

Problem statement:-

To reduce customer churn, telecom companies need to predict which customers are at high risk of churn. In this project, we 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.

Retaining high profitable customers is the main business goal here.

Steps:-

1. Reading, understanding and visualising the data
2. Preparing the data for modelling
3. Building the model
4. Evaluate the model

```
# Importing the libraries
import pandas as pd
import numpy as np

import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns

import warnings
warnings.filterwarnings('ignore')

pd.set_option('display.max_columns', 500)
```

Reading and understanding the data

```
# Reading the dataset
df = pd.read_csv('telecom_churn_data.csv')
df.head()
```

	mobile_number	circle_id	loc_og_t2o_mou	std_og_t2o_mou
0	7000842753	109	0.0	0.0
1	7001865778	109	0.0	0.0
2	7001625959	109	0.0	0.0
3	7001204172	109	0.0	0.0
4	7000142493	109	0.0	0.0

	last_date_of_month_6	last_date_of_month_7	last_date_of_month_8	\	
0	6/30/2014	7/31/2014	8/31/2014		
1	6/30/2014	7/31/2014	8/31/2014		
2	6/30/2014	7/31/2014	8/31/2014		
3	6/30/2014	7/31/2014	8/31/2014		
4	6/30/2014	7/31/2014	8/31/2014		
	last_date_of_month_9	arpu_6	arpu_7	arpu_8	arpu_9
onnet_mou_6	\				
0	9/30/2014	197.385	214.816	213.803	21.100
NaN					
1	9/30/2014	34.047	355.074	268.321	86.285
24.11					
2	9/30/2014	167.690	189.058	210.226	290.714
11.54					
3	9/30/2014	221.338	251.102	508.054	389.500
99.91					
4	9/30/2014	261.636	309.876	238.174	163.426
50.31					
	onnet_mou_7	onnet_mou_8	onnet_mou_9	offnet_mou_6	
offnet_mou_7	\				
0	NaN	0.00	NaN	NaN	NaN
1	78.68	7.68	18.34	15.74	99.84
2	55.24	37.26	74.81	143.33	220.59
3	54.39	310.98	241.71	123.31	109.01
4	149.44	83.89	58.78	76.96	91.88
	offnet_mou_8	offnet_mou_9	roam_ic_mou_6	roam_ic_mou_7	
roam_ic_mou_8	\				
0	0.00	NaN	NaN	NaN	
0.00					
1	304.76	53.76	0.0	0.00	
0.00					
2	208.36	118.91	0.0	0.00	
0.00					
3	71.68	113.54	0.0	54.86	
44.38					
4	124.26	45.81	0.0	0.00	
0.00					
	roam_ic_mou_9	roam_og_mou_6	roam_og_mou_7	roam_og_mou_8	
roam_og_mou_9	\				
0	NaN	NaN	NaN	0.00	
NaN					

1	0.00	0.0	0.00	0.00
0.00				
2	38.49	0.0	0.00	0.00
70.94				
3	0.00	0.0	28.09	39.04
0.00				
4	0.00	0.0	0.00	0.00
0.00				

	loc_og_t2t_mou_6	loc_og_t2t_mou_7	loc_og_t2t_mou_8
loc_og_t2t_mou_9 \			
0	NaN	NaN	0.00
NaN			
1	23.88	74.56	7.68
18.34			
2	7.19	28.74	13.58
14.39			
3	73.68	34.81	10.61
15.49			
4	50.31	149.44	83.89
58.78			

	loc_og_t2m_mou_6	loc_og_t2m_mou_7	loc_og_t2m_mou_8
loc_og_t2m_mou_9 \			
0	NaN	NaN	0.00
NaN			
1	11.51	75.94	291.86
53.76			
2	29.34	16.86	38.46
28.16			
3	107.43	83.21	22.46
65.46			
4	67.64	91.88	124.26
37.89			

	loc_og_t2f_mou_6	loc_og_t2f_mou_7	loc_og_t2f_mou_8
loc_og_t2f_mou_9 \			
0	NaN	NaN	0.00
NaN			
1	0.00	0.00	0.00
0.00			
2	24.11	21.79	15.61
22.24			
3	1.91	0.65	4.91
2.06			
4	0.00	0.00	0.00
1.93			

	loc_og_t2c_mou_6	loc_og_t2c_mou_7	loc_og_t2c_mou_8
loc_og_t2c_mou_9 \			

0	NaN	NaN	0.00
NaN			
1	0.0	2.91	0.00
0.00			
2	0.0	135.54	45.76
0.48			
3	0.0	0.00	0.00
0.00			
4	0.0	0.00	0.00
0.00			
loc_og_mou_6 loc_og_mou_7 loc_og_mou_8 loc_og_mou_9			
std_og_t2t_mou_6 \			
0	NaN	NaN	0.00
NaN			
1	35.39	150.51	299.54
0.23			
2	60.66	67.41	67.66
4.34			
3	183.03	118.68	37.99
26.23			
4	117.96	241.33	208.16
0.00			
std_og_t2t_mou_7 std_og_t2t_mou_8 std_og_t2t_mou_9			
std_og_t2m_mou_6 \			
0	NaN	0.00	NaN
NaN			
1	4.11	0.00	0.00
0.00			
2	26.49	22.58	8.76
41.81			
3	14.89	289.58	226.21
2.99			
4	0.00	0.00	0.00
9.31			
std_og_t2m_mou_7 std_og_t2m_mou_8 std_og_t2m_mou_9			
std_og_t2f_mou_6 \			
0	NaN	0.00	NaN
NaN			
1	0.46	0.13	0.00
0.00			
2	67.41	75.53	9.28
1.48			
3	1.73	6.53	9.99
0.00			
4	0.00	0.00	0.00
0.00			

std_og_t2f_mou_7 std_og_t2c_mou_6 \	std_og_t2f_mou_8	std_og_t2f_mou_9		
0	NaN	0.00	NaN	
NaN				
1	0.00	0.00	0.0	
0.0				
2	14.76	22.83	0.0	
0.0				
3	0.00	0.00	0.0	
0.0				
4	0.00	0.00	0.0	
0.0				
std_og_t2c_mou_7	std_og_t2c_mou_8	std_og_t2c_mou_9	std_og_mou_6	
\				
0	NaN	0.0	NaN	NaN
1	0.0	0.0	0.0	0.23
2	0.0	0.0	0.0	47.64
3	0.0	0.0	0.0	29.23
4	0.0	0.0	0.0	9.31
std_og_mou_7 isd_og_mou_7 \	std_og_mou_8	std_og_mou_9	isd_og_mou_6	
0	NaN	0.00	NaN	NaN
NaN				
1	4.58	0.13	0.00	0.0
0.0				
2	108.68	120.94	18.04	0.0
0.0				
3	16.63	296.11	236.21	0.0
0.0				
4	0.00	0.00	0.00	0.0
0.0				
isd_og_mou_8	isd_og_mou_9	spl_og_mou_6	spl_og_mou_7	
spl_og_mou_8 \				
0	0.0	NaN	NaN	NaN
0.00				
1	0.0	0.0	4.68	23.43
12.76				
2	0.0	0.0	46.56	236.84
96.84				
3	0.0	0.0	10.96	0.00
18.09				
4	0.0	0.0	0.00	0.00

0.00

	spl_og_mou_9	og_others_6	og_others_7	og_others_8	og_others_9	\
0	NaN	NaN	NaN	0.0	NaN	
1	0.00	0.00	0.0	0.0	0.0	
2	42.08	0.45	0.0	0.0	0.0	
3	43.29	0.00	0.0	0.0	0.0	
4	5.98	0.00	0.0	0.0	0.0	

	total_og_mou_6	total_og_mou_7	total_og_mou_8	total_og_mou_9	\
0	0.00	0.00	0.00	0.00	
1	40.31	178.53	312.44	72.11	
2	155.33	412.94	285.46	124.94	
3	223.23	135.31	352.21	362.54	
4	127.28	241.33	208.16	104.59	

	loc_ic_t2t_mou_6	loc_ic_t2t_mou_7	loc_ic_t2t_mou_8
loc_ic_t2t_mou_9 \			
0	NaN	NaN	0.16
NaN			
1	1.61	29.91	29.23
116.09			
2	115.69	71.11	67.46
148.23			
3	62.08	19.98	8.04
41.73			
4	105.68	88.49	233.81
154.56			

	loc_ic_t2m_mou_6	loc_ic_t2m_mou_7	loc_ic_t2m_mou_8
loc_ic_t2m_mou_9 \			
0	NaN	NaN	4.13
NaN			
1	17.48	65.38	375.58
56.93			
2	14.38	15.44	38.89
38.98			
3	113.96	64.51	20.28
52.86			
4	106.84	109.54	104.13
48.24			

	loc_ic_t2f_mou_6	loc_ic_t2f_mou_7	loc_ic_t2f_mou_8
loc_ic_t2f_mou_9 \			
0	NaN	NaN	1.15
NaN			
1	0.00	8.93	3.61
0.00			
2	99.48	122.29	49.63
158.19			

3	57.43	27.09	19.84
65.59			
4	1.50	0.00	0.00
0.00			

loc_ic_mou_6	loc_ic_mou_7	loc_ic_mou_8	loc_ic_mou_9
std_ic_t2t_mou_6 \			
0	NaN	NaN	5.44
NaN			NaN
1	19.09	104.23	408.43
0.00			173.03
2	229.56	208.86	155.99
72.41			345.41
3	233.48	111.59	48.18
43.48			160.19
4	214.03	198.04	337.94
0.00			202.81

std_ic_t2t_mou_7	std_ic_t2t_mou_8	std_ic_t2t_mou_9
std_ic_t2m_mou_6 \		
0	NaN	0.00
NaN		NaN
1	0.00	2.35
5.90		0.00
2	71.29	28.69
45.18		49.44
3	66.44	0.00
1.33		129.84
4	0.00	0.86
1.93		2.31

std_ic_t2m_mou_7	std_ic_t2m_mou_8	std_ic_t2m_mou_9
std_ic_t2f_mou_6 \		
0	NaN	0.00
NaN		NaN
1	0.00	12.49
0.00		15.01
2	177.01	167.09
21.73		118.18
3	38.56	4.94
1.18		13.98
4	0.25	0.00
0.00		0.00

std_ic_t2f_mou_7	std_ic_t2f_mou_8	std_ic_t2f_mou_9
std_ic_t2o_mou_6 \		
0	NaN	0.00
NaN		NaN
1	0.00	0.00
0.0		0.00

2	58.34	43.23	3.86
0.0			
3	0.00	0.00	0.00
0.0			
4	0.00	0.00	0.00
0.0			

	std_ic_t2o_mou_7	std_ic_t2o_mou_8	std_ic_t2o_mou_9	std_ic_mou_6
\				
0	NaN	0.0	NaN	NaN
1	0.0	0.0	0.0	5.90
2	0.0	0.0	0.0	139.33
3	0.0	0.0	0.0	45.99
4	0.0	0.0	0.0	1.93

	std_ic_mou_7	std_ic_mou_8	std_ic_mou_9	total_ic_mou_6
total_ic_mou_7 \				
0	NaN	0.00	NaN	0.00
0.00				
1	0.00	14.84	15.01	26.83
104.23				
2	306.66	239.03	171.49	370.04
519.53				
3	105.01	4.94	143.83	280.08
216.61				
4	0.25	0.86	2.31	216.44
198.29				

	total_ic_mou_8	total_ic_mou_9	spl_ic_mou_6	spl_ic_mou_7
spl_ic_mou_8 \				
0	5.44	0.00	NaN	NaN
0.0				
1	423.28	188.04	0.00	0.0
0.0				
2	395.03	517.74	0.21	0.0
0.0				
3	53.13	305.38	0.59	0.0
0.0				
4	338.81	205.31	0.00	0.0
0.0				

	spl_ic_mou_9	isd_ic_mou_6	isd_ic_mou_7	isd_ic_mou_8
isd_ic_mou_9 \				
0	NaN	NaN	NaN	0.0
NaN				

1	0.00	1.83	0.00	0.0
0.00				
2	0.45	0.00	0.85	0.0
0.01				
3	0.55	0.00	0.00	0.0
0.00				
4	0.18	0.00	0.00	0.0
0.00				

	ic_others_6	ic_others_7	ic_others_8	ic_others_9
total_rech_num_6	\			
0	NaN	NaN	0.0	NaN
4				
1	0.00	0.00	0.0	0.00
4				
2	0.93	3.14	0.0	0.36
5				
3	0.00	0.00	0.0	0.80
10				
4	0.48	0.00	0.0	0.00
5				

	total_rech_num_7	total_rech_num_8	total_rech_num_9
total_rech_amt_6	\		
0	3	2	6
362			
1	9	11	5
74			
2	4	2	7
168			
3	11	18	14
230			
4	6	3	4
196			

	total_rech_amt_7	total_rech_amt_8	total_rech_amt_9
max_rech_amt_6	\		
0	252	252	0
252			
1	384	283	121
44			
2	315	116	358
86			
3	310	601	410
60			
4	350	287	200
56			

	max_rech_amt_7	max_rech_amt_8	max_rech_amt_9	date_of_last_rech_6
\				

0	252	252	0	6/21/2014
1	154	65	50	6/29/2014
2	200	86	100	6/17/2014
3	50	50	50	6/28/2014
4	110	110	50	6/26/2014
	date_of_last_rech_7	date_of_last_rech_8	date_of_last_rech_9	\
0	7/16/2014	8/8/2014	9/28/2014	
1	7/31/2014	8/28/2014	9/30/2014	
2	7/24/2014	8/14/2014	9/29/2014	
3	7/31/2014	8/31/2014	9/30/2014	
4	7/28/2014	8/9/2014	9/28/2014	
	last_day_rch_amt_6	last_day_rch_amt_7	last_day_rch_amt_8	\
0	252	252	252	
1	44	23	30	
2	0	200	86	
3	30	50	50	
4	50	110	110	
	last_day_rch_amt_9	date_of_last_rech_data_6		
0	0	6/21/2014		
1	0	NaN		
2	0	NaN		
3	30	NaN		
4	50	6/4/2014		
	date_of_last_rech_data_8	date_of_last_rech_data_9	total_rech_data_6	\
0	8/8/2014	NaN	1.0	
1	8/10/2014	NaN	NaN	
2	NaN	9/17/2014	NaN	
3	NaN	NaN	NaN	
4	NaN	NaN	1.0	

	total_rech_data_7	total_rech_data_8	total_rech_data_9	
max_rech_data_6 \				
0	1.0	1.0	NaN	
252.0				
1	1.0	2.0	NaN	
NaN				
2	NaN	NaN	1.0	
NaN				
3	NaN	NaN	NaN	
NaN				
4	NaN	NaN	NaN	
56.0				
	max_rech_data_7	max_rech_data_8	max_rech_data_9	count_rech_2g_6
\				
0	252.0	252.0	NaN	0.0
1	154.0	25.0	NaN	NaN
2	NaN	NaN	46.0	NaN
3	NaN	NaN	NaN	NaN
4	NaN	NaN	NaN	1.0
	count_rech_2g_7	count_rech_2g_8	count_rech_2g_9	count_rech_3g_6
\				
0	0.0	0.0	NaN	1.0
1	1.0	2.0	NaN	NaN
2	NaN	NaN	1.0	NaN
3	NaN	NaN	NaN	NaN
4	NaN	NaN	NaN	0.0
	count_rech_3g_7	count_rech_3g_8	count_rech_3g_9	
av_rech_amt_data_6 \				
0	1.0	1.0	NaN	
252.0				
1	0.0	0.0	NaN	
NaN				
2	NaN	NaN	0.0	
NaN				
3	NaN	NaN	NaN	
NaN				
4	NaN	NaN	NaN	
56.0				

	av_rech_amt_data_7	av_rech_amt_data_8	av_rech_amt_data_9
--	--------------------	--------------------	--------------------

vol_2g_mb_6 \			
0	252.0	252.0	NaN
30.13			
1	154.0	50.0	NaN
0.00			
2	NaN	NaN	46.0
0.00			
3	NaN	NaN	NaN
0.00			
4	NaN	NaN	NaN
0.00			

	vol_2g_mb_7	vol_2g_mb_8	vol_2g_mb_9	vol_3g_mb_6	vol_3g_mb_7 \
0	1.32	5.75	0.0	83.57	150.76
1	108.07	365.47	0.0	0.00	0.00
2	0.00	0.00	0.0	0.00	0.00
3	0.00	0.00	0.0	0.00	0.00
4	0.00	0.00	0.0	0.00	0.00

	vol_3g_mb_8	vol_3g_mb_9	arpu_3g_6	arpu_3g_7	arpu_3g_8
arpu_3g_9 \					
0	109.61	0.00	212.17	212.17	212.17
NaN					
1	0.00	0.00	NaN	0.00	0.00
NaN					
2	0.00	8.42	NaN	NaN	NaN
2.84					
3	0.00	0.00	NaN	NaN	NaN
NaN					
4	0.00	0.00	0.00	NaN	NaN
NaN					

	arpu_2g_6	arpu_2g_7	arpu_2g_8	arpu_2g_9	night_pck_user_6 \
0	212.17	212.17	212.17	NaN	0.0
1	NaN	28.61	7.60	NaN	NaN
2	NaN	NaN	NaN	0.0	NaN
3	NaN	NaN	NaN	NaN	NaN
4	0.00	NaN	NaN	NaN	0.0

	night_pck_user_7	night_pck_user_8	night_pck_user_9	monthly_2g_6
\				
0	0.0	0.0	NaN	0
1	0.0	0.0	NaN	0
2	NaN	NaN	0.0	0
3	NaN	NaN	NaN	0

4	NaN	NaN	NaN	0		
	monthly_2g_7	monthly_2g_8	monthly_2g_9	sachet_2g_6	sachet_2g_7	
\						
0	0	0	0	0	0	
1	1	0	0	0	0	
2	0	0	0	0	0	
3	0	0	0	0	0	
4	0	0	0	1	0	
	sachet_2g_8	sachet_2g_9	monthly_3g_6	monthly_3g_7	monthly_3g_8	
\						
0	0	0	1	1	1	
1	2	0	0	0	0	
2	0	1	0	0	0	
3	0	0	0	0	0	
4	0	0	0	0	0	
	monthly_3g_9	sachet_3g_6	sachet_3g_7	sachet_3g_8	sachet_3g_9	\
0	0	0	0	0	0	
1	0	0	0	0	0	
2	0	0	0	0	0	
3	0	0	0	0	0	
4	0	0	0	0	0	
	fb_user_6	fb_user_7	fb_user_8	fb_user_9	aon	aug_vbc_3g
jul_vbc_3g	\					
0	1.0	1.0	1.0	NaN	968	30.4
0.0						
1	NaN	1.0	1.0	NaN	1006	0.0
0.0						
2	NaN	NaN	NaN	1.0	1103	0.0
0.0						
3	NaN	NaN	NaN	NaN	2491	0.0
0.0						
4	0.0	NaN	NaN	NaN	1526	0.0
0.0						
	jun_vbc_3g	sep_vbc_3g				

0	101.20	3.58
1	0.00	0.00
2	4.17	0.00
3	0.00	0.00
4	0.00	0.00

```
df.shape
```

```
(99999, 226)
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 99999 entries, 0 to 99998
Columns: 226 entries, mobile_number to sep_vbc_3g
dtypes: float64(179), int64(35), object(12)
memory usage: 172.4+ MB
```

```
df.describe()
```

	mobile_number	circle_id	loc_og_t2o_mou	std_og_t2o_mou	\
count	9.999900e+04	99999.0	98981.0	98981.0	
mean	7.001207e+09	109.0	0.0	0.0	
std	6.956694e+05	0.0	0.0	0.0	
min	7.000000e+09	109.0	0.0	0.0	
25%	7.000606e+09	109.0	0.0	0.0	
50%	7.001205e+09	109.0	0.0	0.0	
75%	7.001812e+09	109.0	0.0	0.0	
max	7.002411e+09	109.0	0.0	0.0	

	loc_ic_t2o_mou	arpu_6	arpu_7	arpu_8	arpu_9	\
count	98981.0	99999.000000	99999.000000	99999.000000	99999.000000	
mean	0.0	282.987358	278.536648	279.154731	261.645069	
std	0.0	328.439770	338.156291	344.474791	341.998630	
min	0.0	-2258.709000	-2014.045000	-945.808000	-1899.505000	
25%	0.0	93.411500	86.980500	84.126000	62.685000	
50%	0.0	197.704000	191.640000	192.080000	176.849000	
75%	0.0	371.060000	365.344500	369.370500	353.466500	
max	0.0	27731.088000	35145.834000	33543.624000	38805.617000	

	onnet_mou_6	onnet_mou_7	onnet_mou_8	onnet_mou_9	offnet_mou_6	\
--	-------------	-------------	-------------	-------------	--------------	---

count	96062.000000	96140.000000	94621.000000	92254.000000
96062.000000				
mean	132.395875	133.670805	133.018098	130.302327
197.935577				
std	297.207406	308.794148	308.951589	308.477668
316.851613				
min	0.000000	0.000000	0.000000	0.000000
0.000000				
25%	7.380000	6.660000	6.460000	5.330000
34.730000				
50%	34.310000	32.330000	32.360000	29.840000
96.310000				
75%	118.740000	115.595000	115.860000	112.130000
231.860000				
max	7376.710000	8157.780000	10752.560000	10427.460000
8362.360000				

	offnet_mou_7	offnet_mou_8	offnet_mou_9	roam_ic_mou_6
roam_ic_mou_7 \				
count	96140.000000	94621.000000	92254.000000	96062.000000
96140.000000				
mean	197.045133	196.574803	190.337222	9.950013
7.149898				
std	325.862803	327.170662	319.396092	72.825411
73.447948				
min	0.000000	0.000000	0.000000	0.000000
0.000000				
25%	32.190000	31.630000	27.130000	0.000000
0.000000				
50%	91.735000	92.140000	87.290000	0.000000
0.000000				
75%	226.815000	228.260000	220.505000	0.000000
0.000000				
max	9667.130000	14007.340000	10310.760000	13724.380000
15371.040000				

	roam_ic_mou_8	roam_ic_mou_9	roam_og_mou_6	roam_og_mou_7 \
count	94621.000000	92254.000000	96062.000000	96140.000000
mean	7.292981	6.343841	13.911337	9.818732
std	68.402466	57.137537	71.443196	58.455762
min	0.000000	0.000000	0.000000	0.000000
25%	0.000000	0.000000	0.000000	0.000000
50%	0.000000	0.000000	0.000000	0.000000
75%	0.000000	0.000000	0.000000	0.000000
max	13095.360000	8464.030000	3775.110000	2812.040000

	roam_og_mou_8	roam_og_mou_9	loc_og_t2t_mou_6
loc_og_t2t_mou_7 \			
count	94621.000000	92254.000000	96062.000000
96140.000000			

mean	9.971890	8.555519	47.100763
46.473010			
std	64.713221	58.438186	150.856393
155.318705			
min	0.000000	0.000000	0.000000
0.000000			
25%	0.000000	0.000000	1.660000
1.630000			
50%	0.000000	0.000000	11.910000
11.610000			
75%	0.000000	0.000000	40.960000
39.910000			
max	5337.040000	4428.460000	6431.330000
7400.660000			

	loc_og_t2t_mou_8	loc_og_t2t_mou_9	loc_og_t2m_mou_6
loc_og_t2m_mou_7 \			
count	94621.000000	92254.000000	96062.000000
96140.000000			
mean	45.887806	44.584446	93.342088
91.397131			
std	151.184830	147.995390	162.780544
157.492308			
min	0.000000	0.000000	0.000000
0.000000			
25%	1.600000	1.360000	9.880000
10.025000			
50%	11.730000	11.260000	41.030000
40.430000			
75%	40.110000	39.280000	110.390000
107.560000			
max	10752.560000	10389.240000	4729.740000
4557.140000			

	loc_og_t2m_mou_8	loc_og_t2m_mou_9	loc_og_t2f_mou_6
loc_og_t2f_mou_7 \			
count	94621.000000	92254.000000	96062.000000
96140.000000			
mean	91.755128	90.463192	3.751013
3.792985			
std	156.537048	158.681454	14.230438
14.264986			
min	0.000000	0.000000	0.000000
0.000000			
25%	9.810000	8.810000	0.000000
0.000000			
50%	40.360000	39.120000	0.000000
0.000000			
75%	109.090000	106.810000	2.080000

