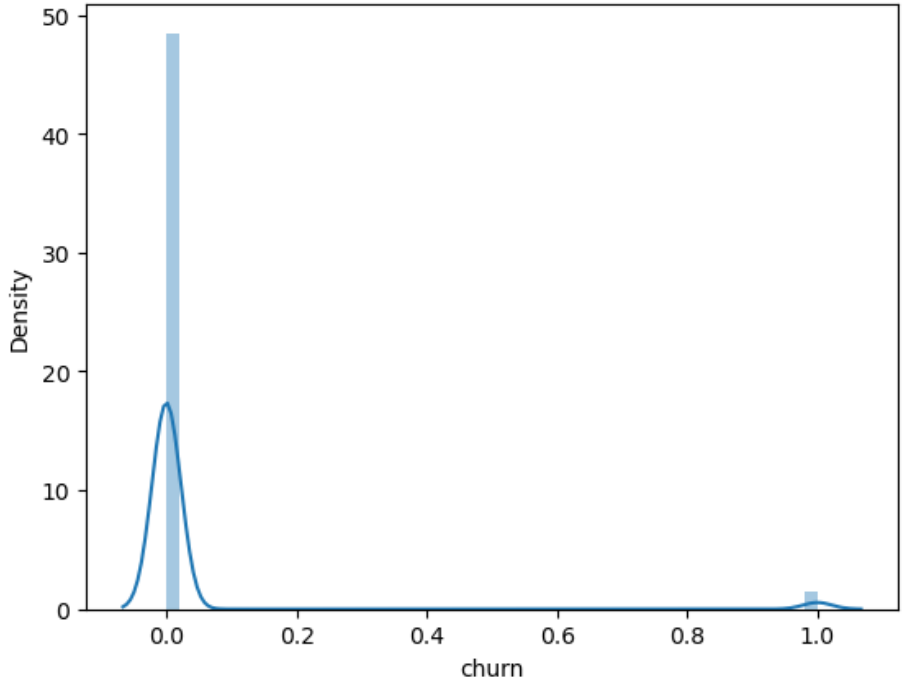
1. Users who recharged with higher amounts tended to use the service for local purposes less frequently compared to those who did lesser recharge amounts.
2. Intuitively, individuals whose maximum recharge amount and local outgoing usage were very low even during the good phase exhibited a higher churn rate.



3. Users with a maximum recharge amount less than 200 displayed a higher churn rate

Observation:-

```
[40]: # Distribution of target variable

sns.distplot(Churn_Data['churn'])
plt.show()
```



- Despite the variable not being skewed, there is a significant imbalance, with approximately 94% of the dataset representing non-churners. To address this imbalance, we will employ the SMOTE algorithm.

Model Building – 1) Logistic Regression

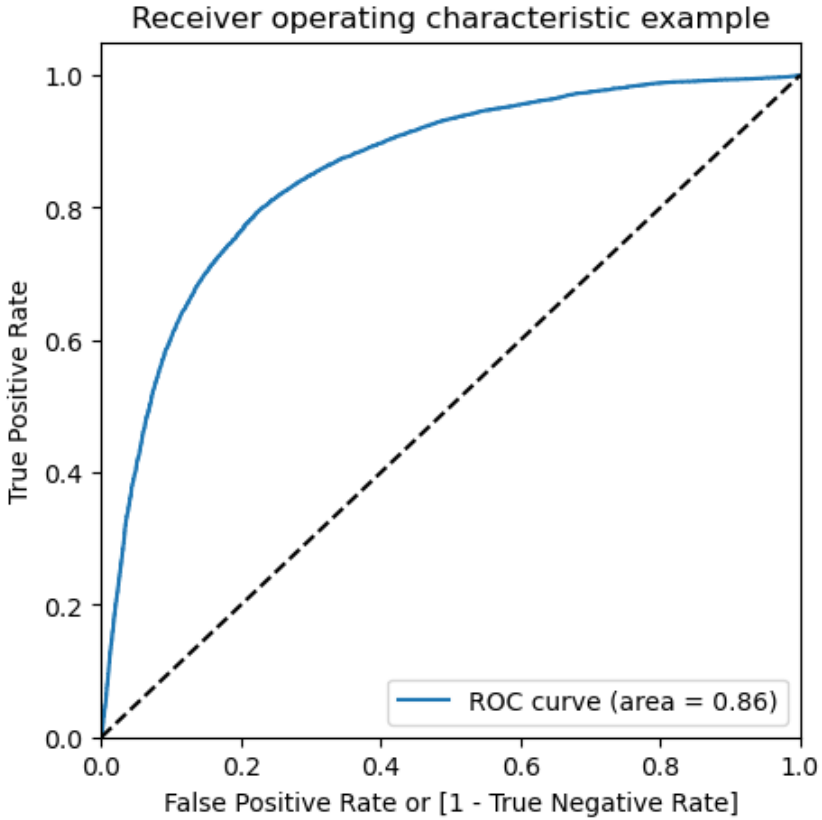
Following steps were performed to arrive at the best fit model –

a) Initial regression model analysis (build model)
b) RFE for feature selection
c) Build model with RFE selected features
d) Predict on Train set
e) Checking Confusion matrix & Accuracy
f) Checking VIF values of the feature variables.