2.090000
max 4961.330000 4429.880000 1466.030000
1196.430000

	loc_og_t2f_mou_8	loc_og_t2f_mou_9	loc_og_t2c_mou_6
loc_og_t2c_mou_7 \			
count	94621.000000	92254.000000	96062.000000
96140.000000			
mean	3.677991	3.655123	1.123056
1.368500			
std	13.270996	13.457549	5.448946
7.533445			
min	0.000000	0.000000	0.000000
0.000000			
25%	0.000000	0.000000	0.000000
0.000000			
50%	0.000000	0.000000	0.000000
0.000000			
75%	2.040000	1.940000	0.000000
0.000000			
max	928.490000	927.410000	342.860000
916.240000			

	loc_og_t2c_mou_8	loc_og_t2c_mou_9	loc_og_mou_6	loc_og_mou_7
\				
count	94621.000000	92254.000000	96062.000000	96140.000000
mean	1.433821	1.232726	144.201175	141.670476
std	6.783335	5.619021	251.751489	248.731086
min	0.000000	0.000000	0.000000	0.000000
25%	0.000000	0.000000	17.110000	17.480000
50%	0.000000	0.000000	65.110000	63.685000
75%	0.000000	0.000000	168.270000	164.382500
max	502.090000	339.840000	10643.380000	7674.780000

	loc_og_mou_8	loc_og_mou_9	std_og_t2t_mou_6	std_og_t2t_mou_7
\				
count	94621.000000	92254.000000	96062.000000	96140.000000
mean	141.328209	138.709970	79.829870	83.299598
std	245.914311	245.934517	252.476533	263.631042
min	0.000000	0.000000	0.000000	0.000000

25%	17.110000	15.560000	0.000000	0.000000
50%	63.730000	61.840000	0.000000	0.000000
75%	166.110000	162.225000	30.807500	31.132500
max	11039.910000	11099.260000	7366.580000	8133.660000

	std_og_t2t_mou_8	std_og_t2t_mou_9	std_og_t2m_mou_6
std_og_t2m_mou_7 \			
count	94621.000000	92254.000000	96062.000000
96140.000000			
mean	83.282673	82.342919	87.299624
90.804137			
std	265.486090	267.184991	255.617850
269.347911			
min	0.000000	0.000000	0.000000
0.000000			
25%	0.000000	0.000000	0.000000
0.000000			
50%	0.000000	0.000000	3.950000
3.635000			
75%	30.580000	28.230000	53.290000
54.040000			
max	8014.430000	9382.580000	8314.760000
9284.740000			

	std_og_t2m_mou_8	std_og_t2m_mou_9	std_og_t2f_mou_6
std_og_t2f_mou_7 \			
count	94621.000000	92254.000000	96062.000000
96140.000000			
mean	89.838390	86.276622	1.129011
1.115010			
std	271.757783	261.407396	7.984970
8.599406			
min	0.000000	0.000000	0.000000
0.000000			
25%	0.000000	0.000000	0.000000
0.000000			
50%	3.310000	2.500000	0.000000
0.000000			
75%	52.490000	48.560000	0.000000
0.000000			
max	13950.040000	10223.430000	628.560000
544.630000			

	std_og_t2f_mou_8	std_og_t2f_mou_9	std_og_t2c_mou_6
std_og_t2c_mou_7 \			

count	94621.000000	92254.000000	96062.0
96140.0			
mean	1.067792	1.042362	0.0
0.0			
std	7.905971	8.261770	0.0
0.0			
min	0.000000	0.000000	0.0
0.0			
25%	0.000000	0.000000	0.0
0.0			
50%	0.000000	0.000000	0.0
0.0			
75%	0.000000	0.000000	0.0
0.0			
max	516.910000	808.490000	0.0
0.0			

	std_og_t2c_mou_8	std_og_t2c_mou_9	std_og_mou_6	std_og_mou_7
\				
count	94621.0	92254.0	96062.000000	96140.000000
mean	0.0	0.0	168.261218	175.221436
std	0.0	0.0	389.948499	408.922934
min	0.0	0.0	0.000000	0.000000
25%	0.0	0.0	0.000000	0.000000
50%	0.0	0.0	11.640000	11.090000
75%	0.0	0.0	144.837500	150.615000
max	0.0	0.0	8432.990000	10936.730000

	std_og_mou_8	std_og_mou_9	isd_og_mou_6	isd_og_mou_7
isd_og_mou_8				
\				
count	94621.000000	92254.000000	96062.000000	96140.000000
94621.000000				
mean	174.191498	169.664466	0.798277	0.776572
0.791247				
std	411.633049	405.138658	25.765248	25.603052
25.544471				
min	0.000000	0.000000	0.000000	0.000000
0.000000				
25%	0.000000	0.000000	0.000000	0.000000
0.000000				
50%	10.410000	8.410000	0.000000	0.000000
0.000000				

75%	147.940000	142.105000	0.000000	0.000000
0.000000				
max	13980.060000	11495.310000	5900.660000	5490.280000
5681.540000				

	isd_og_mou_9	spl_og_mou_6	spl_og_mou_7	spl_og_mou_8
spl_og_mou_9 \				
count	92254.000000	96062.000000	96140.000000	94621.000000
92254.000000				
mean	0.723892	3.916811	4.978279	5.053769
4.412767				
std	21.310751	14.936449	20.661570	17.855111
16.328227				
min	0.000000	0.000000	0.000000	0.000000
0.000000				
25%	0.000000	0.000000	0.000000	0.000000
0.000000				
50%	0.000000	0.000000	0.000000	0.000000
0.000000				
75%	0.000000	2.430000	3.710000	3.990000
3.230000				
max	4244.530000	1023.210000	2372.510000	1390.880000
1635.710000				

	og_others_6	og_others_7	og_others_8	og_others_9
total_og_mou_6 \				
count	96062.000000	96140.000000	94621.000000	92254.000000
99999.000000				
mean	0.454157	0.030235	0.033372	0.047456
305.133424				
std	4.125911	2.161717	2.323464	3.635466
463.419481				
min	0.000000	0.000000	0.000000	0.000000
0.000000				
25%	0.000000	0.000000	0.000000	0.000000
44.740000				
50%	0.000000	0.000000	0.000000	0.000000
145.140000				
75%	0.000000	0.000000	0.000000	0.000000
372.860000				
max	800.890000	370.130000	394.930000	787.790000
10674.030000				

	total_og_mou_7	total_og_mou_8	total_og_mou_9
loc_ic_t2t_mou_6 \			
count	99999.000000	99999.000000	99999.000000
96062.000000			
mean	310.231175	304.119513	289.279198
47.922365			
std	480.031178	478.150031	468.980002

140.258485			
min	0.000000	0.000000	0.000000
0.000000			
25%	43.010000	38.580000	25.510000
2.990000			
50%	141.530000	138.610000	125.460000
15.690000			
75%	378.570000	369.900000	353.480000
46.840000			
max	11365.310000	14043.060000	11517.730000
6626.930000			

	loc_ic_t2t_mou_7	loc_ic_t2t_mou_8	loc_ic_t2t_mou_9
loc_ic_t2m_mou_6 \			
count	96140.000000	94621.000000	92254.000000
96062.000000			
mean	47.990520	47.211362	46.281794
107.475650			
std	145.795055	137.239552	140.130610
171.713903			
min	0.000000	0.000000	0.000000
0.000000			
25%	3.230000	3.280000	3.290000
17.290000			
50%	15.740000	16.030000	15.660000
56.490000			
75%	45.810000	46.290000	45.180000
132.387500			
max	9324.660000	10696.230000	10598.830000
4693.860000			

	loc_ic_t2m_mou_7	loc_ic_t2m_mou_8	loc_ic_t2m_mou_9
loc_ic_t2f_mou_6 \			
count	96140.000000	94621.000000	92254.000000
96062.000000			
mean	107.120493	108.460515	106.155471
12.084305			
std	169.423620	169.723759	165.492803
40.140895			
min	0.000000	0.000000	0.000000
0.000000			
25%	18.590000	18.930000	18.560000
0.000000			
50%	57.080000	58.240000	56.610000
0.880000			
75%	130.960000	133.930000	130.490000
8.140000			
max	4455.830000	6274.190000	5463.780000
1872.340000			

	loc_ic_t2f_mou_7	loc_ic_t2f_mou_8	loc_ic_t2f_mou_9
loc_ic_mou_6 \			
count	96140.000000	94621.000000	92254.000000
96062.000000			
mean	12.599697	11.751834	12.173105
167.491059			
std	42.977442	39.125379	43.840776
254.124029			
min	0.000000	0.000000	0.000000
0.000000			
25%	0.000000	0.000000	0.000000
30.390000			
50%	0.930000	0.930000	0.960000
92.160000			
75%	8.282500	8.110000	8.140000
208.075000			
max	1983.010000	2433.060000	4318.280000
7454.630000			

	loc_ic_mou_7	loc_ic_mou_8	loc_ic_mou_9	std_ic_t2t_mou_6 \
count	96140.000000	94621.000000	92254.000000	96062.000000
mean	167.719540	167.432575	164.619293	9.575993
std	256.242707	250.025523	249.845070	54.330607
min	0.000000	0.000000	0.000000	0.000000
25%	32.460000	32.740000	32.290000	0.000000
50%	92.550000	93.830000	91.640000	0.000000
75%	205.837500	207.280000	202.737500	4.060000
max	9669.910000	10830.160000	10796.290000	5459.560000

	std_ic_t2t_mou_7	std_ic_t2t_mou_8	std_ic_t2t_mou_9
std_ic_t2m_mou_6 \			
count	96140.000000	94621.000000	92254.000000
96062.000000			
mean	10.011904	9.883921	9.432479
20.722240			
std	57.411971	55.073186	53.376273
80.793414			
min	0.000000	0.000000	0.000000
0.000000			
25%	0.000000	0.000000	0.000000
0.000000			
50%	0.000000	0.000000	0.000000
2.030000			
75%	4.230000	4.080000	3.510000
15.030000			
max	5800.930000	4309.290000	3819.830000
5647.160000			

	std_ic_t2m_mou_7	std_ic_t2m_mou_8	std_ic_t2m_mou_9
std_ic_t2f_mou_6 \			

count	96140.000000	94621.000000	92254.000000
96062.000000			
mean	21.656415	21.183211	19.620913
2.156397			
std	86.521393	83.683565	74.913050
16.495594			
min	0.000000	0.000000	0.000000
0.000000			
25%	0.000000	0.000000	0.000000
0.000000			
50%	2.040000	2.030000	1.740000
0.000000			
75%	15.740000	15.360000	14.260000
0.000000			
max	6141.880000	5645.860000	5689.760000
1351.110000			

	std_ic_t2f_mou_7	std_ic_t2f_mou_8	std_ic_t2f_mou_9
std_ic_t2o_mou_6 \			
count	96140.000000	94621.000000	92254.000000
96062.0			
mean	2.216923	2.085004	2.173419
0.0			
std	16.454061	15.812580	15.978601
0.0			
min	0.000000	0.000000	0.000000
0.0			
25%	0.000000	0.000000	0.000000
0.0			
50%	0.000000	0.000000	0.000000
0.0			
75%	0.000000	0.000000	0.000000
0.0			
max	1136.080000	1394.890000	1431.960000
0.0			

	std_ic_t2o_mou_7	std_ic_t2o_mou_8	std_ic_t2o_mou_9
std_ic_mou_6 \			
count	96140.0	94621.0	92254.0
96062.000000			
mean	0.0	0.0	0.0
32.457179			
std	0.0	0.0	0.0
106.283386			
min	0.0	0.0	0.0
0.000000			
25%	0.0	0.0	0.0
0.000000			
50%	0.0	0.0	0.0

5.890000			
75%	0.0	0.0	0.0
26.930000			
max	0.0	0.0	0.0
5712.110000			

	std_ic_mou_7	std_ic_mou_8	std_ic_mou_9	total_ic_mou_6 \
count	96140.000000	94621.000000	92254.000000	99999.000000
mean	33.887833	33.154735	31.229344	200.130037
std	113.720168	110.127008	101.982303	291.651671
min	0.000000	0.000000	0.000000	0.000000
25%	0.000000	0.010000	0.000000	38.530000
50%	5.960000	5.880000	5.380000	114.740000
75%	28.310000	27.710000	25.690000	251.670000
max	6745.760000	5957.140000	5956.660000	7716.140000

	total_ic_mou_7	total_ic_mou_8	total_ic_mou_9	spl_ic_mou_6 \
count	99999.000000	99999.000000	99999.000000	96062.000000
mean	202.853055	198.750783	189.214260	0.061557
std	298.124954	289.321094	284.823024	0.160920
min	0.000000	0.000000	0.000000	0.000000
25%	41.190000	38.290000	32.370000	0.000000
50%	116.340000	114.660000	105.890000	0.000000
75%	250.660000	248.990000	236.320000	0.000000
max	9699.010000	10830.380000	10796.590000	19.760000

	spl_ic_mou_7	spl_ic_mou_8	spl_ic_mou_9	isd_ic_mou_6
isd_ic_mou_7 \				
count	96140.000000	94621.000000	92254.000000	96062.000000
96140.000000				
mean	0.033585	0.040361	0.163137	7.460608
8.334936				
std	0.155725	0.146147	0.527860	59.722948
65.219829				
min	0.000000	0.000000	0.000000	0.000000
0.000000				
25%	0.000000	0.000000	0.000000	0.000000
0.000000				
50%	0.000000	0.000000	0.000000	0.000000
0.000000				
75%	0.000000	0.000000	0.060000	0.000000
0.000000				
max	21.330000	16.860000	62.380000	6789.410000
5289.540000				

	isd_ic_mou_8	isd_ic_mou_9	ic_others_6	ic_others_7
ic_others_8 \				
count	94621.000000	92254.000000	96062.000000	96140.000000
94621.000000				
mean	8.442001	8.063003	0.854656	1.012960

0.970800				
std	63.813098	63.505379	11.955164	12.673099
13.284348				
min	0.000000	0.000000	0.000000	0.000000
0.000000				
25%	0.000000	0.000000	0.000000	0.000000
0.000000				
50%	0.000000	0.000000	0.000000	0.000000
0.000000				
75%	0.000000	0.000000	0.000000	0.000000
0.000000				
max	4127.010000	5057.740000	1362.940000	1495.940000
2327.510000				

	ic_others_9	total_rech_num_6	total_rech_num_7
total_rech_num_8 \			
count	92254.000000	99999.000000	99999.000000
99999.000000			
mean	1.017162	7.558806	7.700367
7.212912			
std	12.381172	7.078405	7.070422
7.203753			
min	0.000000	0.000000	0.000000
0.000000			
25%	0.000000	3.000000	3.000000
3.000000			
50%	0.000000	6.000000	6.000000
5.000000			
75%	0.000000	9.000000	10.000000
9.000000			
max	1005.230000	307.000000	138.000000
196.000000			

	total_rech_num_9	total_rech_amt_6	total_rech_amt_7
total_rech_amt_8 \			
count	99999.000000	99999.000000	99999.000000
99999.000000			
mean	6.893019	327.514615	322.962970
324.157122			
std	7.096261	398.019701	408.114237
416.540455			
min	0.000000	0.000000	0.000000
0.000000			
25%	3.000000	109.000000	100.000000
90.000000			
50%	5.000000	230.000000	220.000000
225.000000			
75%	9.000000	437.500000	428.000000
434.500000			

max	131.000000	35190.000000	40335.000000
45320.000000			

	total_rech_amt_9	max_rech_amt_6	max_rech_amt_7
max_rech_amt_8 \			
count	99999.000000	99999.000000	99999.000000
99999.000000			
mean	303.345673	104.637486	104.752398
107.728207			
std	404.588583	120.614894	124.523970
126.902505			
min	0.000000	0.000000	0.000000
0.000000			
25%	52.000000	30.000000	30.000000
30.000000			
50%	200.000000	110.000000	110.000000
98.000000			
75%	415.000000	120.000000	128.000000
144.000000			
max	37235.000000	4010.000000	4010.000000
4449.000000			

	max_rech_amt_9	last_day_rch_amt_6	last_day_rch_amt_7 \
count	99999.000000	99999.000000	99999.000000
mean	101.943889	63.156252	59.385804
std	125.375109	97.356649	95.915385
min	0.000000	0.000000	0.000000
25%	28.000000	0.000000	0.000000
50%	61.000000	30.000000	30.000000
75%	144.000000	110.000000	110.000000
max	3399.000000	4010.000000	4010.000000

	last_day_rch_amt_8	last_day_rch_amt_9	total_rech_data_6 \
count	99999.000000	99999.000000	25153.000000
mean	62.641716	43.901249	2.463802
std	104.431816	90.809712	2.789128
min	0.000000	0.000000	1.000000
25%	0.000000	0.000000	1.000000
50%	30.000000	0.000000	1.000000
75%	130.000000	50.000000	3.000000
max	4449.000000	3399.000000	61.000000

	total_rech_data_7	total_rech_data_8	total_rech_data_9 \
count	25571.000000	26339.000000	25922.000000
mean	2.666419	2.651999	2.441170
std	3.031593	3.074987	2.516339
min	1.000000	1.000000	1.000000
25%	1.000000	1.000000	1.000000
50%	1.000000	1.000000	2.000000
75%	3.000000	3.000000	3.000000

max	54.000000	60.000000	84.000000
max_rech_data_6	max_rech_data_7	max_rech_data_8	
max_rech_data_9 \			
count	25153.000000	25571.000000	26339.000000
25922.000000			
mean	126.393392	126.729459	125.717301
124.94144			
std	108.477235	109.765267	109.437851
111.36376			
min	1.000000	1.000000	1.000000
1.000000			
25%	25.000000	25.000000	25.000000
25.000000			
50%	145.000000	145.000000	145.000000
145.000000			
75%	177.000000	177.000000	179.000000
179.000000			
max	1555.000000	1555.000000	1555.000000
1555.000000			

count_rech_2g_6	count_rech_2g_7	count_rech_2g_8
count_rech_2g_9 \		
count	25153.000000	25571.000000
25922.000000		26339.000000
mean	1.864668	2.044699
1.781807		2.016288
std	2.570254	2.768332
2.214701		2.720132
min	0.000000	0.000000
0.000000		0.000000
25%	1.000000	1.000000
1.000000		1.000000
50%	1.000000	1.000000
1.000000		1.000000
75%	2.000000	2.000000
2.000000		2.000000
max	42.000000	48.000000
40.000000		44.000000

count_rech_3g_6	count_rech_3g_7	count_rech_3g_8
count_rech_3g_9 \		
count	25153.000000	25571.000000
25922.000000		26339.000000
mean	0.599133	0.621720
0.659363		0.635711
std	1.274428	1.394524
1.411513		1.422827
min	0.000000	0.000000
0.000000		0.000000

25%	0.000000	0.000000	0.000000
0.000000			
50%	0.000000	0.000000	0.000000
0.000000			
75%	1.000000	1.000000	1.000000
1.000000			
max	29.000000	35.000000	45.000000
49.000000			

	av_rech_amt_data_6	av_rech_amt_data_7	av_rech_amt_data_8	\
count	25153.000000	25571.000000	26339.000000	
mean	192.600982	200.981292	197.526489	
std	192.646318	196.791224	191.301305	
min	1.000000	0.500000	0.500000	
25%	82.000000	92.000000	87.000000	
50%	154.000000	154.000000	154.000000	
75%	252.000000	252.000000	252.000000	
max	7546.000000	4365.000000	4076.000000	

	av_rech_amt_data_9	vol_2g_mb_6	vol_2g_mb_7	vol_2g_mb_8	\
count	25922.000000	99999.000000	99999.000000	99999.000000	
mean	192.734315	51.904956	51.229937	50.170154	
std	188.400286	213.356445	212.302217	212.347892	
min	1.000000	0.000000	0.000000	0.000000	
25%	69.000000	0.000000	0.000000	0.000000	
50%	164.000000	0.000000	0.000000	0.000000	
75%	252.000000	0.000000	0.000000	0.000000	
max	4061.000000	10285.900000	7873.550000	11117.610000	

	vol_2g_mb_9	vol_3g_mb_6	vol_3g_mb_7	vol_3g_mb_8	
vol_3g_mb_9	\				
count	99999.000000	99999.000000	99999.000000	99999.000000	
99999.000000					
mean	44.719701	121.396219	128.995847	135.410689	
136.056613					
std	198.653570	544.247227	541.494013	558.775335	
577.394194					
min	0.000000	0.000000	0.000000	0.000000	
0.000000					
25%	0.000000	0.000000	0.000000	0.000000	
0.000000					
50%	0.000000	0.000000	0.000000	0.000000	
0.000000					
75%	0.000000	0.000000	0.000000	0.000000	
0.000000					
max	8993.950000	45735.400000	28144.120000	30036.060000	
39221.270000					

	arpu_3g_6	arpu_3g_7	arpu_3g_8	arpu_3g_9	
arpu_2g_6	\				

count	25153.000000	25571.000000	26339.000000	25922.000000
25153.000000				
mean	89.555057	89.384120	91.173849	100.264116
86.398003				
std	193.124653	195.893924	188.180936	216.291992
172.767523				
min	-30.820000	-26.040000	-24.490000	-71.090000
35.830000				
25%	0.000000	0.000000	0.000000	0.000000
0.000000				
50%	0.480000	0.420000	0.880000	2.605000
10.830000				
75%	122.070000	119.560000	122.070000	140.010000
122.070000				
max	6362.280000	4980.900000	3716.900000	13884.310000
6433.760000				

	arpu_2g_7	arpu_2g_8	arpu_2g_9	night_pck_user_6 \
count	25571.000000	26339.000000	25922.000000	25153.000000
mean	85.914450	86.599478	93.712026	0.025086
std	176.379871	168.247852	171.384224	0.156391
min	-15.480000	-55.830000	-45.740000	0.000000
25%	0.000000	0.000000	0.000000	0.000000
50%	8.810000	9.270000	14.800000	0.000000
75%	122.070000	122.070000	140.010000	0.000000
max	4809.360000	3483.170000	3467.170000	1.000000

	night_pck_user_7	night_pck_user_8	night_pck_user_9
monthly_2g_6 \			
count	25571.000000	26339.000000	25922.000000
99999.000000			
mean	0.023034	0.020844	0.015971
0.079641			
std	0.150014	0.142863	0.125366
0.295058			
min	0.000000	0.000000	0.000000
0.000000			
25%	0.000000	0.000000	0.000000
0.000000			
50%	0.000000	0.000000	0.000000
0.000000			
75%	0.000000	0.000000	0.000000
0.000000			
max	1.000000	1.000000	1.000000
4.000000			

	monthly_2g_7	monthly_2g_8	monthly_2g_9	sachet_2g_6
sachet_2g_7 \				
count	99999.000000	99999.000000	99999.000000	99999.000000
99999.000000				

mean	0.083221	0.081001	0.068781	0.389384
0.439634				
std	0.304395	0.299568	0.278120	1.497320
1.636230				
min	0.000000	0.000000	0.000000	0.000000
0.000000				
25%	0.000000	0.000000	0.000000	0.000000
0.000000				
50%	0.000000	0.000000	0.000000	0.000000
0.000000				
75%	0.000000	0.000000	0.000000	0.000000
0.000000				
max	5.000000	5.000000	4.000000	42.000000
48.000000				

	sachet_2g_8	sachet_2g_9	monthly_3g_6	monthly_3g_7
monthly_3g_8 \				
count	99999.000000	99999.000000	99999.000000	99999.000000
99999.000000				
mean	0.450075	0.393104	0.075921	0.078581
0.082941				
std	1.630263	1.347140	0.363371	0.387231
0.384947				
min	0.000000	0.000000	0.000000	0.000000
0.000000				
25%	0.000000	0.000000	0.000000	0.000000
0.000000				
50%	0.000000	0.000000	0.000000	0.000000
0.000000				
75%	0.000000	0.000000	0.000000	0.000000
0.000000				
max	44.000000	40.000000	14.000000	16.000000
16.000000				

	monthly_3g_9	sachet_3g_6	sachet_3g_7	sachet_3g_8
sachet_3g_9 \				
count	99999.000000	99999.000000	99999.000000	99999.000000
99999.000000				
mean	0.086341	0.074781	0.080401	0.084501
0.084581				
std	0.384978	0.568344	0.628334	0.660234
0.650457				
min	0.000000	0.000000	0.000000	0.000000
0.000000				
25%	0.000000	0.000000	0.000000	0.000000
0.000000				
50%	0.000000	0.000000	0.000000	0.000000
0.000000				
75%	0.000000	0.000000	0.000000	0.000000

0.000000				
max	11.000000	29.000000	35.000000	41.000000
49.000000				
	fb_user_6	fb_user_7	fb_user_8	fb_user_9
aon \				
count	25153.000000	25571.000000	26339.000000	25922.000000
99999.000000				
mean	0.914404	0.908764	0.890808	0.860968
1219.854749				
std	0.279772	0.287950	0.311885	0.345987
954.733842				
min	0.000000	0.000000	0.000000	0.000000
180.000000				
25%	1.000000	1.000000	1.000000	1.000000
467.000000				
50%	1.000000	1.000000	1.000000	1.000000
863.000000				
75%	1.000000	1.000000	1.000000	1.000000
1807.500000				
max	1.000000	1.000000	1.000000	1.000000
4337.000000				
	aug_vbc_3g	jul_vbc_3g	jun_vbc_3g	sep_vbc_3g
count	99999.000000	99999.000000	99999.000000	99999.000000
mean	68.170248	66.839062	60.021204	3.299373
std	267.580450	271.201856	253.938223	32.408353
min	0.000000	0.000000	0.000000	0.000000
25%	0.000000	0.000000	0.000000	0.000000
50%	0.000000	0.000000	0.000000	0.000000
75%	0.000000	0.000000	0.000000	0.000000
max	12916.220000	9165.600000	11166.210000	2618.570000

Handling missing values

Handling missing values in columns

```
# Cheking percent of missing values in columns
df_missing_columns =
(round(((df.isnull().sum())/len(df.index))*100),2).to_frame('null')).so
rt_values('null', ascending=False)
df_missing_columns
```

	null
arpu_3g_6	74.85
night_pck_user_6	74.85
total_rech_data_6	74.85
arpu_2g_6	74.85
max_rech_data_6	74.85

fb_user_6	74.85
av_rech_amt_data_6	74.85
date_of_last_rech_data_6	74.85
count_rech_2g_6	74.85
count_rech_3g_6	74.85
date_of_last_rech_data_7	74.43
total_rech_data_7	74.43
fb_user_7	74.43
max_rech_data_7	74.43
night_pck_user_7	74.43
count_rech_2g_7	74.43
av_rech_amt_data_7	74.43
arpu_2g_7	74.43
count_rech_3g_7	74.43
arpu_3g_7	74.43
total_rech_data_9	74.08
count_rech_3g_9	74.08
fb_user_9	74.08
max_rech_data_9	74.08
arpu_3g_9	74.08
date_of_last_rech_data_9	74.08
night_pck_user_9	74.08
arpu_2g_9	74.08
count_rech_2g_9	74.08
av_rech_amt_data_9	74.08
...	...
circle_id	0.00
total_og_mou_8	0.00
vol_3g_mb_7	0.00
total_og_mou_7	0.00
total_og_mou_6	0.00
arpu_9	0.00
arpu_8	0.00
arpu_7	0.00
arpu_6	0.00
last_date_of_month_6	0.00
total_rech_num_8	0.00
total_rech_num_9	0.00
total_rech_amt_6	0.00
total_rech_amt_7	0.00
vol_3g_mb_6	0.00
vol_2g_mb_9	0.00
vol_2g_mb_8	0.00
vol_2g_mb_7	0.00
vol_2g_mb_6	0.00
last_day_rch_amt_9	0.00
last_day_rch_amt_8	0.00
last_day_rch_amt_7	0.00
last_day_rch_amt_6	0.00


```

max_rech_amt_9      0.00
max_rech_amt_8      0.00
max_rech_amt_7      0.00
max_rech_amt_6      0.00
total_rech_amt_9     0.00
total_rech_amt_8     0.00
sep_vbc_3g          0.00

```

```
[226 rows x 1 columns]
```

```
# List the columns having more than 30% missing values
```

```
col_list_missing_30 =
```

```
list(df_missing_columns.index[df_missing_columns['null'] > 30])
```

```
# Delete the columns having more than 30% missing values
```

```
df = df.drop(col_list_missing_30, axis=1)
```

```
df.shape
```

```
(99999, 186)
```

Deleting the date columns as the date columns are not required in our analysis

```
# List the date columns
```

```
date_cols = [k for k in df.columns.to_list() if 'date' in k]
```

```
print(date_cols)
```

```
['last_date_of_month_6', 'last_date_of_month_7',
'last_date_of_month_8', 'last_date_of_month_9', 'date_of_last_rech_6',
'date_of_last_rech_7', 'date_of_last_rech_8', 'date_of_last_rech_9']
```

```
# Dropping date columns
```

```
df = df.drop(date_cols, axis=1)
```

Dropping circle_id column as this column has only one unique value. Hence there will be no impact of this column on the data analysis.

```
# Drop circle_id column
```

```
df = df.drop('circle_id', axis=1)
```

```
df.shape
```

```
(99999, 177)
```

Filter high-value customers

Creating column avg_rech_amt_6_7 by summing up total recharge amount of month 6 and 7. Then taking the average of the sum.

```
df['avg_rech_amt_6_7'] = (df['total_rech_amt_6'] +
df['total_rech_amt_7'])/2
```

Finding the 70th percentile of the avg_rech_amt_6_7

```
X = df['avg_rech_amt_6_7'].quantile(0.7)
X
```

368.5

Filter the customers, who have recharged more than or equal to X.

```
df = df[df['avg_rech_amt_6_7'] >= X]
df.head()
```

	mobile_number	loc_og_t2o_mou	std_og_t2o_mou	loc_ic_t2o_mou
arpu_6 \				
7	7000701601	0.0	0.0	0.0
1069.180				
8	7001524846	0.0	0.0	0.0
378.721				
13	7002191713	0.0	0.0	0.0
492.846				
16	7000875565	0.0	0.0	0.0
430.975				
17	7000187447	0.0	0.0	0.0
690.008				

	arpu_7	arpu_8	arpu_9	onnet_mou_6	onnet_mou_7	onnet_mou_8
\						
7	1349.850	3171.480	500.000	57.84	54.68	52.29
8	492.223	137.362	166.787	413.69	351.03	35.08
13	205.671	593.260	322.732	501.76	108.39	534.24
16	299.869	187.894	206.490	50.51	74.01	70.61
17	18.980	25.499	257.583	1185.91	9.28	7.79

	onnet_mou_9	offnet_mou_6	offnet_mou_7	offnet_mou_8
offnet_mou_9 \				
7	NaN	453.43	567.16	325.91
NaN				
8	33.46	94.66	80.63	136.48
108.71				
13	244.81	413.31	119.28	482.46
214.06				
16	31.34	296.29	229.74	162.76
224.39				
17	558.51	61.64	0.00	5.54
87.89				

	roam_ic_mou_6	roam_ic_mou_7	roam_ic_mou_8	roam_ic_mou_9
roam_og_mou_6 \				
7	16.23	33.49	31.64	NaN
23.74				
8	0.00	0.00	0.00	0.00
0.00				
13	23.53	144.24	72.11	136.78
7.98				
16	0.00	2.83	0.00	0.00
0.00				
17	0.00	4.76	4.81	0.00
0.00				

	roam_og_mou_7	roam_og_mou_8	roam_og_mou_9	loc_og_t2t_mou_6 \
7	12.59	38.06	NaN	51.39
8	0.00	0.00	0.00	297.13
13	35.26	1.44	12.78	49.63
16	17.74	0.00	0.00	42.61
17	8.46	13.34	17.98	38.99

	loc_og_t2t_mou_7	loc_og_t2t_mou_8	loc_og_t2t_mou_9
loc_og_t2m_mou_6 \			
7	31.38	40.28	NaN
308.63			
8	217.59	12.49	26.13
80.96			
13	6.19	36.01	6.14
151.13			
16	65.16	67.38	26.88
273.29			
17	0.00	0.00	36.41
58.54			

	loc_og_t2m_mou_7	loc_og_t2m_mou_8	loc_og_t2m_mou_9
loc_og_t2f_mou_6 \			
7	447.38	162.28	NaN
62.13			
8	70.58	50.54	34.58
0.00			
13	47.28	294.46	108.24
4.54			
16	145.99	128.28	201.49
0.00			
17	0.00	0.00	9.38
0.00			

	loc_og_t2f_mou_7	loc_og_t2f_mou_8	loc_og_t2f_mou_9
loc_og_t2c_mou_6 \			
7	55.14	53.23	NaN

0.0			
8	0.00	0.00	0.00
0.0			
13	0.00	23.51	5.29
0.0			
16	4.48	10.26	4.66
0.0			
17	0.00	0.00	0.00
0.0			

	loc_og_t2c_mou_7	loc_og_t2c_mou_8	loc_og_t2c_mou_9	loc_og_mou_6
\				
7	0.0	0.00	NaN	422.16
8	0.0	7.15	0.0	378.09
13	0.0	0.49	0.0	205.31
16	0.0	0.00	0.0	315.91
17	0.0	0.00	0.0	97.54

	loc_og_mou_7	loc_og_mou_8	loc_og_mou_9	std_og_t2t_mou_6	\
7	533.91	255.79	NaN	4.30	
8	288.18	63.04	60.71	116.56	
13	53.48	353.99	119.69	446.41	
16	215.64	205.93	233.04	7.89	
17	0.00	0.00	45.79	1146.91	

	std_og_t2t_mou_7	std_og_t2t_mou_8	std_og_t2t_mou_9
std_og_t2m_mou_6	\		
7	23.29	12.01	NaN
49.89			
8	133.43	22.58	7.33
13.69			
13	85.98	498.23	230.38
255.36			
16	2.58	3.23	4.46
22.99			
17	0.81	0.00	504.11
1.55			

	std_og_t2m_mou_7	std_og_t2m_mou_8	std_og_t2m_mou_9
std_og_t2f_mou_6	\		
7	31.76	49.14	NaN
6.66			
8	10.04	75.69	74.13
0.00			
13	52.94	156.94	96.01

0.00			
16	64.51	18.29	13.79
0.00			
17	0.00	0.00	78.51
0.00			

	std_og_t2f_mou_7	std_og_t2f_mou_8	std_og_t2f_mou_9
std_og_t2c_mou_6 \			
7	20.08	16.68	NaN
0.0			
8	0.00	0.00	0.00
0.0			
13	0.00	0.00	0.00
0.0			
16	0.00	0.00	4.43
0.0			
17	0.00	0.00	0.00
0.0			

	std_og_t2c_mou_7	std_og_t2c_mou_8	std_og_t2c_mou_9	std_og_mou_6
\				
7	0.0	0.0	NaN	60.86
8	0.0	0.0	0.0	130.26
13	0.0	0.0	0.0	701.78
16	0.0	0.0	0.0	30.89
17	0.0	0.0	0.0	1148.46

	std_og_mou_7	std_og_mou_8	std_og_mou_9	isd_og_mou_6
isd_og_mou_7 \				
7	75.14	77.84	NaN	0.0
0.18				
8	143.48	98.28	81.46	0.0
0.00				
13	138.93	655.18	326.39	0.0
0.00				
16	67.09	21.53	22.69	0.0
0.00				
17	0.81	0.00	582.63	0.0
0.00				

	isd_og_mou_8	isd_og_mou_9	spl_og_mou_6	spl_og_mou_7
spl_og_mou_8 \				
7	10.01	NaN	4.50	0.00
6.50				
8	0.00	0.0	0.00	0.00

10.23				
13	1.29	0.0	0.00	0.00
4.78				
16	0.00	0.0	0.00	3.26
5.91				
17	0.00	0.0	2.58	0.00
0.00				
spl_og_mou_9 og_others_6 og_others_7 og_others_8				
og_others_9 \				
7	NaN	0.00	0.0	0.0 NaN
8	0.00	0.00	0.0	0.0 0.0
13	0.00	0.00	0.0	0.0 0.0
16	0.00	0.00	0.0	0.0 0.0
17	2.64	0.93	0.0	0.0 0.0
total_og_mou_6 total_og_mou_7 total_og_mou_8 total_og_mou_9 \				
7	487.53	609.24	350.16	0.00
8	508.36	431.66	171.56	142.18
13	907.09	192.41	1015.26	446.09
16	346.81	286.01	233.38	255.74
17	1249.53	0.81	0.00	631.08
loc_ic_t2t_mou_6 loc_ic_t2t_mou_7 loc_ic_t2t_mou_8				
loc_ic_t2t_mou_9 \				
7	58.14	32.26	27.31	
NaN				
8	23.84	9.84	0.31	
4.03				
13	67.88	7.58	52.58	
24.98				
16	41.33	71.44	28.89	
50.23				
17	34.54	0.00	0.00	
40.91				
loc_ic_t2m_mou_6 loc_ic_t2m_mou_7 loc_ic_t2m_mou_8				
loc_ic_t2m_mou_9 \				
7	217.56	221.49	121.19	
NaN				
8	57.58	13.98	15.48	
17.34				
13	142.88	18.53	195.18	
104.79				
16	226.81	149.69	150.16	

172.86			
17	47.41	2.31	0.00
43.86			

	loc_ic_t2f_mou_6	loc_ic_t2f_mou_7	loc_ic_t2f_mou_8
loc_ic_t2f_mou_9 \			
7	152.16	101.46	39.53
NaN			
8	0.00	0.00	0.00
0.00			
13	4.81	0.00	7.49
8.51			
16	8.71	8.68	32.71
65.21			
17	0.00	0.00	0.00
0.71			

	loc_ic_mou_6	loc_ic_mou_7	loc_ic_mou_8	loc_ic_mou_9
std_ic_t2t_mou_6 \				
7	427.88	355.23	188.04	NaN
36.89				
8	81.43	23.83	15.79	21.38
0.00				
13	215.58	26.11	255.26	138.29
115.68				
16	276.86	229.83	211.78	288.31
68.79				
17	81.96	2.31	0.00	85.49
8.63				

	std_ic_t2t_mou_7	std_ic_t2t_mou_8	std_ic_t2t_mou_9
std_ic_t2m_mou_6 \			
7	11.83	30.39	NaN
91.44			
8	0.58	0.10	0.00
22.43			
13	38.29	154.58	62.39
308.13			
16	78.64	6.33	16.66
18.68			
17	0.00	0.00	0.00
1.28			

	std_ic_t2m_mou_7	std_ic_t2m_mou_8	std_ic_t2m_mou_9
std_ic_t2f_mou_6 \			
7	126.99	141.33	NaN
52.19			
8	4.08	0.65	13.53
0.00			
13	29.79	317.91	151.51

0.00				
16	73.08	73.93	29.58	
0.51				
17	0.00	0.00	1.63	
0.00				
	std_ic_t2f_mou_7	std_ic_t2f_mou_8	std_ic_t2f_mou_9	
std_ic_t2o_mou_6 \				
7	34.24	22.21	NaN	
0.0				
8	0.00	0.00	0.0	
0.0				
13	0.00	1.91	0.0	
0.0				
16	0.00	2.18	0.0	
0.0				
17	0.00	0.00	0.0	
0.0				
	std_ic_t2o_mou_7	std_ic_t2o_mou_8	std_ic_t2o_mou_9	std_ic_mou_6
\				
7	0.0	0.0	NaN	180.54
8	0.0	0.0	0.0	22.43
13	0.0	0.0	0.0	423.81
16	0.0	0.0	0.0	87.99
17	0.0	0.0	0.0	9.91
	std_ic_mou_7	std_ic_mou_8	std_ic_mou_9	total_ic_mou_6
total_ic_mou_7 \				
7	173.08	193.94	NaN	626.46
558.04				
8	4.66	0.75	13.53	103.86
28.49				
13	68.09	474.41	213.91	968.61
172.58				
16	151.73	82.44	46.24	364.86
381.56				
17	0.00	0.00	1.63	91.88
2.31				
	total_ic_mou_8	total_ic_mou_9	spl_ic_mou_6	spl_ic_mou_7
spl_ic_mou_8 \				
7	428.74	0.00	0.21	0.0
0.0				
8	16.54	34.91	0.00	0.0

0.0				
13	1144.53	631.86	0.45	0.0
0.0				
16	294.46	334.56	0.00	0.0
0.0				
17	0.00	87.13	0.00	0.0
0.0				

	spl_ic_mou_9	isd_ic_mou_6	isd_ic_mou_7	isd_ic_mou_8
isd_ic_mou_9	\			
7	NaN	2.06	14.53	31.59
NaN				
8	0.0	0.00	0.00	0.00
0.00				
13	0.0	245.28	62.11	393.39
259.33				
16	0.0	0.00	0.00	0.23
0.00				
17	0.0	0.00	0.00	0.00
0.00				

	ic_others_6	ic_others_7	ic_others_8	ic_others_9
total_rech_num_6	\			
7	15.74	15.19	15.14	NaN
5				
8	0.00	0.00	0.00	0.00
19				
13	83.48	16.24	21.44	20.31
6				
16	0.00	0.00	0.00	0.00
10				
17	0.00	0.00	0.00	0.00
19				

	total_rech_num_7	total_rech_num_8	total_rech_num_9
total_rech_amt_6	\		
7	5	7	3
1580			
8	21	14	15
437			
13	4	11	7
507			
16	6	2	1
570			
17	2	4	10
816			

	total_rech_amt_7	total_rech_amt_8	total_rech_amt_9
max_rech_amt_6	\		
7	790	3638	0

1580					
8	601	120	186		
90					
13	253	717	353		
110					
16	348	160	220		
110					
17	0	30	335		
110					
	max_rech_amt_7	max_rech_amt_8	max_rech_amt_9	last_day_rch_amt_6	
\					
7	790	1580	0	0	
8	154	30	36	50	
13	110	130	130	110	
16	110	130	220	100	
17	0	30	130	30	
	last_day_rch_amt_7	last_day_rch_amt_8	last_day_rch_amt_9		
vol_2g_mb_6	\				
7	0	779	0		
0.0					
8	0	10	0		
0.0					
13	50	0	0		
0.0					
16	100	130	220		
0.0					
17	0	0	0		
0.0					
	vol_2g_mb_7	vol_2g_mb_8	vol_2g_mb_9	vol_3g_mb_6	vol_3g_mb_7
7	0.0	0.00	0.0	0.0	0.00
8	356.0	0.03	0.0	0.0	750.95
13	0.0	0.02	0.0	0.0	0.00
16	0.0	0.00	0.0	0.0	0.00
17	0.0	0.00	0.0	0.0	0.00
	vol_3g_mb_8	vol_3g_mb_9	monthly_2g_6	monthly_2g_7	monthly_2g_8
\					
7	0.00	0.0	0	0	0
8	11.94	0.0	0	1	0
13	0.00	0.0	0	0	0

16	0.00	0.0	0	0	0
----	------	-----	---	---	---

17	0.00	0.0	0	0	0
----	------	-----	---	---	---

	monthly_2g_9	sachet_2g_6	sachet_2g_7	sachet_2g_8	
sachet_2g_9 \					
7	0	0	0	0	0

8	0	0	1	3	0
---	---	---	---	---	---

13	0	0	0	3	0
----	---	---	---	---	---

16	0	0	0	0	0
----	---	---	---	---	---

17	0	0	0	0	0
----	---	---	---	---	---

	monthly_3g_6	monthly_3g_7	monthly_3g_8	monthly_3g_9	
sachet_3g_6 \					
7	0	0	0	0	

0					
---	--	--	--	--	--

8	0	0	0	0	
---	---	---	---	---	--

0					
---	--	--	--	--	--

13	0	0	0	0	
----	---	---	---	---	--

0					
---	--	--	--	--	--

16	0	0	0	0	
----	---	---	---	---	--

0					
---	--	--	--	--	--

17	0	0	0	0	
----	---	---	---	---	--

0					
---	--	--	--	--	--

	sachet_3g_7	sachet_3g_8	sachet_3g_9	aon	aug_vbc_3g
jul_vbc_3g \					
7	0	0	0	802	57.74

19.38					
-------	--	--	--	--	--

8	0	0	0	315	21.03
---	---	---	---	-----	-------

910.65					
--------	--	--	--	--	--

13	0	0	0	2607	0.00
----	---	---	---	------	------

0.00					
------	--	--	--	--	--

16	0	0	0	511	0.00
----	---	---	---	-----	------

2.45					
------	--	--	--	--	--

17	0	0	0	667	0.00
----	---	---	---	-----	------

0.00					
------	--	--	--	--	--

	jun_vbc_3g	sep_vbc_3g	avg_rech_amt_6_7	
7	18.74	0.0	1185.0	
8	122.16	0.0	519.0	
13	0.00	0.0	380.0	

16	21.89	0.0	459.0
17	0.00	0.0	408.0

```
df.shape
```

```
(30011, 178)
```

We can see that we have around **~30K** rows after filtering

Handling missing values in rows

```
# Count the rows having more than 50% missing values
```

```
df_missing_rows_50 = df[(df.isnull().sum(axis=1)) >
(len(df.columns)//2)]
```

```
df_missing_rows_50.shape
```

```
(114, 178)
```

```
# Deleting the rows having more than 50% missing values
```

```
df = df.drop(df_missing_rows_50.index)
```

```
df.shape
```

```
(29897, 178)
```

```
# Checking the missing values in columns again
```

```
df_missing_columns =
(round(((df.isnull().sum())/len(df.index))*100),2).to_frame('null').so
rt_values('null', ascending=False)
df_missing_columns
```

	null
loc_ic_mou_9	5.32
og_others_9	5.32
loc_og_t2t_mou_9	5.32
loc_ic_t2t_mou_9	5.32
loc_og_t2m_mou_9	5.32
loc_og_t2f_mou_9	5.32
loc_og_t2c_mou_9	5.32
std_ic_t2m_mou_9	5.32
loc_og_mou_9	5.32
std_og_t2t_mou_9	5.32
roam_og_mou_9	5.32
std_ic_t2o_mou_9	5.32
std_og_t2m_mou_9	5.32
std_og_t2f_mou_9	5.32
spl_og_mou_9	5.32
std_og_t2c_mou_9	5.32
std_og_mou_9	5.32
isd_og_mou_9	5.32
std_ic_t2t_mou_9	5.32
std_ic_mou_9	5.32

onnet_mou_9	5.32
spl_ic_mou_9	5.32
ic_others_9	5.32
isd_ic_mou_9	5.32
loc_ic_t2f_mou_9	5.32
offnet_mou_9	5.32
loc_ic_t2m_mou_9	5.32
std_ic_t2f_mou_9	5.32
roam_ic_mou_9	5.32
loc_og_t2t_mou_8	2.76
...	...
total_ic_mou_8	0.00
std_og_t2o_mou	0.00
loc_ic_t2o_mou	0.00
arpu_6	0.00
arpu_7	0.00
arpu_8	0.00
arpu_9	0.00
total_og_mou_6	0.00
total_og_mou_7	0.00
total_og_mou_8	0.00
total_og_mou_9	0.00
loc_og_t2o_mou	0.00
total_ic_mou_6	0.00
total_ic_mou_7	0.00
total_ic_mou_9	0.00
last_day_rch_amt_7	0.00
total_rech_num_6	0.00
total_rech_num_7	0.00
total_rech_num_8	0.00
total_rech_num_9	0.00
total_rech_amt_6	0.00
total_rech_amt_7	0.00
total_rech_amt_8	0.00
total_rech_amt_9	0.00
max_rech_amt_6	0.00
max_rech_amt_7	0.00
max_rech_amt_8	0.00
max_rech_amt_9	0.00
last_day_rch_amt_6	0.00
avg_rech_amt_6_7	0.00

[178 rows x 1 columns]

Looks like MOU for all the types of calls for the month of September (9) have missing values together for any particular record.