Final Model after multiple iterations

```
[80]: # Let's see the sensitivity of our logistic regression model
      TP / float(TP+FN)

[80]: 0.8142476349762191

[81]: # Let us calculate specificity
      TN / float(TN+FP)

[81]: 0.7508385744234801

[82]: # Calculate false postive rate - predicting churn when customer does not have churned
      print(FP/ float(TN+FP))

      0.2491614255765199

[83]: # positive predictive value
      print (TP / float(TP+FP))

      0.7661928884080067

[84]: # Negative predictive value
      print (TN / float(TN+ FN))

      0.8012304250559285
```

Plotting ROC :-



- Logistic regression yields an accuracy of 78.5% on the training data and 78.8% on the testing data.
- Notably, critical features predominantly emerge from the action phase, aligning with the business understanding that the action phase requires heightened attention.

2) Decision Tree:-

[92]:
```
Grid_search.best_score_
```

[92]:
```
0.8879178946161582
```

[93]:
```
# Best estimator
dt_best = Grid_search.best_estimator_
dt_best
```

[93]:
```
DecisionTreeClassifier(max_depth=40, random_state=42)
```
In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

[94]:
```
y_train_pred = dt_best.predict(X_train)
y_test_pred = dt_best.predict(X_test)

# Print the report
print(metrics.classification_report(y_test, y_test_pred))
```

```
              precision    recall  f1-score   support

           0       0.92      0.87      0.90      8215
           1       0.88      0.92      0.90      8162

    accuracy                           0.90     16377
   macro avg       0.90      0.90      0.90     16377
weighted avg       0.90      0.90      0.90     16377
```

[95]:
```
# ROC
plot_roc_curve=(dt_best, X_train, y_train)
plt.show()
```

> An accuracy of 90% is achieved on the test data using the decision tree model.

3) Random Forest:-

[96]:
```
from sklearn.ensemble import RandomForestClassifier

rf = RandomForestClassifier(n_estimators=15, max_depth=10, max_features=5, random_state=25, oob_score=True)
rf.fit(X_train, y_train)

y_train_pred = rf.predict(X_train)

# Train Accuracy
y_train_pred = intial_dt.predict(X_train)
print(f'Train accuracy : {metrics.accuracy_score(y_train, y_train_pred)}')

y_test_pred = rf.predict(X_test)

# Print the report
print(metrics.classification_report(y_test, y_test_pred))

# Plotting ROC
plot_roc_curve=(rf, X_train, y_train)
plt.show()
```

```
Train accuracy : 0.8775285897469448
              precision    recall  f1-score   support

           0       0.92      0.86      0.89      8215
           1       0.87      0.92      0.90      8162

    accuracy                           0.89     16377
   macro avg       0.89      0.89      0.89     16377
weighted avg       0.89      0.89      0.89     16377
```

Conclusion :

1. In our business problem of customer retention, prioritizing higher recall is crucial. Identifying true positives with precision is more cost-effective than losing a customer or acquiring new ones.
2. Upon comparing the trained models, the tuned Random Forest stand out as the top performer. Exhibiting the highest accuracy, along with impressive recall rates of 95%.
3. Considering the balance between performance and model simplicity, we opt for Random Forest for its comparative simplicity while maintaining strong predictive capabilities.

Insights and strategies derived from the analysis:

1.Action Phase Significance:

1. The majority of top predictors are from the action phase, indicating that the observed drop in user engagement during this phase is crucial for predicting churn.

2.Additional Insights from EDA:

1. Users with a maximum recharge amount less than 200 even in the good phase should be tagged and regularly reassessed, as they exhibit a higher likelihood of churning.
2. Monitoring users who have been associated with the network for less than 4 years is essential, as this group tends to show a higher churn rate.
3. While MOU is a significant factor, data usage, especially VBC (Volume Based Charging), becomes relevant, especially if the user is not using a data pack.

Business Insights :-

1. The telecom company should prioritize addressing roaming rates & consider offering attracting packages to customers who frequently use services in roaming zone. This approach can contribute to customer satisfaction and retention.
2. Special attention should be given to users who receive incoming calls from fixed lines significantly below the average by 1.27 standard deviations. These users are identified as having a higher likelihood of churning, and targeted efforts to retain them, such as personalized offers or incentives, may be effective in reducing churn.
3.There is a need for the company to evaluate and potentially revise STD and ISD rates. If the current rates are perceived as too high, introducing competitive packages in these categories could enhance customer loyalty.