Lets check the records for the MOU for Sep(9), in which these coulms have missing values together.

```

# Listing the columns of MOU Sep(9)
print((df_missing_columns[df_missing_columns['null'] ==
5.32]).index().to_list())

['loc_ic_mou_9', 'og_others_9', 'loc_og_t2t_mou_9',
'loc_ic_t2t_mou_9', 'loc_og_t2m_mou_9', 'loc_og_t2f_mou_9',
'loc_og_t2c_mou_9', 'std_ic_t2m_mou_9', 'loc_og_mou_9',
'std_og_t2t_mou_9', 'roam_og_mou_9', 'std_ic_t2o_mou_9',
'std_og_t2m_mou_9', 'std_og_t2f_mou_9', 'spl_og_mou_9',
'std_og_t2c_mou_9', 'std_og_mou_9', 'isd_og_mou_9',
'std_ic_t2t_mou_9', 'std_ic_mou_9', 'onnet_mou_9', 'spl_ic_mou_9',
'ic_others_9', 'isd_ic_mou_9', 'loc_ic_t2f_mou_9', 'offnet_mou_9',
'loc_ic_t2m_mou_9', 'std_ic_t2f_mou_9', 'roam_ic_mou_9']

# Creating a dataframe with the condition, in which MOU for Sep(9) are
null
df_null_mou_9 = df[(df['loc_og_t2m_mou_9'].isnull()) &
(df['loc_ic_t2f_mou_9'].isnull()) & (df['roam_og_mou_9'].isnull()) &
(df['std_ic_t2m_mou_9'].isnull()) &
(df['loc_og_t2t_mou_9'].isnull()) &
(df['std_ic_t2t_mou_9'].isnull()) & (df['loc_og_t2f_mou_9'].isnull())
& (df['loc_ic_mou_9'].isnull()) &
(df['loc_og_t2c_mou_9'].isnull()) & (df['loc_og_mou_9'].isnull()) &
(df['std_og_t2t_mou_9'].isnull()) & (df['roam_ic_mou_9'].isnull()) &
(df['loc_ic_t2m_mou_9'].isnull()) &
(df['std_og_t2m_mou_9'].isnull()) & (df['loc_ic_t2t_mou_9'].isnull())
& (df['std_og_t2f_mou_9'].isnull()) &
(df['std_og_t2c_mou_9'].isnull()) & (df['og_others_9'].isnull()) &
(df['std_og_mou_9'].isnull()) & (df['spl_og_mou_9'].isnull()) &
(df['std_ic_t2f_mou_9'].isnull()) & (df['isd_og_mou_9'].isnull()) &
(df['std_ic_mou_9'].isnull()) & (df['offnet_mou_9'].isnull()) &
(df['isd_ic_mou_9'].isnull()) & (df['ic_others_9'].isnull()) &
(df['std_ic_t2o_mou_9'].isnull()) & (df['onnet_mou_9'].isnull()) &
(df['spl_ic_mou_9'].isnull())]

df_null_mou_9.head()

```

	mobile_number	loc_og_t2o_mou	std_og_t2o_mou	loc_ic_t2o_mou
arpu_6 \				
7	7000701601	0.0	0.0	0.0
1069.180				
97	7000589828	0.0	0.0	0.0
374.863				
111	7001300706	0.0	0.0	0.0
596.301				
143	7000106299	0.0	0.0	0.0
695.609				
188	7000340381	0.0	0.0	0.0
734.641				

	arpu_7	arpu_8	arpu_9	onnet_mou_6	onnet_mou_7	onnet_mou_8
\						
7	1349.850	3171.480	500.0	57.84	54.68	52.29
97	294.023	183.043	0.0	433.59	415.66	221.06
111	146.073	0.000	0.0	55.19	3.26	NaN
143	39.981	0.000	0.0	1325.91	28.61	NaN
188	183.668	0.000	0.0	4.38	0.98	NaN
	onnet_mou_9	offnet_mou_6	offnet_mou_7	offnet_mou_8		
offnet_mou_9 \						
7	NaN	453.43	567.16	325.91		
NaN						
97	NaN	74.54	43.66	31.86		
NaN						
111	NaN	45.51	12.34	NaN		
NaN						
143	NaN	13.91	1.89	NaN		
NaN						
188	NaN	105.16	39.39	NaN		
NaN						
	roam_ic_mou_6	roam_ic_mou_7	roam_ic_mou_8	roam_ic_mou_9	\	
7	16.23	33.49	31.64	NaN		
97	0.00	0.00	6.16	NaN		
111	0.00	0.00	NaN	NaN		
143	0.00	8.94	NaN	NaN		
188	0.00	0.00	NaN	NaN		
	roam_og_mou_6	roam_og_mou_7	roam_og_mou_8	roam_og_mou_9	\	
7	23.74	12.59	38.06	NaN		
97	0.00	0.00	23.91	NaN		
111	0.00	0.00	NaN	NaN		
143	0.00	8.53	NaN	NaN		
188	0.00	0.00	NaN	NaN		
	loc_og_t2t_mou_6	loc_og_t2t_mou_7	loc_og_t2t_mou_8			
loc_og_t2t_mou_9 \						
7	51.39	31.38	40.28			
NaN						
97	2.83	16.19	9.73			
NaN						
111	55.19	3.26	NaN			
NaN						
143	18.89	6.83	NaN			
NaN						

188	4.38	0.98	NaN	
NaN				
	loc_og_t2m_mou_6	loc_og_t2m_mou_7	loc_og_t2m_mou_8	
loc_og_t2m_mou_9 \				
7	308.63	447.38	162.28	
NaN				
97	16.99	23.14	17.79	
NaN				
111	43.83	12.34	NaN	
NaN				
143	8.58	1.56	NaN	
NaN				
188	99.81	38.98	NaN	
NaN				
	loc_og_t2f_mou_6	loc_og_t2f_mou_7	loc_og_t2f_mou_8	
loc_og_t2f_mou_9 \				
7	62.13	55.14	53.23	
NaN				
97	3.54	1.46	1.83	
NaN				
111	0.00	0.00	NaN	
NaN				
143	0.00	0.00	NaN	
NaN				
188	5.34	0.41	NaN	
NaN				
	loc_og_t2c_mou_6	loc_og_t2c_mou_7	loc_og_t2c_mou_8	
loc_og_t2c_mou_9 \				
7	0.00	0.0	0.0	
NaN				
97	0.40	0.0	0.0	
NaN				
111	0.00	0.0	NaN	
NaN				
143	2.09	0.0	NaN	
NaN				
188	0.00	0.0	NaN	
NaN				
	loc_og_mou_6	loc_og_mou_7	loc_og_mou_8	loc_og_mou_9
std_og_t2t_mou_6 \				
7	422.16	533.91	255.79	NaN
4.30				
97	23.38	40.81	29.36	NaN
430.76				
111	99.03	15.61	NaN	NaN
0.00				

143	27.48	8.39	NaN	NaN
1307.01				
188	109.54	40.38	NaN	NaN
0.00				

	std_og_t2t_mou_7	std_og_t2t_mou_8	std_og_t2t_mou_9
std_og_t2m_mou_6 \			
7	23.29	12.01	NaN
49.89			
97	399.46	191.31	NaN
53.59			
111	0.00	NaN	NaN
0.00			
143	13.58	NaN	NaN
1.95			
188	0.00	NaN	NaN
0.00			

	std_og_t2m_mou_7	std_og_t2m_mou_8	std_og_t2m_mou_9
std_og_t2f_mou_6 \			
7	31.76	49.14	NaN
6.66			
97	13.81	8.33	NaN
0.00			
111	0.00	NaN	NaN
1.30			
143	0.00	NaN	NaN
0.00			
188	0.00	NaN	NaN
0.00			

	std_og_t2f_mou_7	std_og_t2f_mou_8	std_og_t2f_mou_9
std_og_t2c_mou_6 \			
7	20.08	16.68	NaN
0.0			
97	0.00	0.00	NaN
0.0			
111	0.00	NaN	NaN
0.0			
143	0.00	NaN	NaN
0.0			
188	0.00	NaN	NaN
0.0			

	std_og_t2c_mou_7	std_og_t2c_mou_8	std_og_t2c_mou_9
std_og_mou_6 \			
7	0.0	0.0	NaN
60.86			
97	0.0	0.0	NaN
484.36			

111	0.0	NaN	NaN	
1.30				
143	0.0	NaN	NaN	
1308.96				
188	0.0	NaN	NaN	
0.00				
std_og_mou_7	std_og_mou_8	std_og_mou_9	isd_og_mou_6	
isd_og_mou_7 \				
7	75.14	77.84	NaN	
0.18			0.0	
97	413.28	199.64	NaN	
0.00			0.0	
111	0.00	NaN	NaN	
0.00			0.0	
143	13.58	NaN	NaN	
0.00			0.0	
188	0.00	NaN	NaN	
0.00			0.0	
isd_og_mou_8	isd_og_mou_9	spl_og_mou_6	spl_og_mou_7	
spl_og_mou_8 \				
7	10.01	NaN	4.50	
6.50			0.00	
97	0.00	NaN	2.54	
2.01			11.81	
111	NaN	NaN	0.38	
NaN			2.71	
143	NaN	NaN	3.38	
NaN			0.00	
188	NaN	NaN	0.00	
NaN			0.00	
spl_og_mou_9	og_others_6	og_others_7	og_others_8	og_others_9
\				
7	NaN	0.00	0.0	0.0
				NaN
97	NaN	0.86	0.0	0.0
				NaN
111	NaN	1.29	0.0	NaN
				NaN
143	NaN	1.20	0.0	NaN
				NaN
188	NaN	0.00	0.0	NaN
				NaN
total_og_mou_6	total_og_mou_7	total_og_mou_8	total_og_mou_9	\
7	487.53	609.24	350.16	0.0
97	511.16	465.91	231.03	0.0
111	102.01	18.33	0.00	0.0

143	1341.03	21.98	0.00	0.0
188	109.54	40.38	0.00	0.0
	loc_ic_t2t_mou_6	loc_ic_t2t_mou_7	loc_ic_t2t_mou_8	
loc_ic_t2t_mou_9 \				
7	58.14	32.26	27.31	
NaN				
97	11.61	32.89	4.46	
NaN				
111	50.01	16.66	NaN	
NaN				
143	30.19	7.06	NaN	
NaN				
188	21.18	13.44	NaN	
NaN				
	loc_ic_t2m_mou_6	loc_ic_t2m_mou_7	loc_ic_t2m_mou_8	
loc_ic_t2m_mou_9 \				
7	217.56	221.49	121.19	
NaN				
97	16.94	26.94	26.63	
NaN				
111	160.68	58.53	NaN	
NaN				
143	27.98	1.35	NaN	
NaN				
188	217.03	56.63	NaN	
NaN				
	loc_ic_t2f_mou_6	loc_ic_t2f_mou_7	loc_ic_t2f_mou_8	
loc_ic_t2f_mou_9 \				
7	152.16	101.46	39.53	
NaN				
97	0.98	0.63	0.00	
NaN				
111	5.06	0.40	NaN	
NaN				
143	10.13	0.00	NaN	
NaN				
188	18.28	2.94	NaN	
NaN				
	loc_ic_mou_6	loc_ic_mou_7	loc_ic_mou_8	loc_ic_mou_9
std_ic_t2t_mou_6 \				
7	427.88	355.23	188.04	NaN
36.89				
97	29.54	60.48	31.09	NaN
0.49				
111	215.76	75.59	NaN	NaN
0.00				

143	68.31	8.41	NaN	NaN
25.56				
188	256.49	73.03	NaN	NaN
0.00				

	std_ic_t2t_mou_7	std_ic_t2t_mou_8	std_ic_t2t_mou_9
std_ic_t2m_mou_6 \			
7	11.83	30.39	NaN
91.44			
97	1.36	1.06	NaN
0.00			
111	0.00	NaN	NaN
0.00			
143	0.00	NaN	NaN
0.00			
188	0.00	NaN	NaN
0.00			

	std_ic_t2m_mou_7	std_ic_t2m_mou_8	std_ic_t2m_mou_9
std_ic_t2f_mou_6 \			
7	126.99	141.33	NaN
52.19			
97	4.16	0.00	NaN
0.00			
111	0.00	NaN	NaN
1.13			
143	0.00	NaN	NaN
0.00			
188	0.00	NaN	NaN
0.00			

	std_ic_t2f_mou_7	std_ic_t2f_mou_8	std_ic_t2f_mou_9
std_ic_t2o_mou_6 \			
7	34.24	22.21	NaN
0.0			
97	0.00	0.00	NaN
0.0			
111	0.00	NaN	NaN
0.0			
143	0.00	NaN	NaN
0.0			
188	0.00	NaN	NaN
0.0			

	std_ic_t2o_mou_7	std_ic_t2o_mou_8	std_ic_t2o_mou_9
std_ic_mou_6 \			
7	0.0	0.0	NaN
180.54			
97	0.0	0.0	NaN
0.49			

111	0.0	NaN	NaN
1.13			
143	0.0	NaN	NaN
25.56			
188	0.0	NaN	NaN
0.00			

	std_ic_mou_7	std_ic_mou_8	std_ic_mou_9	total_ic_mou_6
total_ic_mou_7 \				
7	173.08	193.94	NaN	626.46
558.04				
97	5.53	1.06	NaN	32.04
67.84				
111	0.00	NaN	NaN	217.04
75.59				
143	0.00	NaN	NaN	93.88
8.41				
188	0.00	NaN	NaN	256.49
73.03				

	total_ic_mou_8	total_ic_mou_9	spl_ic_mou_6	spl_ic_mou_7
spl_ic_mou_8 \				
7	428.74	0.0	0.21	0.0
0.0				
97	32.16	0.0	0.63	0.0
0.0				
111	0.00	0.0	0.00	0.0
NaN				
143	0.00	0.0	0.00	0.0
NaN				
188	0.00	0.0	0.00	0.0
NaN				

	spl_ic_mou_9	isd_ic_mou_6	isd_ic_mou_7	isd_ic_mou_8
isd_ic_mou_9 \				
7	NaN	2.06	14.53	31.59
NaN				
97	NaN	0.00	0.00	0.00
NaN				
111	NaN	0.00	0.00	NaN
NaN				
143	NaN	0.00	0.00	NaN
NaN				
188	NaN	0.00	0.00	NaN
NaN				

	ic_others_6	ic_others_7	ic_others_8	ic_others_9
total_rech_num_6 \				
7	15.74	15.19	15.14	NaN
5				

97	1.36	1.83	0.00	NaN
14				
111	0.15	0.00	NaN	NaN
12				
143	0.00	0.00	NaN	NaN
31				
188	0.00	0.00	NaN	NaN
6				

	total_rech_num_7	total_rech_num_8	total_rech_num_9
total_rech_amt_6 \			
7	5	7	3
1580			
97	17	14	3
432			
111	8	5	2
704			
143	6	4	2
796			
188	1	0	0
864			

	total_rech_amt_7	total_rech_amt_8	total_rech_amt_9
max_rech_amt_6 \			
7	790	3638	0
1580			
97	328	206	0
36			
111	178	0	0
154			
143	40	0	0
90			
188	120	0	0
252			

	max_rech_amt_7	max_rech_amt_8	max_rech_amt_9
last_day_rch_amt_6 \			
7	790	1580	0
0			
97	44	36	0
30			
111	50	0	0
154			
143	30	0	0
10			
188	120	0	0
252			

	last_day_rch_amt_7	last_day_rch_amt_8	last_day_rch_amt_9
vol_2g_mb_6 \			

7	0	779	0		
0.00					
97	20	0	0		
0.00					
111	30	0	0		
284.50					
143	0	0	0		
0.00					
188	120	0	0		
58.44					
vol_2g_mb_7	vol_2g_mb_8	vol_2g_mb_9	vol_3g_mb_6		
vol_3g_mb_7 \					
7	0.0	0.0	0.0	0.0	0.0
97	0.0	0.0	0.0	0.0	0.0
111	0.0	0.0	0.0	0.0	0.0
143	0.0	0.0	0.0	0.0	0.0
188	0.0	0.0	0.0	1522.4	0.0
vol_3g_mb_8	vol_3g_mb_9	monthly_2g_6	monthly_2g_7		
monthly_2g_8 \					
7	0.0	0.0	0	0	
0					
97	0.0	0.0	0	0	
0					
111	0.0	0.0	1	0	
0					
143	0.0	0.0	0	0	
0					
188	0.0	0.0	0	0	
0					
monthly_2g_9	sachet_2g_6	sachet_2g_7	sachet_2g_8	sachet_2g_9	
\					
7	0	0	0	0	0
97	0	0	0	0	0
111	0	0	0	0	0
143	0	0	0	0	0
188	0	0	0	0	0
monthly_3g_6	monthly_3g_7	monthly_3g_8	monthly_3g_9		

sachet_3g_6 \				
7	0	0	0	0
0				
97	0	0	0	0
0				
111	0	0	0	0
1				
143	0	0	0	0
0				
188	2	0	0	0
0				

	sachet_3g_7	sachet_3g_8	sachet_3g_9	aon	aug_vbc_3g
jul_vbc_3g \					
7	0	0	0	802	57.74
19.38					
97	0	0	0	502	0.00
0.00					
111	0	0	0	332	0.00
0.00					
143	0	0	0	264	0.00
0.00					
188	0	0	0	244	0.00
831.48					

	jun_vbc_3g	sep_vbc_3g	avg_rech_amt_6_7
7	18.74	0.0	1185.0
97	0.00	0.0	380.0
111	0.00	0.0	441.0
143	0.00	0.0	418.0
188	1223.04	0.0	492.0

```
df_null_mou_9.shape
```

```
(1590, 178)
```

```
# Deleting the records for which MOU for Sep(9) are null
```

```
df = df.drop(df_null_mou_9.index)
```

```
# Again Cheking percent of missing values in columns
```

```
df_missing_columns =
```

```
(round(((df.isnull().sum())/len(df.index))*100),2).to_frame('null')).so  
rt_values('null', ascending=False)
```

```
df_missing_columns
```

	null
isd_og_mou_8	0.55
roam_ic_mou_8	0.55
loc_og_mou_8	0.55
std_ic_t2o_mou_8	0.55
roam_og_mou_8	0.55

loc_ic_t2f_mou_8	0.55
loc_og_t2t_mou_8	0.55
std_ic_t2f_mou_8	0.55
std_og_t2m_mou_8	0.55
loc_og_t2m_mou_8	0.55
std_og_t2t_mou_8	0.55
std_ic_t2m_mou_8	0.55
loc_og_t2f_mou_8	0.55
spl_og_mou_8	0.55
loc_ic_mou_8	0.55
loc_og_t2c_mou_8	0.55
std_ic_t2t_mou_8	0.55
loc_ic_t2m_mou_8	0.55
std_og_t2f_mou_8	0.55
spl_ic_mou_8	0.55
std_ic_mou_8	0.55
offnet_mou_8	0.55
ic_others_8	0.55
og_others_8	0.55
loc_ic_t2t_mou_8	0.55
onnet_mou_8	0.55
isd_ic_mou_8	0.55
std_og_t2c_mou_8	0.55
std_og_mou_8	0.55
isd_og_mou_6	0.50
...	...
arpu_9	0.00
arpu_8	0.00
arpu_7	0.00
arpu_6	0.00
loc_ic_t2o_mou	0.00
std_og_t2o_mou	0.00
std_og_mou_9	0.00
spl_og_mou_9	0.00
isd_ic_mou_9	0.00
og_others_9	0.00
spl_ic_mou_9	0.00
total_ic_mou_9	0.00
total_ic_mou_8	0.00
total_ic_mou_7	0.00
total_ic_mou_6	0.00
std_ic_mou_9	0.00
std_ic_t2o_mou_9	0.00
std_ic_t2f_mou_9	0.00
std_ic_t2m_mou_9	0.00
std_ic_t2t_mou_9	0.00
loc_ic_mou_9	0.00
loc_ic_t2f_mou_9	0.00
loc_og_t2o_mou	0.00

```
loc_ic_t2m_mou_9    0.00
loc_ic_t2t_mou_9    0.00
total_og_mou_9      0.00
total_og_mou_8      0.00
total_og_mou_7      0.00
total_og_mou_6      0.00
avg_rech_amt_6_7    0.00
```

```
[178 rows x 1 columns]
```

Looks like MOU for all the types of calls for the month of Aug (8) have missing values together for any particular record.

Lets check the records for the MOU for Aug(8), in which these coulmnns have missing values together.

```
# Listing the columns of MOU Aug(8)
print((df_missing_columns[df_missing_columns['null'] ==
0.55]).index().to_list())

['isd_og_mou_8', 'roam_ic_mou_8', 'loc_og_mou_8', 'std_ic_t2o_mou_8',
'roam_og_mou_8', 'loc_ic_t2f_mou_8', 'loc_og_t2t_mou_8',
'std_ic_t2f_mou_8', 'std_og_t2m_mou_8', 'loc_og_t2m_mou_8',
'std_og_t2t_mou_8', 'std_ic_t2m_mou_8', 'loc_og_t2f_mou_8',
'spl_og_mou_8', 'loc_ic_mou_8', 'loc_og_t2c_mou_8',
'std_ic_t2t_mou_8', 'loc_ic_t2m_mou_8', 'std_og_t2f_mou_8',
'spl_ic_mou_8', 'std_ic_mou_8', 'offnet_mou_8', 'ic_others_8',
'og_others_8', 'loc_ic_t2t_mou_8', 'onnet_mou_8', 'isd_ic_mou_8',
'std_og_t2c_mou_8', 'std_og_mou_8']

# Creating a dataframe with the condition, in which MOU for Aug(8) are
null
df_null_mou_8 = df[(df['loc_og_t2m_mou_8'].isnull()) &
(df['loc_ic_t2f_mou_8'].isnull()) & (df['roam_og_mou_8'].isnull()) &
(df['std_ic_t2m_mou_8'].isnull()) &
(df['loc_og_t2t_mou_8'].isnull()) &
(df['std_ic_t2t_mou_8'].isnull()) & (df['loc_og_t2f_mou_8'].isnull())
& (df['loc_ic_mou_8'].isnull()) &
(df['loc_og_t2c_mou_8'].isnull()) & (df['loc_og_mou_8'].isnull()) &
(df['std_og_t2t_mou_8'].isnull()) & (df['roam_ic_mou_8'].isnull()) &
(df['loc_ic_t2m_mou_8'].isnull()) &
(df['std_og_t2m_mou_8'].isnull()) & (df['loc_ic_t2t_mou_8'].isnull())
& (df['std_og_t2f_mou_8'].isnull()) &
(df['std_og_t2c_mou_8'].isnull()) & (df['og_others_8'].isnull()) &
(df['std_og_mou_8'].isnull()) & (df['spl_og_mou_8'].isnull()) &
(df['std_ic_t2f_mou_8'].isnull()) & (df['isd_og_mou_8'].isnull()) &
(df['std_ic_mou_8'].isnull()) & (df['offnet_mou_8'].isnull()) &
(df['isd_ic_mou_8'].isnull()) & (df['ic_others_8'].isnull()) &
(df['std_ic_t2o_mou_8'].isnull()) & (df['onnet_mou_8'].isnull()) &
(df['spl_ic_mou_8'].isnull())]
```

```
df_null_mou_8.head()
```

	mobile_number	loc_og_t2o_mou	std_og_t2o_mou	loc_ic_t2o_mou
arpu_6 \				
375	7002252754	0.0	0.0	0.0
580.477				
578	7000248548	0.0	0.0	0.0
569.612				
788	7000636808	0.0	0.0	0.0
532.742				
1802	7000516213	0.0	0.0	0.0
810.455				
4837	7002192662	0.0	0.0	0.0
649.150				

	arpu_7	arpu_8	arpu_9	onnet_mou_6	onnet_mou_7	onnet_mou_8
\						
375	111.878	0.0	378.881	249.43	39.64	NaN
578	237.289	0.0	4.440	718.01	212.73	NaN
788	546.756	0.0	269.274	1173.39	891.83	NaN
1802	0.000	0.0	0.000	91.33	NaN	NaN
4837	149.572	0.0	0.250	1354.24	85.13	NaN

	onnet_mou_9	offnet_mou_6	offnet_mou_7	offnet_mou_8
offnet_mou_9 \				
375	245.06	62.24	37.24	NaN
144.53				
578	0.00	487.06	139.71	NaN
1.26				
788	149.34	61.59	137.14	NaN
428.36				
1802	0.00	1371.04	NaN	NaN
0.00				
4837	0.43	50.63	37.13	NaN
0.00				

	roam_ic_mou_6	roam_ic_mou_7	roam_ic_mou_8	roam_ic_mou_9	\
375	25.49	19.43	NaN	0.00	
578	0.00	2.01	NaN	6.43	
788	0.00	1.48	NaN	0.00	
1802	1.21	NaN	NaN	0.00	
4837	0.00	12.84	NaN	1.25	

	roam_og_mou_6	roam_og_mou_7	roam_og_mou_8	roam_og_mou_9	\
--	---------------	---------------	---------------	---------------	---

375	312.59	78.58	NaN	0.00
578	0.00	6.30	NaN	1.26
788	0.00	14.43	NaN	0.00
1802	11.23	NaN	NaN	3.91
4837	0.00	44.78	NaN	0.43

	loc_og_t2t_mou_6	loc_og_t2t_mou_7	loc_og_t2t_mou_8
loc_og_t2t_mou_9 \			
375	0.00	0.00	NaN
11.54			
578	11.28	27.89	NaN
0.00			
788	31.06	27.49	NaN
7.39			
1802	17.86	NaN	NaN
0.00			
4837	6.71	1.35	NaN
0.00			

	loc_og_t2m_mou_6	loc_og_t2m_mou_7	loc_og_t2m_mou_8
loc_og_t2m_mou_9 \			
375	0.00	0.00	NaN
25.31			
578	42.24	46.94	NaN
0.00			
788	34.66	60.86	NaN
34.23			
1802	84.51	NaN	NaN
0.00			
4837	15.18	15.76	NaN
0.00			

	loc_og_t2f_mou_6	loc_og_t2f_mou_7	loc_og_t2f_mou_8
loc_og_t2f_mou_9 \			
375	0.0	0.0	NaN
0.0			
578	0.0	0.0	NaN
0.0			
788	0.0	0.0	NaN
0.0			
1802	0.0	NaN	NaN
0.0			
4837	0.0	0.0	NaN
0.0			

	loc_og_t2c_mou_6	loc_og_t2c_mou_7	loc_og_t2c_mou_8
loc_og_t2c_mou_9 \			
375	0.00	0.0	NaN
0.41			
578	2.33	0.0	NaN

0.00				
788	0.00	0.0	NaN	
5.58				
1802	10.29	NaN	NaN	
0.00				
4837	0.00	0.0	NaN	
0.00				
	loc_og_mou_6	loc_og_mou_7	loc_og_mou_8	loc_og_mou_9 \
375	0.00	0.00	NaN	36.86
578	53.53	74.84	NaN	0.00
788	65.73	88.36	NaN	41.63
1802	102.38	NaN	NaN	0.00
4837	21.89	17.11	NaN	0.00
	std_og_t2t_mou_6	std_og_t2t_mou_7	std_og_t2t_mou_8	
std_og_t2t_mou_9 \				
375	0.00	0.00	NaN	
233.51				
578	706.73	178.53	NaN	
0.00				
788	1142.33	854.08	NaN	
141.94				
1802	73.46	NaN	NaN	
0.00				
4837	1347.53	48.48	NaN	
0.00				
	std_og_t2m_mou_6	std_og_t2m_mou_7	std_og_t2m_mou_8	
std_og_t2m_mou_9 \				
375	0.00	0.00	NaN	
118.79				
578	442.48	92.76	NaN	
0.00				
788	26.93	67.24	NaN	
388.54				
1802	1207.86	NaN	NaN	
0.00				
4837	35.44	11.88	NaN	
0.00				
	std_og_t2f_mou_6	std_og_t2f_mou_7	std_og_t2f_mou_8	
std_og_t2f_mou_9 \				
375	0.0	0.0	NaN	
0.0				
578	0.0	0.0	NaN	
0.0				
788	0.0	0.0	NaN	
0.0				
1802	0.0	NaN	NaN	

0.0			
4837	0.0	0.0	NaN
0.0			

	std_og_t2c_mou_6	std_og_t2c_mou_7	std_og_t2c_mou_8
std_og_t2c_mou_9 \			
375	0.0	0.0	NaN
0.0			
578	0.0	0.0	NaN
0.0			
788	0.0	0.0	NaN
0.0			
1802	0.0	NaN	NaN
0.0			
4837	0.0	0.0	NaN
0.0			

	std_og_mou_6	std_og_mou_7	std_og_mou_8	std_og_mou_9
isd_og_mou_6 \				
375	0.00	0.00	NaN	352.31
0.0				
578	1149.21	271.29	NaN	0.00
0.0				
788	1169.26	921.33	NaN	530.49
0.0				
1802	1281.33	NaN	NaN	0.00
0.0				
4837	1382.98	60.36	NaN	0.00
0.0				

	isd_og_mou_7	isd_og_mou_8	isd_og_mou_9	spl_og_mou_6
spl_og_mou_7 \				
375	0.0	NaN	0.0	0.00
0.00				
578	0.0	NaN	0.0	2.58
1.21				
788	0.0	NaN	0.0	0.00
4.85				
1802	NaN	NaN	0.0	91.94
NaN				
4837	0.0	NaN	0.0	0.00
0.00				

	spl_og_mou_8	spl_og_mou_9	og_others_6	og_others_7
og_others_8 \				
375	NaN	4.78	0.00	0.0
NaN				
578	NaN	0.00	1.55	0.0
NaN				
788	NaN	5.58	0.00	0.0

NaN				
1802	NaN	0.00	1.53	NaN
NaN				
4837	NaN	0.00	0.00	0.0
NaN				
	og_others_9	total_og_mou_6	total_og_mou_7	total_og_mou_8 \
375	0.0	0.00	0.00	0.0
578	0.0	1206.88	347.36	0.0
788	0.0	1234.99	1014.54	0.0
1802	0.0	1477.19	0.00	0.0
4837	0.0	1404.88	77.48	0.0
	total_og_mou_9	loc_ic_t2t_mou_6	loc_ic_t2t_mou_7	
	loc_ic_t2t_mou_8 \			
375	393.96	0.00	0.00	
NaN				
578	0.00	48.01	63.39	
NaN				
788	577.71	54.19	52.64	
NaN				
1802	0.00	17.68	NaN	
NaN				
4837	0.00	104.46	3.15	
NaN				
	loc_ic_t2t_mou_9	loc_ic_t2m_mou_6	loc_ic_t2m_mou_7	
	loc_ic_t2m_mou_8 \			
375	6.74	0.00	0.00	
NaN				
578	0.00	83.09	64.31	
NaN				
788	12.51	54.69	187.96	
NaN				
1802	0.00	39.46	NaN	
NaN				
4837	0.00	162.01	17.94	
NaN				
	loc_ic_t2m_mou_9	loc_ic_t2f_mou_6	loc_ic_t2f_mou_7	
	loc_ic_t2f_mou_8 \			
375	38.53	0.00	0.00	
NaN				
578	0.00	0.00	0.00	
NaN				
788	81.83	1.16	2.01	
NaN				
1802	0.00	0.70	NaN	
NaN				
4837	0.00	0.00	0.00	

NaN

	loc_ic_t2f_mou_9	loc_ic_mou_6	loc_ic_mou_7	loc_ic_mou_8	\
375	0.0	0.00	0.00	NaN	
578	0.0	131.11	127.71	NaN	
788	0.0	110.06	242.63	NaN	
1802	0.0	57.84	NaN	NaN	
4837	0.0	266.48	21.09	NaN	

	loc_ic_mou_9	std_ic_t2t_mou_6	std_ic_t2t_mou_7
std_ic_t2t_mou_8 \			
375	45.28	0.00	0.00
NaN			
578	0.00	24.98	46.43
NaN			
788	94.34	14.55	5.48
NaN			
1802	0.00	1.88	NaN
NaN			
4837	0.00	35.11	31.96
NaN			

	std_ic_t2t_mou_9	std_ic_t2m_mou_6	std_ic_t2m_mou_7
std_ic_t2m_mou_8 \			
375	8.31	0.00	0.00
NaN			
578	0.00	1.63	16.69
NaN			
788	25.61	11.49	62.19
NaN			
1802	0.00	11.98	NaN
NaN			
4837	0.00	48.48	0.00
NaN			

	std_ic_t2m_mou_9	std_ic_t2f_mou_6	std_ic_t2f_mou_7
std_ic_t2f_mou_8 \			
375	27.31	0.00	0.0
NaN			
578	0.00	0.00	0.0
NaN			
788	13.93	0.00	0.0
NaN			
1802	0.00	0.00	NaN
NaN			
4837	0.00	0.28	0.0
NaN			

	std_ic_t2f_mou_9	std_ic_t2o_mou_6	std_ic_t2o_mou_7
std_ic_t2o_mou_8 \			

375	0.0	0.0	0.0
NaN			
578	0.0	0.0	0.0
NaN			
788	0.0	0.0	0.0
NaN			
1802	0.0	0.0	NaN
NaN			
4837	0.0	0.0	0.0
NaN			

	std_ic_t2o_mou_9	std_ic_mou_6	std_ic_mou_7	std_ic_mou_8	\
375	0.0	0.00	0.00	NaN	
578	0.0	26.61	63.13	NaN	
788	0.0	26.04	67.68	NaN	
1802	0.0	13.86	NaN	NaN	
4837	0.0	83.88	31.96	NaN	

	std_ic_mou_9	total_ic_mou_6	total_ic_mou_7	total_ic_mou_8	\
375	35.63	0.00	0.00	0.0	
578	0.00	157.73	190.84	0.0	
788	39.54	140.74	310.31	0.0	
1802	0.00	71.71	0.00	0.0	
4837	0.00	350.36	53.06	0.0	

	total_ic_mou_9	spl_ic_mou_6	spl_ic_mou_7	spl_ic_mou_8	
spl_ic_mou_9	\				
375	80.91	0.00	0.0	NaN	
0.00					
578	0.00	0.00	0.0	NaN	
0.00					
788	134.14	0.73	0.0	NaN	
0.25					
1802	0.00	0.00	NaN	NaN	
0.00					
4837	0.00	0.00	0.0	NaN	
0.00					

	isd_ic_mou_6	isd_ic_mou_7	isd_ic_mou_8	isd_ic_mou_9	
ic_others_6	\				
375	0.0	0.0	NaN	0.0	
0.00					
578	0.0	0.0	NaN	0.0	
0.00					
788	0.0	0.0	NaN	0.0	
3.89					
1802	0.0	NaN	NaN	0.0	
0.00					
4837	0.0	0.0	NaN	0.0	
0.00					

	ic_others_7	ic_others_8	ic_others_9	total_rech_num_6	\
375	0.0	NaN	0.0	17	
578	0.0	NaN	0.0	19	
788	0.0	NaN	0.0	10	
1802	NaN	NaN	0.0	21	
4837	0.0	NaN	0.0	11	

	total_rech_num_7	total_rech_num_8	total_rech_num_9
total_rech_amt_6 \			
375	6	3	11
700			
578	10	0	4
717			
788	7	4	5
714			
1802	3	0	0
955			
4837	6	3	4
666			

	total_rech_amt_7	total_rech_amt_8	total_rech_amt_9
max_rech_amt_6 \			
375	130	0	440
80			
578	220	0	0
110			
788	494	0	336
128			
1802	0	0	0
110			
4837	176	0	0
110			

	max_rech_amt_7	max_rech_amt_8	max_rech_amt_9
last_day_rch_amt_6 \			
375	50	0	50
30			
578	50	0	0
27			
788	128	0	130
128			
1802	0	0	0
30			
4837	110	0	0
20			

	last_day_rch_amt_7	last_day_rch_amt_8	last_day_rch_amt_9
vol_2g_mb_6 \			
375	0	0	30

0.0			
578	30	0	0
0.0			
788	0	0	130
0.0			
1802	0	0	0
0.0			
4837	0	0	0
0.0			

	vol_2g_mb_7	vol_2g_mb_8	vol_2g_mb_9	vol_3g_mb_6	vol_3g_mb_7
\					
375	0.0	0.0	0.0	0.0	0.0
578	0.0	0.0	0.0	0.0	0.0
788	0.0	0.0	0.0	0.0	0.0
1802	0.0	0.0	0.0	0.0	0.0
4837	0.0	0.0	0.0	0.0	0.0

	vol_3g_mb_8	vol_3g_mb_9	monthly_2g_6	monthly_2g_7
monthly_2g_8				
\				
375	0.0	0.0	0	0
0				
578	0.0	0.0	0	0
0				
788	0.0	0.0	0	0
0				
1802	0.0	0.0	0	0
0				
4837	0.0	0.0	0	0
0				

	monthly_2g_9	sachet_2g_6	sachet_2g_7	sachet_2g_8	sachet_2g_9
\					
375	0	0	0	0	0
578	0	0	0	0	0
788	0	0	0	0	0
1802	0	0	0	0	0
4837	0	0	0	0	0

	monthly_3g_6	monthly_3g_7	monthly_3g_8	monthly_3g_9
sachet_3g_6				
\				

375	0	0	0	0
0				
578	0	0	0	0
0				
788	0	0	0	0
0				
1802	0	0	0	0
0				
4837	0	0	0	0
0				

	sachet_3g_7	sachet_3g_8	sachet_3g_9	aon	aug_vbc_3g
jul_vbc_3g \					
375	0	0	0	1102	0.0
0.0					
578	0	0	0	274	0.0
0.0					
788	0	0	0	936	0.0
0.0					
1802	0	0	0	755	0.0
0.0					
4837	0	0	0	520	0.0
0.0					

	jun_vbc_3g	sep_vbc_3g	avg_rech_amt_6_7
375	0.0	0.0	415.0
578	0.0	0.0	468.5
788	0.0	0.0	604.0
1802	0.0	0.0	477.5
4837	0.0	0.0	421.0

Deleting the records for which MOU for Aug(8) are null

```
df = df.drop(df_null_mou_8.index)
```

Again cheking percent of missing values in columns

```
df_missing_columns =
(round(((df.isnull()).sum())/len(df.index))*100),2).to_frame('null')).so
rt_values('null', ascending=False)
df_missing_columns
```

	null
roam_ic_mou_6	0.44
spl_og_mou_6	0.44
og_others_6	0.44
loc_ic_t2t_mou_6	0.44
loc_og_t2m_mou_6	0.44
loc_og_t2c_mou_6	0.44
loc_ic_t2m_mou_6	0.44
isd_og_mou_6	0.44
loc_og_t2t_mou_6	0.44

std_og_t2m_mou_6	0.44
loc_ic_t2f_mou_6	0.44
ic_others_6	0.44
roam_og_mou_6	0.44
loc_ic_mou_6	0.44
std_og_mou_6	0.44
loc_og_t2f_mou_6	0.44
isd_ic_mou_6	0.44
std_ic_t2t_mou_6	0.44
std_ic_mou_6	0.44
std_og_t2t_mou_6	0.44
std_ic_t2o_mou_6	0.44
std_og_t2f_mou_6	0.44
std_ic_t2f_mou_6	0.44
spl_ic_mou_6	0.44
onnet_mou_6	0.44
std_og_t2c_mou_6	0.44
std_ic_t2m_mou_6	0.44
offnet_mou_6	0.44
loc_og_mou_6	0.44
std_og_t2f_mou_7	0.16
...	...
loc_ic_t2m_mou_8	0.00
std_ic_t2o_mou_8	0.00
std_ic_t2f_mou_9	0.00
std_ic_t2f_mou_8	0.00
std_ic_t2m_mou_9	0.00
std_ic_t2m_mou_8	0.00
std_ic_t2t_mou_9	0.00
std_ic_t2t_mou_8	0.00
loc_ic_mou_9	0.00
loc_ic_mou_8	0.00
loc_ic_t2f_mou_9	0.00
loc_ic_t2f_mou_8	0.00
loc_og_t2o_mou	0.00
loc_ic_t2m_mou_9	0.00
loc_ic_t2t_mou_9	0.00
std_og_t2c_mou_9	0.00
loc_ic_t2t_mou_8	0.00
total_og_mou_9	0.00
total_og_mou_8	0.00
total_og_mou_7	0.00
total_og_mou_6	0.00
og_others_9	0.00
og_others_8	0.00
spl_og_mou_9	0.00
spl_og_mou_8	0.00
isd_og_mou_9	0.00
isd_og_mou_8	0.00

```
std_og_mou_9      0.00
std_og_mou_8      0.00
avg_rech_amt_6_7  0.00
```

```
[178 rows x 1 columns]
```

Looks like MOU for all the types of calls for the month of Jun (6) have missing values together for any particular record.

Lets check the records for the MOU for Jun(6), in which these coulmnns have missing values together.

```
# Listing the columns of MOU Jun(6)
print((df_missing_columns[df_missing_columns['null'] ==
0.44]).index().to_list())

['roam_ic_mou_6', 'spl_og_mou_6', 'og_others_6', 'loc_ic_t2t_mou_6',
'loc_og_t2m_mou_6', 'loc_og_t2c_mou_6', 'loc_ic_t2m_mou_6',
'isd_og_mou_6', 'loc_og_t2t_mou_6', 'std_og_t2m_mou_6',
'loc_ic_t2f_mou_6', 'ic_others_6', 'roam_og_mou_6', 'loc_ic_mou_6',
'std_og_mou_6', 'loc_og_t2f_mou_6', 'isd_ic_mou_6',
'std_ic_t2t_mou_6', 'std_ic_mou_6', 'std_og_t2t_mou_6',
'std_ic_t2o_mou_6', 'std_og_t2f_mou_6', 'std_ic_t2f_mou_6',
'spl_ic_mou_6', 'onnet_mou_6', 'std_og_t2c_mou_6', 'std_ic_t2m_mou_6',
'offnet_mou_6', 'loc_og_mou_6']

# Creating a dataframe with the condition, in which MOU for Jun(6) are
null
df_null_mou_6 = df[(df['loc_og_t2m_mou_6'].isnull()) &
(df['loc_ic_t2f_mou_6'].isnull()) & (df['roam_og_mou_6'].isnull()) &
(df['std_ic_t2m_mou_6'].isnull()) &
(df['loc_og_t2t_mou_6'].isnull()) &
(df['std_ic_t2t_mou_6'].isnull()) & (df['loc_og_t2f_mou_6'].isnull())
& (df['loc_ic_mou_6'].isnull()) &
(df['loc_og_t2c_mou_6'].isnull()) & (df['loc_og_mou_6'].isnull()) &
(df['std_og_t2t_mou_6'].isnull()) & (df['roam_ic_mou_6'].isnull()) &
(df['loc_ic_t2m_mou_6'].isnull()) &
(df['std_og_t2m_mou_6'].isnull()) & (df['loc_ic_t2t_mou_6'].isnull())
& (df['std_og_t2f_mou_6'].isnull()) &
(df['std_og_t2c_mou_6'].isnull()) & (df['og_others_6'].isnull()) &
(df['std_og_mou_6'].isnull()) & (df['spl_og_mou_6'].isnull()) &
(df['std_ic_t2f_mou_6'].isnull()) & (df['isd_og_mou_6'].isnull()) &
(df['std_ic_mou_6'].isnull()) & (df['offnet_mou_6'].isnull()) &
(df['isd_ic_mou_6'].isnull()) & (df['ic_others_6'].isnull()) &
(df['std_ic_t2o_mou_6'].isnull()) & (df['onnet_mou_6'].isnull()) &
(df['spl_ic_mou_6'].isnull())]

df_null_mou_6.head()
```

mobile_number	loc_og_t2o_mou	std_og_t2o_mou	loc_ic_t2o_mou		
arpu_6 \					
77 7001328263	0.0	0.0	0.0		
30.000					
364 7002168045	0.0	0.0	0.0		
0.000					
423 7000635248	0.0	0.0	0.0		
213.802					
934 7002152278	0.0	0.0	0.0		
48.000					
1187 7000486275	0.0	0.0	0.0		
0.000					
arpu_7	arpu_8	arpu_9	onnet_mou_6	onnet_mou_7	onnet_mou_8
\					
77 82.378	674.950	158.710	NaN	34.23	149.69
364 792.112	989.368	923.040	NaN	433.49	198.96
423 304.194	149.710	329.643	NaN	0.00	0.00
934 764.152	500.030	194.400	NaN	14.24	17.48
1187 757.170	995.719	0.000	NaN	1366.71	2268.91
onnet_mou_9	offnet_mou_6	offnet_mou_7	offnet_mou_8		
offnet_mou_9 \					
77 6.31	NaN	39.44	179.18		
57.68					
364 571.99	NaN	845.11	923.58		
828.29					
423 0.00	NaN	10.03	1.45		
0.34					
934 7.69	NaN	16.99	76.86		
43.64					
1187 0.00	NaN	7.78	36.13		
0.00					
roam_ic_mou_6	roam_ic_mou_7	roam_ic_mou_8	roam_ic_mou_9	\	
77 NaN	0.0	0.00	0.0		
364 NaN	0.0	0.00	0.0		
423 NaN	0.0	0.00	0.0		
934 NaN	0.0	8.81	0.0		
1187 NaN	0.0	8.08	0.0		
roam_og_mou_6	roam_og_mou_7	roam_og_mou_8	roam_og_mou_9	\	
77 NaN	0.0	0.00	0.00		
364 NaN	0.0	0.00	0.00		
423 NaN	0.0	0.00	0.00		
934 NaN	0.0	1.56	0.00		

1187	NaN	0.0	25.23	0.21
loc_og_t2t_mou_6	loc_og_t2t_mou_7	loc_og_t2t_mou_8		
loc_og_t2t_mou_9 \				
77	NaN	34.23	149.69	
6.31				
364	NaN	28.78	7.46	
64.73				
423	NaN	0.00	0.00	
0.00				
934	NaN	0.08	17.48	
7.69				
1187	NaN	4.76	46.18	
0.00				
loc_og_t2m_mou_6	loc_og_t2m_mou_7	loc_og_t2m_mou_8		
loc_og_t2m_mou_9 \				
77	NaN	32.18	101.63	
29.41				
364	NaN	78.78	584.76	
490.71				
423	NaN	0.00	0.58	
0.33				
934	NaN	16.99	63.23	
39.99				
1187	NaN	7.78	31.29	
0.00				
loc_og_t2f_mou_6	loc_og_t2f_mou_7	loc_og_t2f_mou_8		
loc_og_t2f_mou_9 \				
77	NaN	0.91	29.86	
28.26				
364	NaN	21.58	9.43	
0.00				
423	NaN	0.00	0.00	
0.00				
934	NaN	0.00	12.08	
3.65				
1187	NaN	0.00	0.00	
0.00				
loc_og_t2c_mou_6	loc_og_t2c_mou_7	loc_og_t2c_mou_8		
loc_og_t2c_mou_9 \				
77	NaN	0.0	3.9	
0.00				
364	NaN	0.0	0.0	
2.78				
423	NaN	0.0	0.0	
0.00				
934	NaN	0.0	0.0	

0.00

1187	NaN	0.0	0.0
------	-----	-----	-----

0.00

	loc_og_mou_6	loc_og_mou_7	loc_og_mou_8	loc_og_mou_9	\
77	NaN	67.33	281.19	63.99	
364	NaN	129.14	601.66	555.44	
423	NaN	0.00	0.58	0.33	
934	NaN	17.08	92.79	51.34	
1187	NaN	12.54	77.48	0.00	

	std_og_t2t_mou_6	std_og_t2t_mou_7	std_og_t2t_mou_8	std_og_t2t_mou_9	\
77	NaN	0.00	0.00		
0.00					
364	NaN	404.71	191.49		
507.26					
423	NaN	0.00	0.00		
0.00					
934	NaN	14.16	0.00		
0.00					
1187	NaN	1361.94	2202.03		
0.00					

	std_og_t2m_mou_6	std_og_t2m_mou_7	std_og_t2m_mou_8	std_og_t2m_mou_9	\
77	NaN	0.00	0.00		
0.00					
364	NaN	722.01	321.41		
302.91					
423	NaN	0.00	0.25		
0.00					
934	NaN	0.00	0.00		
0.00					
1187	NaN	0.00	1.13		
0.00					

	std_og_t2f_mou_6	std_og_t2f_mou_7	std_og_t2f_mou_8	std_og_t2f_mou_9	\
77	NaN	6.35	40.09		
0.0					
364	NaN	0.00	0.00		
0.0					
423	NaN	0.00	0.61		
0.0					
934	NaN	0.00	0.00		
0.0					
1187	NaN	0.00	0.00		
0.0					

	std_og_t2c_mou_6	std_og_t2c_mou_7	std_og_t2c_mou_8
std_og_t2c_mou_9 \			
77	NaN	0.0	0.0
0.0			
364	NaN	0.0	0.0
0.0			
423	NaN	0.0	0.0
0.0			
934	NaN	0.0	0.0
0.0			
1187	NaN	0.0	0.0
0.0			

	std_og_mou_6	std_og_mou_7	std_og_mou_8	std_og_mou_9
isd_og_mou_6 \				
77	NaN	6.35	40.09	0.00
NaN				
364	NaN	1126.73	512.91	810.18
NaN				
423	NaN	0.00	0.86	0.00
NaN				
934	NaN	14.16	0.00	0.00
NaN				
1187	NaN	1361.94	2203.16	0.00
NaN				

	isd_og_mou_7	isd_og_mou_8	isd_og_mou_9	spl_og_mou_6
spl_og_mou_7 \				
77	2.93	28.04	3.25	NaN
0.00				
364	0.00	0.00	0.00	NaN
45.14				
423	10.03	0.00	0.01	NaN
0.00				
934	20.13	8.41	0.00	NaN
0.00				
1187	0.00	0.00	0.00	NaN
3.34				

	spl_og_mou_8	spl_og_mou_9	og_others_6	og_others_7
og_others_8 \				
77	7.58	0.00	NaN	0.0
0.0				
364	13.84	37.74	NaN	0.0
0.0				
423	0.00	0.00	NaN	0.0
0.0				
934	0.00	0.00	NaN	0.0
0.0				
1187	1.78	0.00	NaN	0.0

0.0

	og_others_9	total_og_mou_6	total_og_mou_7	total_og_mou_8	\
77	0.0	0.0	76.61	356.93	
364	0.0	0.0	1301.03	1128.43	
423	0.0	0.0	10.03	1.45	
934	0.0	0.0	51.38	101.21	
1187	0.0	0.0	1377.84	2282.43	

	total_og_mou_9	loc_ic_t2t_mou_6	loc_ic_t2t_mou_7
loc_ic_t2t_mou_8	\		
77	67.24	NaN	79.46
191.24			
364	1403.38	NaN	7.41
10.23			
423	0.34	NaN	0.00
0.00			
934	51.34	NaN	0.39
20.09			
1187	0.00	NaN	19.34
56.38			

	loc_ic_t2t_mou_9	loc_ic_t2m_mou_6	loc_ic_t2m_mou_7
loc_ic_t2m_mou_8	\		
77	5.26	NaN	43.31
94.18			
364	17.46	NaN	69.39
93.48			
423	0.00	NaN	0.00
0.00			
934	12.19	NaN	4.53
51.16			
1187	0.00	NaN	28.19
16.31			

	loc_ic_t2m_mou_9	loc_ic_t2f_mou_6	loc_ic_t2f_mou_7
loc_ic_t2f_mou_8	\		
77	16.39	NaN	2.03
0.00			
364	44.89	NaN	0.00
0.83			
423	0.00	NaN	0.00
0.00			
934	59.83	NaN	7.80
17.08			
1187	0.00	NaN	0.00
0.00			

	loc_ic_t2f_mou_9	loc_ic_mou_6	loc_ic_mou_7	loc_ic_mou_8	\
77	15.78	NaN	124.81	285.43	

364	0.00	NaN	76.81	104.54
423	0.00	NaN	0.00	0.00
934	5.13	NaN	12.73	88.34
1187	0.00	NaN	47.54	72.69

	loc_ic_mou_9	std_ic_t2t_mou_6	std_ic_t2t_mou_7
std_ic_t2t_mou_8 \			
77	37.44	NaN	8.00
0.00			
364	62.36	NaN	5.81
10.09			
423	0.00	NaN	0.00
0.00			
934	77.16	NaN	0.00
0.00			
1187	0.00	NaN	125.44
149.81			

	std_ic_t2t_mou_9	std_ic_t2m_mou_6	std_ic_t2m_mou_7
std_ic_t2m_mou_8 \			
77	0.00	NaN	0.00
0.00			
364	22.36	NaN	37.94
86.63			
423	0.00	NaN	0.00
0.00			
934	0.00	NaN	0.00
0.00			
1187	0.00	NaN	9.84
17.06			

	std_ic_t2m_mou_9	std_ic_t2f_mou_6	std_ic_t2f_mou_7
std_ic_t2f_mou_8 \			
77	0.00	NaN	0.0
0.00			
364	34.49	NaN	0.0
0.00			
423	0.00	NaN	0.0
0.36			
934	0.00	NaN	0.0
0.00			
1187	0.00	NaN	0.0
0.00			

	std_ic_t2f_mou_9	std_ic_t2o_mou_6	std_ic_t2o_mou_7
std_ic_t2o_mou_8 \			
77	15.93	NaN	0.0
0.0			
364	0.00	NaN	0.0
0.0			

423	0.00	NaN	0.0		
0.0					
934	0.00	NaN	0.0		
0.0					
1187	0.00	NaN	0.0		
0.0					
	std_ic_t2o_mou_9	std_ic_mou_6	std_ic_mou_7	std_ic_mou_8	\
77	0.0	NaN	8.00	0.00	
364	0.0	NaN	43.76	96.73	
423	0.0	NaN	0.00	0.36	
934	0.0	NaN	0.00	0.00	
1187	0.0	NaN	135.29	166.88	
	std_ic_mou_9	total_ic_mou_6	total_ic_mou_7	total_ic_mou_8	\
77	15.93	0.0	135.38	289.33	
364	56.86	0.0	185.14	219.59	
423	0.00	0.0	8.31	0.36	
934	0.00	0.0	14.69	100.94	
1187	0.00	0.0	182.84	239.58	
	total_ic_mou_9	spl_ic_mou_6	spl_ic_mou_7	spl_ic_mou_8	
spl_ic_mou_9	\				
77	53.38	NaN	0.0	0.0	
0.0					
364	129.19	NaN	0.0	0.0	
0.0					
423	0.00	NaN	0.0	0.0	
0.0					
934	78.99	NaN	0.0	0.0	
0.0					
1187	0.00	NaN	0.0	0.0	
0.0					
	isd_ic_mou_6	isd_ic_mou_7	isd_ic_mou_8	isd_ic_mou_9	
ic_others_6	\				
77	NaN	2.56	0.50	0.00	
NaN					
364	NaN	64.56	18.31	9.96	
NaN					
423	NaN	8.31	0.00	0.00	
NaN					
934	NaN	1.96	12.59	1.83	
NaN					
1187	NaN	0.00	0.00	0.00	
NaN					
	ic_others_7	ic_others_8	ic_others_9	total_rech_num_6	\
77	0.0	3.39	0.0	4	
364	0.0	0.00	0.0	4	

423	0.0	0.00	0.0	4
934	0.0	0.00	0.0	3
1187	0.0	0.00	0.0	2
total_rech_num_7 total_rech_num_8 total_rech_num_9				
total_rech_amt_6 \				
77	5	3	3	
0				
364	12	24	20	
0				
423	4	3	3	
252				
934	4	9	4	
0				
1187	20	24	6	
0				
total_rech_amt_7 total_rech_amt_8 total_rech_amt_9				
max_rech_amt_6 \				
77	1154	750	0	
0				
364	970	1104	1214	
0				
423	591	0	382	
252				
934	1302	150	108	
0				
1187	883	1160	0	
0				
max_rech_amt_7 max_rech_amt_8 max_rech_amt_9				
last_day_rch_amt_6 \				
77	1000	750	0	
0				
364	154	154	250	
0				
423	339	0	252	
252				
934	550	150	54	
0				
1187	150	250	0	
0				
last_day_rch_amt_7 last_day_rch_amt_8 last_day_rch_amt_9				
vol_2g_mb_6 \				
77	0	750	0	
0.0				
364	50	50	0	
0.0				
423	0	0	0	

3.3			
934	0	150	0
0.0			
1187	30	0	0
0.0			

	vol_2g_mb_7	vol_2g_mb_8	vol_2g_mb_9	vol_3g_mb_6	vol_3g_mb_7
\					
77	96.48	0.00	0.00	0.00	0.00
364	565.78	2108.66	0.00	0.00	0.00
423	38.45	0.00	4.52	669.36	837.18
934	0.31	38.77	78.66	0.00	1045.79
1187	0.00	0.00	0.00	0.00	0.00

	vol_3g_mb_8	vol_3g_mb_9	monthly_2g_6	monthly_2g_7
monthly_2g_8 \				
77	0.00	0.00	0	1
0				
364	0.00	0.00	0	1
1				
423	0.00	423.59	0	0
0				
934	245.91	471.48	0	0
0				
1187	0.00	0.00	0	0
0				

	monthly_2g_9	sachet_2g_6	sachet_2g_7	sachet_2g_8	sachet_2g_9
\					
77	0	0	0	0	0
364	0	0	0	2	0
423	0	0	0	0	0
934	0	0	0	0	0
1187	0	0	0	0	0

	monthly_3g_6	monthly_3g_7	monthly_3g_8	monthly_3g_9
sachet_3g_6 \				
77	0	0	0	0
0				
364	0	0	0	0
0				

423	1	1	0	1
0				
934	0	1	1	0
0				
1187	0	0	0	0
0				

	sachet_3g_7	sachet_3g_8	sachet_3g_9	aon	aug_vbc_3g
jul_vbc_3g \					
77	0	0	0	1894	0.00
0.00					
364	0	1	0	424	0.00
0.00					
423	0	0	0	945	73.55
266.94					
934	0	2	1	490	188.83
215.00					
1187	0	0	0	737	0.00
0.00					

	jun_vbc_3g	sep_vbc_3g	avg_rech_amt_6_7
77	0.00	0.00	577.0
364	0.00	0.00	485.0
423	63.04	0.00	421.5
934	0.00	24.18	651.0
1187	0.00	0.00	441.5

Deleting the records for which MOU for Jun(6) are null

```
df = df.drop(df_null_mou_6.index)
```

Again cheking percent of missing values in columns

```
df_missing_columns =
(round(((df.isnull().sum())/len(df.index))*100),2).to_frame('null')).so
rt_values('null', ascending=False)
df_missing_columns
```

	null
loc_ic_t2f_mou_7	0.12
isd_ic_mou_7	0.12
loc_og_t2f_mou_7	0.12
loc_og_t2c_mou_7	0.12
loc_og_mou_7	0.12
std_og_t2t_mou_7	0.12
std_og_t2f_mou_7	0.12
std_og_t2c_mou_7	0.12
std_og_mou_7	0.12
ic_others_7	0.12
isd_og_mou_7	0.12
spl_og_mou_7	0.12
loc_og_t2t_mou_7	0.12

og_others_7	0.12
spl_ic_mou_7	0.12
loc_ic_t2t_mou_7	0.12
std_ic_mou_7	0.12
loc_ic_t2m_mou_7	0.12
std_ic_t2o_mou_7	0.12
std_ic_t2f_mou_7	0.12
loc_ic_mou_7	0.12
std_ic_t2t_mou_7	0.12
loc_og_t2m_mou_7	0.12
std_og_t2m_mou_7	0.12
std_ic_t2m_mou_7	0.12
roam_ic_mou_7	0.12
onnet_mou_7	0.12
roam_og_mou_7	0.12
offnet_mou_7	0.12
isd_ic_mou_8	0.00
...	...
loc_ic_t2m_mou_8	0.00
loc_ic_t2f_mou_6	0.00
std_og_t2f_mou_6	0.00
loc_og_t2o_mou	0.00
loc_ic_t2f_mou_8	0.00
loc_ic_t2f_mou_9	0.00
loc_ic_mou_6	0.00
loc_ic_mou_8	0.00
loc_ic_mou_9	0.00
std_ic_t2t_mou_6	0.00
total_og_mou_7	0.00
total_og_mou_6	0.00
og_others_9	0.00
og_others_8	0.00
std_og_t2f_mou_8	0.00
std_og_t2f_mou_9	0.00
std_og_t2c_mou_6	0.00
std_og_t2c_mou_8	0.00
std_og_t2c_mou_9	0.00
std_og_mou_6	0.00
std_og_mou_8	0.00
std_og_mou_9	0.00
isd_og_mou_6	0.00
isd_og_mou_8	0.00
isd_og_mou_9	0.00
spl_og_mou_6	0.00
spl_og_mou_8	0.00
spl_og_mou_9	0.00
og_others_6	0.00
avg_rech_amt_6_7	0.00

```
[178 rows x 1 columns]
```

Looks like MOU for all the types of calls for the month of July (7) have missing values together for any particular record.

Lets check the records for the MOU for Jul(7), in which these coulmnns have missing values together.

```
# Listing the columns of MOU Jul(7)
print((df_missing_columns[df_missing_columns['null'] ==
0.12]).index().to_list())

['loc_ic_t2f_mou_7', 'isd_ic_mou_7', 'loc_og_t2f_mou_7',
'loc_og_t2c_mou_7', 'loc_og_mou_7', 'std_og_t2t_mou_7',
'std_og_t2f_mou_7', 'std_og_t2c_mou_7', 'std_og_mou_7', 'ic_others_7',
'isd_og_mou_7', 'spl_og_mou_7', 'loc_og_t2t_mou_7', 'og_others_7',
'spl_ic_mou_7', 'loc_ic_t2t_mou_7', 'std_ic_mou_7',
'loc_ic_t2m_mou_7', 'std_ic_t2o_mou_7', 'std_ic_t2f_mou_7',
'loc_ic_mou_7', 'std_ic_t2t_mou_7', 'loc_og_t2m_mou_7',
'std_og_t2m_mou_7', 'std_ic_t2m_mou_7', 'roam_ic_mou_7',
'onnet_mou_7', 'roam_og_mou_7', 'offnet_mou_7']

# Creating a dataframe with the condition, in which MOU for Jul(7) are
null
df_null_mou_7 = df[(df['loc_og_t2m_mou_7'].isnull()) &
(df['loc_ic_t2f_mou_7'].isnull()) & (df['roam_og_mou_7'].isnull()) &
(df['std_ic_t2m_mou_7'].isnull()) &
(df['loc_og_t2t_mou_7'].isnull()) &
(df['std_ic_t2t_mou_7'].isnull()) & (df['loc_og_t2f_mou_7'].isnull())
& (df['loc_ic_mou_7'].isnull()) &
(df['loc_og_t2c_mou_7'].isnull()) & (df['loc_og_mou_7'].isnull()) &
(df['std_og_t2t_mou_7'].isnull()) & (df['roam_ic_mou_7'].isnull()) &
(df['loc_ic_t2m_mou_7'].isnull()) &
(df['std_og_t2m_mou_7'].isnull()) & (df['loc_ic_t2t_mou_7'].isnull())
& (df['std_og_t2f_mou_7'].isnull()) &
(df['std_og_t2c_mou_7'].isnull()) & (df['og_others_7'].isnull()) &
(df['std_og_mou_7'].isnull()) & (df['spl_og_mou_7'].isnull()) &
(df['std_ic_t2f_mou_7'].isnull()) & (df['isd_og_mou_7'].isnull()) &
(df['std_ic_mou_7'].isnull()) & (df['offnet_mou_7'].isnull()) &
(df['isd_ic_mou_7'].isnull()) & (df['ic_others_7'].isnull()) &
(df['std_ic_t2o_mou_7'].isnull()) & (df['onnet_mou_7'].isnull()) &
(df['spl_ic_mou_7'].isnull())]

df_null_mou_7.head()

   mobile_number  loc_og_t2o_mou  std_og_t2o_mou
loc_ic_t2o_mou \
5616      7001238202           0.0           0.0           0.0
```

9451	7001477649		0.0	0.0	0.0	
9955	7001658068		0.0	0.0	0.0	
10724	7001391499		0.0	0.0	0.0	
12107	7000131738		0.0	0.0	0.0	
	arpu_6	arpu_7	arpu_8	arpu_9	onnet_mou_6	onnet_mou_7
\						
5616	760.815	531.088	992.818	1144.676	324.91	NaN
9451	1129.566	0.000	128.252	802.648	11.89	NaN
9955	925.028	189.000	789.761	445.707	46.39	NaN
10724	894.818	85.000	207.040	363.314	117.21	NaN
12107	1803.475	0.000	0.600	25.243	1742.61	NaN
	onnet_mou_8	onnet_mou_9	offnet_mou_6	offnet_mou_7		
offnet_mou_8	\					
5616	386.13	1180.29	350.29	NaN		
399.64						
9451	1.46	33.89	259.18	NaN		
26.21						
9955	43.39	56.61	333.78	NaN		
196.53						
10724	97.01	35.43	119.79	NaN		
12.79						
12107	0.00	0.00	278.79	NaN		
14.29						
	offnet_mou_9	roam_ic_mou_6	roam_ic_mou_7	roam_ic_mou_8	\	
5616	887.76	463.63	NaN	221.46		
9451	241.18	9.98	NaN	1.73		
9955	144.73	0.00	NaN	0.00		
10724	92.04	0.00	NaN	0.00		
12107	4.50	0.00	NaN	0.00		
	roam_ic_mou_9	roam_og_mou_6	roam_og_mou_7	roam_og_mou_8	\	
5616	0.0	505.71	NaN	175.93		
9451	0.0	5.66	NaN	2.46		
9955	0.0	0.00	NaN	0.00		
10724	0.0	0.00	NaN	0.00		
12107	0.0	0.00	NaN	0.00		
	roam_og_mou_9	loc_og_t2t_mou_6	loc_og_t2t_mou_7			
loc_og_t2t_mou_8	\					

5616	0.0	145.91	NaN
243.43			
9451	0.0	6.73	NaN
1.46			
9955	0.0	46.39	NaN
43.39			
10724	0.0	115.08	NaN
97.01			
12107	0.0	96.08	NaN
0.00			
loc_og_t2t_mou_9 loc_og_t2m_mou_6 loc_og_t2m_mou_7			
loc_og_t2m_mou_8 \			
5616	1108.38	0.85	NaN
184.78			
9451	20.84	171.46	NaN
20.54			
9955	56.61	227.91	NaN
163.68			
10724	34.98	86.39	NaN
6.59			
12107	0.00	64.98	NaN
0.86			
loc_og_t2m_mou_9 loc_og_t2f_mou_6 loc_og_t2f_mou_7			
loc_og_t2f_mou_8 \			
5616	300.19	1.13	NaN
7.94			
9451	148.88	0.00	NaN
0.00			
9955	121.54	104.69	NaN
28.96			
10724	55.44	17.18	NaN
6.19			
12107	0.00	0.00	NaN
0.00			
loc_og_t2f_mou_9 loc_og_t2c_mou_6 loc_og_t2c_mou_7			
loc_og_t2c_mou_8 \			
5616	67.11	0.00	NaN
12.51			
9451	0.00	0.00	NaN
0.00			
9955	21.04	0.00	NaN
0.00			
10724	28.08	0.00	NaN
0.00			
12107	0.00	50.03	NaN
13.43			

	loc_og_t2c_mou_9	loc_og_mou_6	loc_og_mou_7	loc_og_mou_8	\
5616	18.89	147.89	NaN	436.16	
9451	0.00	178.19	NaN	22.01	
9955	0.00	379.01	NaN	236.04	
10724	0.05	218.66	NaN	109.81	
12107	4.50	161.06	NaN	0.86	
	loc_og_mou_9	std_og_t2t_mou_6	std_og_t2t_mou_7		
std_og_t2t_mou_8	\				
5616	1475.69	0.96	NaN		
17.06					
9451	169.73	5.16	NaN		
0.00					
9955	199.21	0.00	NaN		
0.00					
10724	118.51	2.13	NaN		
0.00					
12107	0.00	1646.53	NaN		
0.00					
	std_og_t2t_mou_9	std_og_t2m_mou_6	std_og_t2m_mou_7		
std_og_t2m_mou_8	\				
5616	69.51	15.91	NaN		
144.04					
9451	13.05	0.00	NaN		
0.00					
9955	0.00	0.00	NaN		
0.00					
10724	0.45	2.43	NaN		
0.00					
12107	0.00	140.16	NaN		
0.00					
	std_og_t2m_mou_9	std_og_t2f_mou_6	std_og_t2f_mou_7		
std_og_t2f_mou_8	\				
5616	490.61	0.00	NaN		
0.0					
9451	0.00	0.00	NaN		
0.0					
9955	1.26	1.16	NaN		
2.9					
10724	7.18	6.09	NaN		
0.0					
12107	0.00	1.26	NaN		
0.0					
	std_og_t2f_mou_9	std_og_t2c_mou_6	std_og_t2c_mou_7		
std_og_t2c_mou_8	\				
5616	13.33	0.0	NaN		
0.0					

9451	0.00	0.0	NaN
0.0			
9955	0.00	0.0	NaN
0.0			
10724	1.28	0.0	NaN
0.0			
12107	0.00	0.0	NaN
0.0			

	std_og_t2c_mou_9	std_og_mou_6	std_og_mou_7	std_og_mou_8	\
5616	0.0	16.88	NaN	161.11	
9451	0.0	5.16	NaN	0.00	
9955	0.0	1.16	NaN	2.90	
10724	0.0	10.66	NaN	0.00	
12107	0.0	1787.96	NaN	0.00	

	std_og_mou_9	isd_og_mou_6	isd_og_mou_7	isd_og_mou_8
isd_og_mou_9 \				
5616	573.46	0.00	NaN	0.00
0.00				
9451	13.05	74.91	NaN	4.74
92.29				
9955	1.26	53.14	NaN	31.06
33.69				
10724	8.91	16.86	NaN	6.21
2.18				
12107	0.00	0.00	NaN	0.00
0.00				

	spl_og_mou_6	spl_og_mou_7	spl_og_mou_8	spl_og_mou_9
og_others_6 \				
5616	4.71	NaN	12.56	18.89
0.00				
9451	7.13	NaN	0.00	1.08
0.00				
9955	0.00	NaN	0.00	0.00
0.00				
10724	0.00	NaN	0.00	0.05
0.00				
12107	72.61	NaN	13.43	4.50
1.76				

	og_others_7	og_others_8	og_others_9	total_og_mou_6
total_og_mou_7 \				
5616	NaN	0.0	0.0	169.49
0.0				
9451	NaN	0.0	0.0	265.41
0.0				
9955	NaN	0.0	0.0	433.33
0.0				

10724	NaN	0.0	0.0	246.19
0.0				
12107	NaN	0.0	0.0	2023.41
0.0				
	total_og_mou_8	total_og_mou_9	loc_ic_t2t_mou_6	
loc_ic_t2t_mou_7 \				
5616	609.84	2068.06	78.76	
NaN				
9451	26.76	276.16	17.24	
NaN				
9955	270.01	234.18	80.98	
NaN				
10724	116.03	129.66	887.04	
NaN				
12107	14.29	4.50	65.76	
NaN				
	loc_ic_t2t_mou_8	loc_ic_t2t_mou_9	loc_ic_t2m_mou_6	
loc_ic_t2m_mou_7 \				
5616	233.66	558.84	1.36	
NaN				
9451	0.60	36.69	130.09	
NaN				
9955	32.69	112.14	201.38	
NaN				
10724	200.51	408.66	104.18	
NaN				
12107	1.73	5.88	92.18	
NaN				
	loc_ic_t2m_mou_8	loc_ic_t2m_mou_9	loc_ic_t2f_mou_6	
loc_ic_t2f_mou_7 \				
5616	11.53	75.31	6.61	
NaN				
9451	16.54	110.19	25.46	
NaN				
9955	169.24	155.58	41.68	
NaN				
10724	22.24	76.39	16.74	
NaN				
12107	5.59	2.75	0.00	
NaN				
	loc_ic_t2f_mou_8	loc_ic_t2f_mou_9	loc_ic_mou_6	loc_ic_mou_7
\				
5616	0.00	31.81	86.74	NaN
9451	8.76	40.24	172.81	NaN

9955	25.68	12.33	324.04	NaN
10724	1.61	28.18	1007.98	NaN
12107	0.00	0.00	157.94	NaN
	loc_ic_mou_8	loc_ic_mou_9	std_ic_t2t_mou_6	std_ic_t2t_mou_7
\				
5616	245.19	665.98	0.00	NaN
9451	25.91	187.14	1.50	NaN
9955	227.63	280.06	0.00	NaN
10724	224.38	513.24	0.00	NaN
12107	7.33	8.63	103.66	NaN
	std_ic_t2t_mou_8	std_ic_t2t_mou_9	std_ic_t2m_mou_6	
std_ic_t2m_mou_7	\			
5616	12.13	42.39	21.76	
NaN				
9451	0.00	0.00	0.41	
NaN				
9955	0.00	0.00	0.98	
NaN				
10724	0.00	0.00	5.94	
NaN				
12107	0.00	0.00	3.01	
NaN				
	std_ic_t2m_mou_8	std_ic_t2m_mou_9	std_ic_t2f_mou_6	
std_ic_t2f_mou_7	\			
5616	110.99	263.98	0.0	
NaN				
9451	0.00	12.29	0.0	
NaN				
9955	2.13	2.58	0.0	
NaN				
10724	0.00	4.88	0.0	
NaN				
12107	0.00	0.00	0.0	
NaN				
	std_ic_t2f_mou_8	std_ic_t2f_mou_9	std_ic_t2o_mou_6	
std_ic_t2o_mou_7	\			
5616	0.00	6.43	0.0	
NaN				

9451	0.00	4.48	0.0
NaN			
9955	0.23	0.00	0.0
NaN			
10724	10.03	1.26	0.0
NaN			
12107	0.00	0.00	0.0
NaN			
std_ic_t2o_mou_8 std_ic_t2o_mou_9 std_ic_mou_6 std_ic_mou_7			
\			
5616	0.0	0.0	21.76 NaN
9451	0.0	0.0	1.91 NaN
9955	0.0	0.0	0.98 NaN
10724	0.0	0.0	5.94 NaN
12107	0.0	0.0	106.68 NaN
std_ic_mou_8 std_ic_mou_9 total_ic_mou_6 total_ic_mou_7			
\			
5616	123.13	312.81	189.81 0.0
9451	0.00	16.78	217.33 0.0
9955	2.36	2.58	332.33 0.0
10724	10.03	6.14	1140.54 0.0
12107	0.00	0.00	265.03 0.0
total_ic_mou_8 total_ic_mou_9 spl_ic_mou_6 spl_ic_mou_7			
\			
5616	397.13	1020.16	0.00 NaN
9451	43.44	307.43	0.00 NaN
9955	506.94	526.54	0.00 NaN
10724	342.78	642.33	0.14 NaN
12107	7.33	8.63	0.00 NaN
spl_ic_mou_8 spl_ic_mou_9 isd_ic_mou_6 isd_ic_mou_7			
isd_ic_mou_8	\		
5616	0.00	0.13	81.29 NaN
28.79			
9451	0.00	0.00	42.59 NaN
17.53			
9955	0.00	0.00	7.29 NaN
173.61			
10724	0.08	0.09	126.13 NaN
106.53			
12107	0.00	0.00	0.00 NaN
0.00			
isd_ic_mou_9 ic_others_6 ic_others_7 ic_others_8			

ic_others_9 \				
5616	41.23	0.00	NaN	0.00
0.00				
9451	103.49	0.00	NaN	0.00
0.00				
9955	229.44	0.00	NaN	103.33
14.45				
10724	116.83	0.33	NaN	1.74
5.99				
12107	0.00	0.40	NaN	0.00
0.00				

	total_rech_num_6	total_rech_num_7	total_rech_num_8
total_rech_num_9 \			
5616	5	7	9
13			
9451	14	4	4
9			
9955	6	1	4
3			
10724	8	3	3
5			
12107	17	2	1
2			

	total_rech_amt_6	total_rech_amt_7	total_rech_amt_8
total_rech_amt_9 \			
5616	776	780	904
1591			
9451	1206	0	223
991			
9955	1385	0	835
912			
10724	1020	0	360
480			
12107	1990	0	0
30			

	max_rech_amt_6	max_rech_amt_7	max_rech_amt_8	max_rech_amt_9
\				
5616	250	330	200	289
9451	250	0	130	130
9955	350	0	300	479
10724	500	0	130	150
12107	250	0	0	30

	last_day_rch_amt_6	last_day_rch_amt_7	last_day_rch_amt_8	\
5616	250	0	130	
9451	250	0	130	
9955	250	0	300	
10724	500	0	130	
12107	110	0	0	

	last_day_rch_amt_9	vol_2g_mb_6	vol_2g_mb_7	vol_2g_mb_8
vol_2g_mb_9 \				
5616	250	0.00	0.0	11.26
83.32				
9451	130	321.86	0.0	0.00
431.85				
9955	479	0.00	0.0	0.00
0.00				
10724	0	0.00	0.0	0.00
0.00				
12107	30	0.00	0.0	0.00
0.00				

	vol_3g_mb_6	vol_3g_mb_7	vol_3g_mb_8	vol_3g_mb_9
monthly_2g_6 \				
5616	0.0	0.0	79.94	668.4
0				
9451	0.0	0.0	0.00	0.0
1				
9955	0.0	0.0	0.00	0.0
0				
10724	0.0	0.0	0.00	0.0
0				
12107	0.0	0.0	0.00	0.0
0				

	monthly_2g_7	monthly_2g_8	monthly_2g_9	sachet_2g_6
sachet_2g_7 \				
5616	0	1	1	0
0				
9451	0	0	1	1
0				
9955	0	0	0	0
0				
10724	0	0	0	0
0				
12107	0	0	0	0
0				

	sachet_2g_8	sachet_2g_9	monthly_3g_6	monthly_3g_7
monthly_3g_8 \				
5616	0	0	0	0

0				
9451	0	2	0	0
0				
9955	0	0	0	0
0				
10724	0	0	0	0
0				
12107	0	0	0	0
0				

	monthly_3g_9	sachet_3g_6	sachet_3g_7	sachet_3g_8
sachet_3g_9 aon \				
5616	0	0	0	0
0 576				
9451	0	0	0	0
0 672				
9955	0	0	0	0
0 3107				
10724	0	0	0	0
0 2664				
12107	0	0	0	0
0 219				

	aug_vbc_3g	jul_vbc_3g	jun_vbc_3g	sep_vbc_3g
avg_rech_amt_6_7				
5616	63.38	0.0	0.0	163.39
778.0				
9451	0.00	0.0	0.0	0.00
603.0				
9955	0.00	0.0	0.0	0.00
692.5				
10724	0.00	0.0	0.0	0.00
510.0				
12107	0.00	0.0	0.0	0.00
995.0				

Deleting the records for which MOU for Jul(7) are null

```
df = df.drop(df_null_mou_7.index)
```

Again cheking percent of missing values in columns

```
df_missing_columns =
(round(((df.isnull().sum())/len(df.index))*100),2).to_frame('null')).so
rt_values('null', ascending=False)
df_missing_columns
```

	null
mobile_number	0.0
total_rech_num_7	0.0
std_ic_mou_7	0.0
std_ic_mou_8	0.0

std_ic_mou_9	0.0
total_ic_mou_6	0.0
total_ic_mou_7	0.0
total_ic_mou_8	0.0
total_ic_mou_9	0.0
spl_ic_mou_6	0.0
spl_ic_mou_7	0.0
spl_ic_mou_8	0.0
spl_ic_mou_9	0.0
isd_ic_mou_6	0.0
isd_ic_mou_7	0.0
isd_ic_mou_8	0.0
isd_ic_mou_9	0.0
ic_others_6	0.0
ic_others_7	0.0
ic_others_8	0.0
ic_others_9	0.0
std_ic_mou_6	0.0
std_ic_t2o_mou_9	0.0
std_ic_t2o_mou_8	0.0
std_ic_t2t_mou_9	0.0
loc_ic_t2f_mou_9	0.0
loc_ic_mou_6	0.0
loc_ic_mou_7	0.0
loc_ic_mou_8	0.0
loc_ic_mou_9	0.0
...	...
loc_ic_t2t_mou_6	0.0
loc_ic_t2t_mou_7	0.0
loc_ic_t2t_mou_8	0.0
loc_ic_t2t_mou_9	0.0
loc_ic_t2m_mou_6	0.0
loc_ic_t2m_mou_7	0.0
loc_ic_t2m_mou_8	0.0
spl_og_mou_6	0.0
isd_og_mou_8	0.0
std_og_t2t_mou_8	0.0
std_og_t2f_mou_9	0.0
std_og_t2t_mou_9	0.0
std_og_t2m_mou_6	0.0
std_og_t2m_mou_7	0.0
std_og_t2m_mou_8	0.0
std_og_t2m_mou_9	0.0
std_og_t2f_mou_6	0.0
std_og_t2f_mou_7	0.0
std_og_t2f_mou_8	0.0
std_og_t2c_mou_6	0.0
isd_og_mou_7	0.0
std_og_t2c_mou_7	0.0

```
std_og_t2c_mou_8    0.0
std_og_t2c_mou_9    0.0
std_og_mou_6        0.0
std_og_mou_7        0.0
std_og_mou_8        0.0
std_og_mou_9        0.0
isd_og_mou_6        0.0
avg_rech_amt_6_7    0.0
```

```
[178 rows x 1 columns]
```

We can see there are no more missing values in any columns.

```
df.shape
```

```
(27991, 178)
```

```
# Checking percentage of rows we have lost while handling the missing values
```

```
round((1- (len(df.index)/30011)),2)
```

```
0.07
```

We can see that we have lost almost 7% records. But we have enough number of records to do our analysis.

Tag churners

Now tag the churned customers (churn=1, else 0) based on the fourth month as follows: Those who have not made any calls (either incoming or outgoing) AND have not used mobile internet even once in the churn phase.

```
df['churn'] = np.where((df['total_ic_mou_9']==0) &
(df['total_og_mou_9']==0) & (df['vol_2g_mb_9']==0) &
(df['vol_3g_mb_9']==0), 1, 0)
```

```
df.head()
```

	mobile_number	loc_og_t2o_mou	std_og_t2o_mou	loc_ic_t2o_mou
arpu_6 \				
8	7001524846	0.0	0.0	0.0
378.721				
13	7002191713	0.0	0.0	0.0
492.846				
16	7000875565	0.0	0.0	0.0
430.975				
17	7000187447	0.0	0.0	0.0
690.008				
21	7002124215	0.0	0.0	0.0
514.453				

	arpu_7	arpu_8	arpu_9	onnet_mou_6	onnet_mou_7	
onnet_mou_8 \						
8	492.223	137.362	166.787	413.69	351.03	35.08
13	205.671	593.260	322.732	501.76	108.39	534.24
16	299.869	187.894	206.490	50.51	74.01	70.61
17	18.980	25.499	257.583	1185.91	9.28	7.79
21	597.753	637.760	578.596	102.41	132.11	85.14

	onnet_mou_9	offnet_mou_6	offnet_mou_7	offnet_mou_8
offnet_mou_9 \				
8	33.46	94.66	80.63	136.48
108.71				
13	244.81	413.31	119.28	482.46
214.06				
16	31.34	296.29	229.74	162.76
224.39				
17	558.51	61.64	0.00	5.54
87.89				
21	161.63	757.93	896.68	983.39
869.89				

	roam_ic_mou_6	roam_ic_mou_7	roam_ic_mou_8	roam_ic_mou_9
roam_og_mou_6 \				
8	0.00	0.00	0.00	0.00
0.00				
13	23.53	144.24	72.11	136.78
7.98				
16	0.00	2.83	0.00	0.00
0.00				
17	0.00	4.76	4.81	0.00
0.00				
21	0.00	0.00	0.00	0.00
0.00				

	roam_og_mou_7	roam_og_mou_8	roam_og_mou_9	loc_og_t2t_mou_6 \
8	0.00	0.00	0.00	297.13
13	35.26	1.44	12.78	49.63
16	17.74	0.00	0.00	42.61
17	8.46	13.34	17.98	38.99
21	0.00	0.00	0.00	4.48

	loc_og_t2t_mou_7	loc_og_t2t_mou_8	loc_og_t2t_mou_9
loc_og_t2m_mou_6 \			
8	217.59	12.49	26.13

80.96				
13	6.19	36.01	6.14	
151.13				
16	65.16	67.38	26.88	
273.29				
17	0.00	0.00	36.41	
58.54				
21	6.16	23.34	29.98	
91.81				
	loc_og_t2m_mou_7	loc_og_t2m_mou_8	loc_og_t2m_mou_9	
loc_og_t2f_mou_6 \				
8	70.58	50.54	34.58	
0.00				
13	47.28	294.46	108.24	
4.54				
16	145.99	128.28	201.49	
0.00				
17	0.00	0.00	9.38	
0.00				
21	87.93	104.81	107.54	
0.75				
	loc_og_t2f_mou_7	loc_og_t2f_mou_8	loc_og_t2f_mou_9	
loc_og_t2c_mou_6 \				
8	0.00	0.00	0.00	
0.0				
13	0.00	23.51	5.29	
0.0				
16	4.48	10.26	4.66	
0.0				
17	0.00	0.00	0.00	
0.0				
21	0.00	1.58	0.00	
0.0				
	loc_og_t2c_mou_7	loc_og_t2c_mou_8	loc_og_t2c_mou_9	loc_og_mou_6
\				
8	0.0	7.15	0.0	378.09
13	0.0	0.49	0.0	205.31
16	0.0	0.00	0.0	315.91
17	0.0	0.00	0.0	97.54
21	0.0	0.00	0.0	97.04
	loc_og_mou_7	loc_og_mou_8	loc_og_mou_9	std_og_t2t_mou_6 \

8	288.18	63.04	60.71	116.56
13	53.48	353.99	119.69	446.41
16	215.64	205.93	233.04	7.89
17	0.00	0.00	45.79	1146.91
21	94.09	129.74	137.53	97.93
std_og_t2t_mou_7 std_og_t2t_mou_8 std_og_t2t_mou_9				
std_og_t2m_mou_6 \				
8	133.43	22.58	7.33	
13.69				
13	85.98	498.23	230.38	
255.36				
16	2.58	3.23	4.46	
22.99				
17	0.81	0.00	504.11	
1.55				
21	125.94	61.79	131.64	
665.36				
std_og_t2m_mou_7 std_og_t2m_mou_8 std_og_t2m_mou_9				
std_og_t2f_mou_6 \				
8	10.04	75.69	74.13	
0.0				
13	52.94	156.94	96.01	
0.0				
16	64.51	18.29	13.79	
0.0				
17	0.00	0.00	78.51	
0.0				
21	808.74	876.99	762.34	
0.0				
std_og_t2f_mou_7 std_og_t2f_mou_8 std_og_t2f_mou_9				
std_og_t2c_mou_6 \				
8	0.0	0.0	0.00	
0.0				
13	0.0	0.0	0.00	
0.0				
16	0.0	0.0	4.43	
0.0				
17	0.0	0.0	0.00	
0.0				
21	0.0	0.0	0.00	
0.0				
std_og_t2c_mou_7 std_og_t2c_mou_8 std_og_t2c_mou_9 std_og_mou_6				
\				
8	0.0	0.0	0.0	130.26
13	0.0	0.0	0.0	701.78

16	0.0	0.0	0.0	30.89
17	0.0	0.0	0.0	1148.46
21	0.0	0.0	0.0	763.29
std_og_mou_7 isd_og_mou_7 \	std_og_mou_8	std_og_mou_9	isd_og_mou_6	
8 143.48	98.28	81.46	0.0	
0.0				
13 138.93	655.18	326.39	0.0	
0.0				
16 67.09	21.53	22.69	0.0	
0.0				
17 0.81	0.00	582.63	0.0	
0.0				
21 934.69	938.79	893.99	0.0	
0.0				
isd_og_mou_8 spl_og_mou_8 \	isd_og_mou_9	spl_og_mou_6	spl_og_mou_7	
8 0.00	0.0	0.00	0.00	
10.23				
13 1.29	0.0	0.00	0.00	
4.78				
16 0.00	0.0	0.00	3.26	
5.91				
17 0.00	0.0	2.58	0.00	
0.00				
21 0.00	0.0	0.00	0.00	
0.00				
spl_og_mou_9 og_others_9 \	og_others_6	og_others_7	og_others_8	
8 0.00	0.00	0.0	0.0	0.0
13 0.00	0.00	0.0	0.0	0.0
16 0.00	0.00	0.0	0.0	0.0
17 2.64	0.93	0.0	0.0	0.0
21 0.00	0.00	0.0	0.0	0.0
total_og_mou_6 8 508.36	total_og_mou_7 431.66	total_og_mou_8 171.56	total_og_mou_9 142.18	\
13 907.09	192.41	1015.26	446.09	

16	346.81	286.01	233.38	255.74
17	1249.53	0.81	0.00	631.08
21	860.34	1028.79	1068.54	1031.53

	loc_ic_t2t_mou_6	loc_ic_t2t_mou_7	loc_ic_t2t_mou_8
loc_ic_t2t_mou_9 \			
8	23.84	9.84	0.31
4.03			
13	67.88	7.58	52.58
24.98			
16	41.33	71.44	28.89
50.23			
17	34.54	0.00	0.00
40.91			
21	2.48	10.19	19.54
17.99			

	loc_ic_t2m_mou_6	loc_ic_t2m_mou_7	loc_ic_t2m_mou_8
loc_ic_t2m_mou_9 \			
8	57.58	13.98	15.48
17.34			
13	142.88	18.53	195.18
104.79			
16	226.81	149.69	150.16
172.86			
17	47.41	2.31	0.00
43.86			
21	118.23	74.63	129.16
113.46			

	loc_ic_t2f_mou_6	loc_ic_t2f_mou_7	loc_ic_t2f_mou_8
loc_ic_t2f_mou_9 \			
8	0.00	0.00	0.00
0.00			
13	4.81	0.00	7.49
8.51			
16	8.71	8.68	32.71
65.21			
17	0.00	0.00	0.00
0.71			
21	4.61	2.84	10.39
8.41			

	loc_ic_mou_6	loc_ic_mou_7	loc_ic_mou_8	loc_ic_mou_9
std_ic_t2t_mou_6 \				
8	81.43	23.83	15.79	21.38
0.00				
13	215.58	26.11	255.26	138.29
115.68				
16	276.86	229.83	211.78	288.31

68.79				
17	81.96	2.31	0.00	85.49
8.63				
21	125.33	87.68	159.11	139.88
14.06				

	std_ic_t2t_mou_7	std_ic_t2t_mou_8	std_ic_t2t_mou_9
std_ic_t2m_mou_6 \			
8	0.58	0.10	0.00
22.43			
13	38.29	154.58	62.39
308.13			
16	78.64	6.33	16.66
18.68			
17	0.00	0.00	0.00
1.28			
21	5.98	0.18	16.74
67.69			

	std_ic_t2m_mou_7	std_ic_t2m_mou_8	std_ic_t2m_mou_9
std_ic_t2f_mou_6 \			
8	4.08	0.65	13.53
0.00			
13	29.79	317.91	151.51
0.00			
16	73.08	73.93	29.58
0.51			
17	0.00	0.00	1.63
0.00			
21	38.23	101.74	95.98
0.00			

	std_ic_t2f_mou_7	std_ic_t2f_mou_8	std_ic_t2f_mou_9
std_ic_t2o_mou_6 \			
8	0.0	0.00	0.0
0.0			
13	0.0	1.91	0.0
0.0			
16	0.0	2.18	0.0
0.0			
17	0.0	0.00	0.0
0.0			
21	0.0	0.00	0.0
0.0			

	std_ic_t2o_mou_7	std_ic_t2o_mou_8	std_ic_t2o_mou_9	std_ic_mou_6
\				
8	0.0	0.0	0.0	22.43
13	0.0	0.0	0.0	423.81

16	0.0	0.0	0.0	87.99
17	0.0	0.0	0.0	9.91
21	0.0	0.0	0.0	81.76

	std_ic_mou_7	std_ic_mou_8	std_ic_mou_9	total_ic_mou_6
total_ic_mou_7 \				
8	4.66	0.75	13.53	103.86
28.49				
13	68.09	474.41	213.91	968.61
172.58				
16	151.73	82.44	46.24	364.86
381.56				
17	0.00	0.00	1.63	91.88
2.31				
21	44.21	101.93	112.73	207.09
131.89				

	total_ic_mou_8	total_ic_mou_9	spl_ic_mou_6	spl_ic_mou_7
spl_ic_mou_8 \				
8	16.54	34.91	0.00	0.0
0.0				
13	1144.53	631.86	0.45	0.0
0.0				
16	294.46	334.56	0.00	0.0
0.0				
17	0.00	87.13	0.00	0.0
0.0				
21	261.04	252.61	0.00	0.0
0.0				

	spl_ic_mou_9	isd_ic_mou_6	isd_ic_mou_7	isd_ic_mou_8
isd_ic_mou_9 \				
8	0.0	0.00	0.00	0.00
0.00				
13	0.0	245.28	62.11	393.39
259.33				
16	0.0	0.00	0.00	0.23
0.00				
17	0.0	0.00	0.00	0.00
0.00				
21	0.0	0.00	0.00	0.00
0.00				

	ic_others_6	ic_others_7	ic_others_8	ic_others_9
total_rech_num_6 \				
8	0.00	0.00	0.00	0.00

19				
13	83.48	16.24	21.44	20.31
6				
16	0.00	0.00	0.00	0.00
10				
17	0.00	0.00	0.00	0.00
19				
21	0.00	0.00	0.00	0.00
22				

	total_rech_num_7	total_rech_num_8	total_rech_num_9
total_rech_amt_6 \			
8	21	14	15
437			
13	4	11	7
507			
16	6	2	1
570			
17	2	4	10
816			
21	26	27	17
600			

	total_rech_amt_7	total_rech_amt_8	total_rech_amt_9
max_rech_amt_6 \			
8	601	120	186
90			
13	253	717	353
110			
16	348	160	220
110			
17	0	30	335
110			
21	680	718	680
50			

	max_rech_amt_7	max_rech_amt_8	max_rech_amt_9	last_day_rch_amt_6
\				
8	154	30	36	50
13	110	130	130	110
16	110	130	220	100
17	0	30	130	30
21	50	50	50	30

last_day_rch_amt_7 last_day_rch_amt_8 last_day_rch_amt_9

vol_2g_mb_6 \			
8	0	10	0
0.0			
13	50	0	0
0.0			
16	100	130	220
0.0			
17	0	0	0
0.0			
21	20	50	30
0.0			

	vol_2g_mb_7	vol_2g_mb_8	vol_2g_mb_9	vol_3g_mb_6	vol_3g_mb_7 \
8	356.0	0.03	0.0	0.0	750.95
13	0.0	0.02	0.0	0.0	0.00
16	0.0	0.00	0.0	0.0	0.00
17	0.0	0.00	0.0	0.0	0.00
21	0.0	0.00	0.0	0.0	0.00

	vol_3g_mb_8	vol_3g_mb_9	monthly_2g_6	monthly_2g_7	monthly_2g_8
\					
8	11.94	0.0	0	1	0
13	0.00	0.0	0	0	0
16	0.00	0.0	0	0	0
17	0.00	0.0	0	0	0
21	0.00	0.0	0	0	0

	monthly_2g_9	sachet_2g_6	sachet_2g_7	sachet_2g_8
sachet_2g_9 \				
8	0	0	1	3
13	0	0	0	3
16	0	0	0	0
17	0	0	0	0
21	0	0	0	0

	monthly_3g_6	monthly_3g_7	monthly_3g_8	monthly_3g_9
sachet_3g_6 \				
8	0	0	0	0
0				
13	0	0	0	0
0				

16	0	0	0	0
0				
17	0	0	0	0
0				
21	0	0	0	0
0				

	sachet_3g_7	sachet_3g_8	sachet_3g_9	aon	aug_vbc_3g
jul_vbc_3g \					
8	0	0	0	315	21.03
910.65					
13	0	0	0	2607	0.00
0.00					
16	0	0	0	511	0.00
2.45					
17	0	0	0	667	0.00
0.00					
21	0	0	0	720	0.00
0.00					

	jun_vbc_3g	sep_vbc_3g	avg_rech_amt_6_7	churn
8	122.16	0.0	519.0	0
13	0.00	0.0	380.0	0
16	21.89	0.0	459.0	0
17	0.00	0.0	408.0	0
21	0.00	0.0	640.0	0

Deleting all the attributes corresponding to the churn phase

```
# List the columns for churn month(9)
col_9 = [col for col in df.columns.to_list() if '_9' in col]
print(col_9)

['arpu_9', 'onnet_mou_9', 'offnet_mou_9', 'roam_ic_mou_9',
'roam_og_mou_9', 'loc_og_t2t_mou_9', 'loc_og_t2m_mou_9',
'loc_og_t2f_mou_9', 'loc_og_t2c_mou_9', 'loc_og_mou_9',
'std_og_t2t_mou_9', 'std_og_t2m_mou_9', 'std_og_t2f_mou_9',
'std_og_t2c_mou_9', 'std_og_mou_9', 'isd_og_mou_9', 'spl_og_mou_9',
'og_others_9', 'total_og_mou_9', 'loc_ic_t2t_mou_9',
'loc_ic_t2m_mou_9', 'loc_ic_t2f_mou_9', 'loc_ic_mou_9',
'std_ic_t2t_mou_9', 'std_ic_t2m_mou_9', 'std_ic_t2f_mou_9',
'std_ic_t2o_mou_9', 'std_ic_mou_9', 'total_ic_mou_9', 'spl_ic_mou_9',
'isd_ic_mou_9', 'ic_others_9', 'total_rech_num_9', 'total_rech_amt_9',
'max_rech_amt_9', 'last_day_rch_amt_9', 'vol_2g_mb_9', 'vol_3g_mb_9',
'monthly_2g_9', 'sachet_2g_9', 'monthly_3g_9', 'sachet_3g_9']

# Deleting the churn month columns
df = df.drop(col_9, axis=1)
```



```
# Dropping sep_vbc_3g column
df = df.drop('sep_vbc_3g', axis=1)
```

Checking churn percentage

```
round(100*(df['churn'].mean()),2)

3.39
```

There is very little percentage of churn rate. We will take care of the class imbalance later.

Outliers treatment

In the filtered dataset except mobile_number and churn columns all the columns are numeric types. Hence, converting mobile_number and churn datatype to object.

```
df['mobile_number'] = df['mobile_number'].astype(object)
df['churn'] = df['churn'].astype(object)

df.info()

<class 'pandas.core.frame.DataFrame'>
Int64Index: 27991 entries, 8 to 99997
Columns: 136 entries, mobile_number to churn
dtypes: float64(109), int64(25), object(2)
memory usage: 29.3+ MB

# List only the numeric columns
numeric_cols = df.select_dtypes(exclude=['object']).columns
print(numeric_cols)

Index(['loc_og_t2o_mou', 'std_og_t2o_mou', 'loc_ic_t2o_mou', 'arpu_6',
      'arpu_7', 'arpu_8', 'onnet_mou_6', 'onnet_mou_7',
      'onnet_mou_8',
      'offnet_mou_6',
      ...,
      'monthly_3g_7', 'monthly_3g_8', 'sachet_3g_6', 'sachet_3g_7',
      'sachet_3g_8', 'aon', 'aug_vbc_3g', 'jul_vbc_3g', 'jun_vbc_3g',
      'avg_rech_amt_6_7'],
      dtype='object', length=134)

# Removing outliers below 10th and above 90th percentile
for col in numeric_cols:
    q1 = df[col].quantile(0.10)
    q3 = df[col].quantile(0.90)
    iqr = q3-q1
    range_low = q1-1.5*iqr
    range_high = q3+1.5*iqr
    # Assigning the filtered dataset into data
    data = df.loc[(df[col] > range_low) & (df[col] < range_high)]
```

```
data.shape  
(27705, 136)
```

Derive new features

```
# List the columns of total mou, rech_num and rech_amt  
[total for total in data.columns.to_list() if 'total' in total]  
  
['total_og_mou_6',  
 'total_og_mou_7',  
 'total_og_mou_8',  
 'total_ic_mou_6',  
 'total_ic_mou_7',  
 'total_ic_mou_8',  
 'total_rech_num_6',  
 'total_rech_num_7',  
 'total_rech_num_8',  
 'total_rech_amt_6',  
 'total_rech_amt_7',  
 'total_rech_amt_8']
```

Deriving new column decrease_mou_action

This column indicates whether the minutes of usage of the customer has decreased in the action phase than the good phase.

```
# Total mou at good phase incoming and outgoing  
data['total_mou_good'] = (data['total_og_mou_6'] +  
data['total_ic_mou_6'])  
  
# Avg. mou at action phase  
# We are taking average because there are two months(7 and 8) in  
# action phase  
data['avg_mou_action'] = (data['total_og_mou_7'] +  
data['total_og_mou_8'] + data['total_ic_mou_7'] +  
data['total_ic_mou_8'])/2  
  
# Difference avg_mou_good and avg_mou_action  
data['diff_mou'] = data['avg_mou_action'] - data['total_mou_good']  
  
# Checking whether the mou has decreased in action phase  
data['decrease_mou_action'] = np.where((data['diff_mou'] < 0), 1, 0)  
  
data.head()  
  
   mobile_number  loc_og_t2o_mou  std_og_t2o_mou  loc_ic_t2o_mou  
arpu_6 \  
8      7001524846             0.0             0.0             0.0  
378.721
```

13	7002191713	0.0	0.0	0.0
492.846				
16	7000875565	0.0	0.0	0.0
430.975				
17	7000187447	0.0	0.0	0.0
690.008				
21	7002124215	0.0	0.0	0.0
514.453				

	arpu_7	arpu_8	onnet_mou_6	onnet_mou_7	onnet_mou_8
offnet_mou_6	\				
8	492.223	137.362	413.69	351.03	35.08
94.66					
13	205.671	593.260	501.76	108.39	534.24
413.31					
16	299.869	187.894	50.51	74.01	70.61
296.29					
17	18.980	25.499	1185.91	9.28	7.79
61.64					
21	597.753	637.760	102.41	132.11	85.14
757.93					

	offnet_mou_7	offnet_mou_8	roam_ic_mou_6	roam_ic_mou_7
roam_ic_mou_8	\			
8	80.63	136.48	0.00	0.00
0.00				
13	119.28	482.46	23.53	144.24
72.11				
16	229.74	162.76	0.00	2.83
0.00				
17	0.00	5.54	0.00	4.76
4.81				
21	896.68	983.39	0.00	0.00
0.00				

	roam_og_mou_6	roam_og_mou_7	roam_og_mou_8	loc_og_t2t_mou_6	\
8	0.00	0.00	0.00	297.13	
13	7.98	35.26	1.44	49.63	
16	0.00	17.74	0.00	42.61	
17	0.00	8.46	13.34	38.99	
21	0.00	0.00	0.00	4.48	

	loc_og_t2t_mou_7	loc_og_t2t_mou_8	loc_og_t2m_mou_6
loc_og_t2m_mou_7	\		
8	217.59	12.49	80.96
70.58			
13	6.19	36.01	151.13
47.28			
16	65.16	67.38	273.29
145.99			

17	0.00	0.00	58.54
0.00			
21	6.16	23.34	91.81
87.93			
loc_og_t2m_mou_8	loc_og_t2f_mou_6	loc_og_t2f_mou_7	
loc_og_t2f_mou_8 \			
8	50.54	0.00	0.00
0.00			
13	294.46	4.54	0.00
23.51			
16	128.28	0.00	4.48
10.26			
17	0.00	0.00	0.00
0.00			
21	104.81	0.75	0.00
1.58			
loc_og_t2c_mou_6	loc_og_t2c_mou_7	loc_og_t2c_mou_8	loc_og_mou_6
\			
8	0.0	0.0	7.15 378.09
13	0.0	0.0	0.49 205.31
16	0.0	0.0	0.00 315.91
17	0.0	0.0	0.00 97.54
21	0.0	0.0	0.00 97.04
loc_og_mou_7	loc_og_mou_8	std_og_t2t_mou_6	std_og_t2t_mou_7 \
8	288.18	63.04	116.56 133.43
13	53.48	353.99	446.41 85.98
16	215.64	205.93	7.89 2.58
17	0.00	0.00	1146.91 0.81
21	94.09	129.74	97.93 125.94
std_og_t2t_mou_8	std_og_t2m_mou_6	std_og_t2m_mou_7	
std_og_t2m_mou_8 \			
8	22.58	13.69	10.04
75.69			
13	498.23	255.36	52.94
156.94			
16	3.23	22.99	64.51
18.29			
17	0.00	1.55	0.00
0.00			
21	61.79	665.36	808.74
876.99			

	std_og_t2f_mou_6	std_og_t2f_mou_7	std_og_t2f_mou_8
std_og_t2c_mou_6 \			
8	0.0	0.0	0.0
0.0			
13	0.0	0.0	0.0
0.0			
16	0.0	0.0	0.0
0.0			
17	0.0	0.0	0.0
0.0			
21	0.0	0.0	0.0
0.0			

	std_og_t2c_mou_7	std_og_t2c_mou_8	std_og_mou_6	std_og_mou_7 \
8	0.0	0.0	130.26	143.48
13	0.0	0.0	701.78	138.93
16	0.0	0.0	30.89	67.09
17	0.0	0.0	1148.46	0.81
21	0.0	0.0	763.29	934.69

	std_og_mou_8	isd_og_mou_6	isd_og_mou_7	isd_og_mou_8
spl_og_mou_6 \				
8	98.28	0.0	0.0	0.00
0.00				
13	655.18	0.0	0.0	1.29
0.00				
16	21.53	0.0	0.0	0.00
0.00				
17	0.00	0.0	0.0	0.00
2.58				
21	938.79	0.0	0.0	0.00
0.00				

	spl_og_mou_7	spl_og_mou_8	og_others_6	og_others_7	og_others_8
\					
8	0.00	10.23	0.00	0.0	0.0
13	0.00	4.78	0.00	0.0	0.0
16	3.26	5.91	0.00	0.0	0.0
17	0.00	0.00	0.93	0.0	0.0
21	0.00	0.00	0.00	0.0	0.0

	total_og_mou_6	total_og_mou_7	total_og_mou_8
loc_ic_t2t_mou_6 \			
8	508.36	431.66	171.56
			23.84

13	907.09	192.41	1015.26	67.88
16	346.81	286.01	233.38	41.33
17	1249.53	0.81	0.00	34.54
21	860.34	1028.79	1068.54	2.48

	loc_ic_t2t_mou_7	loc_ic_t2t_mou_8	loc_ic_t2m_mou_6	
loc_ic_t2m_mou_7 \				
8	9.84	0.31	57.58	
13.98				
13	7.58	52.58	142.88	
18.53				
16	71.44	28.89	226.81	
149.69				
17	0.00	0.00	47.41	
2.31				
21	10.19	19.54	118.23	
74.63				

	loc_ic_t2m_mou_8	loc_ic_t2f_mou_6	loc_ic_t2f_mou_7	
loc_ic_t2f_mou_8 \				
8	15.48	0.00	0.00	
0.00				
13	195.18	4.81	0.00	
7.49				
16	150.16	8.71	8.68	
32.71				
17	0.00	0.00	0.00	
0.00				
21	129.16	4.61	2.84	
10.39				

	loc_ic_mou_6	loc_ic_mou_7	loc_ic_mou_8	std_ic_t2t_mou_6 \
8	81.43	23.83	15.79	0.00
13	215.58	26.11	255.26	115.68
16	276.86	229.83	211.78	68.79
17	81.96	2.31	0.00	8.63
21	125.33	87.68	159.11	14.06

	std_ic_t2t_mou_7	std_ic_t2t_mou_8	std_ic_t2m_mou_6	
std_ic_t2m_mou_7 \				
8	0.58	0.10	22.43	
4.08				
13	38.29	154.58	308.13	
29.79				
16	78.64	6.33	18.68	

73.08				
17	0.00	0.00	1.28	
0.00				
21	5.98	0.18	67.69	
38.23				
std_ic_t2m_mou_8 std_ic_t2f_mou_6 std_ic_t2f_mou_7				
std_ic_t2f_mou_8 \				
8	0.65	0.00	0.0	
0.00				
13	317.91	0.00	0.0	
1.91				
16	73.93	0.51	0.0	
2.18				
17	0.00	0.00	0.0	
0.00				
21	101.74	0.00	0.0	
0.00				
std_ic_t2o_mou_6 std_ic_t2o_mou_7 std_ic_t2o_mou_8 std_ic_mou_6				
\				
8	0.0	0.0	0.0	22.43
13	0.0	0.0	0.0	423.81
16	0.0	0.0	0.0	87.99
17	0.0	0.0	0.0	9.91
21	0.0	0.0	0.0	81.76
std_ic_mou_7 std_ic_mou_8 total_ic_mou_6 total_ic_mou_7 \				
8	4.66	0.75	103.86	28.49
13	68.09	474.41	968.61	172.58
16	151.73	82.44	364.86	381.56
17	0.00	0.00	91.88	2.31
21	44.21	101.93	207.09	131.89
total_ic_mou_8 spl_ic_mou_6 spl_ic_mou_7 spl_ic_mou_8				
isd_ic_mou_6 \				
8	16.54	0.00	0.0	0.0
0.00				
13	1144.53	0.45	0.0	0.0
245.28				
16	294.46	0.00	0.0	0.0
0.00				
17	0.00	0.00	0.0	0.0
0.00				
21	261.04	0.00	0.0	0.0

0.00

	isd_ic_mou_7	isd_ic_mou_8	ic_others_6	ic_others_7	ic_others_8
\					
8	0.00	0.00	0.00	0.00	0.00
13	62.11	393.39	83.48	16.24	21.44
16	0.00	0.23	0.00	0.00	0.00
17	0.00	0.00	0.00	0.00	0.00
21	0.00	0.00	0.00	0.00	0.00

	total_rech_num_6	total_rech_num_7	total_rech_num_8
total_rech_amt_6 \			
8	19	21	14
437			
13	6	4	11
507			
16	10	6	2
570			
17	19	2	4
816			
21	22	26	27
600			

	total_rech_amt_7	total_rech_amt_8	max_rech_amt_6	max_rech_amt_7
\				
8	601	120	90	154
13	253	717	110	110
16	348	160	110	110
17	0	30	110	0
21	680	718	50	50

	max_rech_amt_8	last_day_rch_amt_6	last_day_rch_amt_7	\
8	30	50	0	
13	130	110	50	
16	130	100	100	
17	30	30	0	
21	50	30	20	

	last_day_rch_amt_8	vol_2g_mb_6	vol_2g_mb_7	vol_2g_mb_8
vol_3g_mb_6 \				
8	10	0.0	356.0	0.03

0.0					
13		0	0.0	0.0	0.02
0.0					
16		130	0.0	0.0	0.00
0.0					
17		0	0.0	0.0	0.00
0.0					
21		50	0.0	0.0	0.00
0.0					
	vol_3g_mb_7	vol_3g_mb_8	monthly_2g_6	monthly_2g_7	monthly_2g_8
\					
8	750.95	11.94	0	1	0
13	0.00	0.00	0	0	0
16	0.00	0.00	0	0	0
17	0.00	0.00	0	0	0
21	0.00	0.00	0	0	0
	sachet_2g_6	sachet_2g_7	sachet_2g_8	monthly_3g_6	monthly_3g_7
\					
8	0	1	3	0	0
13	0	0	3	0	0
16	0	0	0	0	0
17	0	0	0	0	0
21	0	0	0	0	0
	monthly_3g_8	sachet_3g_6	sachet_3g_7	sachet_3g_8	aon
aug_vbc_3g	\				
8	0	0	0	0	315
21.03					
13	0	0	0	0	2607
0.00					
16	0	0	0	0	511
0.00					
17	0	0	0	0	667
0.00					
21	0	0	0	0	720
0.00					
	jul_vbc_3g	jun_vbc_3g	avg_rech_amt_6_7	churn	total_mou_good
8	910.65	122.16	519.0	0	612.22
					\

13	0.00	0.00	380.0	0	1875.70
16	2.45	21.89	459.0	0	711.67
17	0.00	0.00	408.0	0	1341.41
21	0.00	0.00	640.0	0	1067.43

	avg_mou_action	diff_mou	decrease_mou_action
8	324.125	-288.095	1
13	1262.390	-613.310	1
16	597.705	-113.965	1
17	1.560	-1339.850	1
21	1245.130	177.700	0

Deriving new column `decrease_rech_num_action`

This column indicates whether the number of recharge of the customer has decreased in the action phase than the good phase.

```
# Avg rech number at action phase
data['avg_rech_num_action'] = (data['total_rech_num_7'] +
data['total_rech_num_8'])/2

# Difference total_rech_num_6 and avg_rech_action
data['diff_rech_num'] = data['avg_rech_num_action'] -
data['total_rech_num_6']

# Checking if rech_num has decreased in action phase
data['decrease_rech_num_action'] = np.where((data['diff_rech_num'] <
0), 1, 0)
```

```
data.head()
```

	mobile_number	loc_og_t2o_mou	std_og_t2o_mou	loc_ic_t2o_mou
arpu_6 \				
8	7001524846	0.0	0.0	0.0
378.721				
13	7002191713	0.0	0.0	0.0
492.846				
16	7000875565	0.0	0.0	0.0
430.975				
17	7000187447	0.0	0.0	0.0
690.008				
21	7002124215	0.0	0.0	0.0
514.453				

	arpu_7	arpu_8	onnet_mou_6	onnet_mou_7	onnet_mou_8
offnet_mou_6 \					
8	492.223	137.362	413.69	351.03	35.08
94.66					
13	205.671	593.260	501.76	108.39	534.24
413.31					

16	299.869	187.894	50.51	74.01	70.61
296.29					
17	18.980	25.499	1185.91	9.28	7.79
61.64					
21	597.753	637.760	102.41	132.11	85.14
757.93					

	offnet_mou_7	offnet_mou_8	roam_ic_mou_6	roam_ic_mou_7
roam_ic_mou_8 \				
8	80.63	136.48	0.00	0.00
0.00				
13	119.28	482.46	23.53	144.24
72.11				
16	229.74	162.76	0.00	2.83
0.00				
17	0.00	5.54	0.00	4.76
4.81				
21	896.68	983.39	0.00	0.00
0.00				

	roam_og_mou_6	roam_og_mou_7	roam_og_mou_8	loc_og_t2t_mou_6 \
8	0.00	0.00	0.00	297.13
13	7.98	35.26	1.44	49.63
16	0.00	17.74	0.00	42.61
17	0.00	8.46	13.34	38.99
21	0.00	0.00	0.00	4.48

	loc_og_t2t_mou_7	loc_og_t2t_mou_8	loc_og_t2m_mou_6
loc_og_t2m_mou_7 \			
8	217.59	12.49	80.96
70.58			
13	6.19	36.01	151.13
47.28			
16	65.16	67.38	273.29
145.99			
17	0.00	0.00	58.54
0.00			
21	6.16	23.34	91.81
87.93			

	loc_og_t2m_mou_8	loc_og_t2f_mou_6	loc_og_t2f_mou_7
loc_og_t2f_mou_8 \			
8	50.54	0.00	0.00
0.00			
13	294.46	4.54	0.00
23.51			
16	128.28	0.00	4.48
10.26			
17	0.00	0.00	0.00
0.00			

21	104.81	0.75	0.00	
1.58				
	loc_og_t2c_mou_6	loc_og_t2c_mou_7	loc_og_t2c_mou_8	loc_og_mou_6
\				
8	0.0	0.0	7.15	378.09
13	0.0	0.0	0.49	205.31
16	0.0	0.0	0.00	315.91
17	0.0	0.0	0.00	97.54
21	0.0	0.0	0.00	97.04
	loc_og_mou_7	loc_og_mou_8	std_og_t2t_mou_6	std_og_t2t_mou_7 \
8	288.18	63.04	116.56	133.43
13	53.48	353.99	446.41	85.98
16	215.64	205.93	7.89	2.58
17	0.00	0.00	1146.91	0.81
21	94.09	129.74	97.93	125.94
	std_og_t2t_mou_8	std_og_t2m_mou_6	std_og_t2m_mou_7	
std_og_t2m_mou_8 \				
8	22.58	13.69	10.04	
75.69				
13	498.23	255.36	52.94	
156.94				
16	3.23	22.99	64.51	
18.29				
17	0.00	1.55	0.00	
0.00				
21	61.79	665.36	808.74	
876.99				
	std_og_t2f_mou_6	std_og_t2f_mou_7	std_og_t2f_mou_8	
std_og_t2c_mou_6 \				
8	0.0	0.0	0.0	
0.0				
13	0.0	0.0	0.0	
0.0				
16	0.0	0.0	0.0	
0.0				
17	0.0	0.0	0.0	
0.0				
21	0.0	0.0	0.0	
0.0				
	std_og_t2c_mou_7	std_og_t2c_mou_8	std_og_mou_6	std_og_mou_7 \

8	0.0	0.0	130.26	143.48
13	0.0	0.0	701.78	138.93
16	0.0	0.0	30.89	67.09
17	0.0	0.0	1148.46	0.81
21	0.0	0.0	763.29	934.69

	std_og_mou_8	isd_og_mou_6	isd_og_mou_7	isd_og_mou_8
spl_og_mou_6 \				
8	98.28	0.0	0.0	0.00
0.00				
13	655.18	0.0	0.0	1.29
0.00				
16	21.53	0.0	0.0	0.00
0.00				
17	0.00	0.0	0.0	0.00
2.58				
21	938.79	0.0	0.0	0.00
0.00				

	spl_og_mou_7	spl_og_mou_8	og_others_6	og_others_7	og_others_8
\					
8	0.00	10.23	0.00	0.0	0.0
13	0.00	4.78	0.00	0.0	0.0
16	3.26	5.91	0.00	0.0	0.0
17	0.00	0.00	0.93	0.0	0.0
21	0.00	0.00	0.00	0.0	0.0

	total_og_mou_6	total_og_mou_7	total_og_mou_8	
loc_ic_t2t_mou_6 \				
8	508.36	431.66	171.56	23.84
13	907.09	192.41	1015.26	67.88
16	346.81	286.01	233.38	41.33
17	1249.53	0.81	0.00	34.54
21	860.34	1028.79	1068.54	2.48

	loc_ic_t2t_mou_7	loc_ic_t2t_mou_8	loc_ic_t2m_mou_6
loc_ic_t2m_mou_7 \			
8	9.84	0.31	57.58
13.98			
13	7.58	52.58	142.88
18.53			

16	71.44	28.89	226.81
149.69			
17	0.00	0.00	47.41
2.31			
21	10.19	19.54	118.23
74.63			

	loc_ic_t2m_mou_8	loc_ic_t2f_mou_6	loc_ic_t2f_mou_7
loc_ic_t2f_mou_8 \			
8	15.48	0.00	0.00
0.00			
13	195.18	4.81	0.00
7.49			
16	150.16	8.71	8.68
32.71			
17	0.00	0.00	0.00
0.00			
21	129.16	4.61	2.84
10.39			

	loc_ic_mou_6	loc_ic_mou_7	loc_ic_mou_8	std_ic_t2t_mou_6 \
8	81.43	23.83	15.79	0.00
13	215.58	26.11	255.26	115.68
16	276.86	229.83	211.78	68.79
17	81.96	2.31	0.00	8.63
21	125.33	87.68	159.11	14.06

	std_ic_t2t_mou_7	std_ic_t2t_mou_8	std_ic_t2m_mou_6
std_ic_t2m_mou_7 \			
8	0.58	0.10	22.43
4.08			
13	38.29	154.58	308.13
29.79			
16	78.64	6.33	18.68
73.08			
17	0.00	0.00	1.28
0.00			
21	5.98	0.18	67.69
38.23			

	std_ic_t2m_mou_8	std_ic_t2f_mou_6	std_ic_t2f_mou_7
std_ic_t2f_mou_8 \			
8	0.65	0.00	0.0
0.00			
13	317.91	0.00	0.0
1.91			
16	73.93	0.51	0.0
2.18			
17	0.00	0.00	0.0
0.00			

21	101.74	0.00	0.0		
0.00					
	std_ic_t2o_mou_6	std_ic_t2o_mou_7	std_ic_t2o_mou_8	std_ic_mou_6	
\					
8	0.0	0.0	0.0	22.43	
13	0.0	0.0	0.0	423.81	
16	0.0	0.0	0.0	87.99	
17	0.0	0.0	0.0	9.91	
21	0.0	0.0	0.0	81.76	
	std_ic_mou_7	std_ic_mou_8	total_ic_mou_6	total_ic_mou_7	\
8	4.66	0.75	103.86	28.49	
13	68.09	474.41	968.61	172.58	
16	151.73	82.44	364.86	381.56	
17	0.00	0.00	91.88	2.31	
21	44.21	101.93	207.09	131.89	
	total_ic_mou_8	spl_ic_mou_6	spl_ic_mou_7	spl_ic_mou_8	
isd_ic_mou_6	\				
8	16.54	0.00	0.0	0.0	
0.00					
13	1144.53	0.45	0.0	0.0	
245.28					
16	294.46	0.00	0.0	0.0	
0.00					
17	0.00	0.00	0.0	0.0	
0.00					
21	261.04	0.00	0.0	0.0	
0.00					
	isd_ic_mou_7	isd_ic_mou_8	ic_others_6	ic_others_7	ic_others_8
\					
8	0.00	0.00	0.00	0.00	0.00
13	62.11	393.39	83.48	16.24	21.44
16	0.00	0.23	0.00	0.00	0.00
17	0.00	0.00	0.00	0.00	0.00
21	0.00	0.00	0.00	0.00	0.00
	total_rech_num_6	total_rech_num_7	total_rech_num_8		
total_rech_amt_6	\				

8	19	21	14
437			
13	6	4	11
507			
16	10	6	2
570			
17	19	2	4
816			
21	22	26	27
600			

	total_rech_amt_7	total_rech_amt_8	max_rech_amt_6	max_rech_amt_7
\				
8	601	120	90	154
13	253	717	110	110
16	348	160	110	110
17	0	30	110	0
21	680	718	50	50

	max_rech_amt_8	last_day_rch_amt_6	last_day_rch_amt_7	\
8	30	50	0	
13	130	110	50	
16	130	100	100	
17	30	30	0	
21	50	30	20	

	last_day_rch_amt_8	vol_2g_mb_6	vol_2g_mb_7	vol_2g_mb_8
vol_3g_mb_6 \				
8	10	0.0	356.0	0.03
0.0				
13	0	0.0	0.0	0.02
0.0				
16	130	0.0	0.0	0.00
0.0				
17	0	0.0	0.0	0.00
0.0				
21	50	0.0	0.0	0.00
0.0				

	vol_3g_mb_7	vol_3g_mb_8	monthly_2g_6	monthly_2g_7	monthly_2g_8
\					
8	750.95	11.94	0	1	0
13	0.00	0.00	0	0	0

16	0.00	0.00	0	0	0
17	0.00	0.00	0	0	0
21	0.00	0.00	0	0	0
	sachet_2g_6	sachet_2g_7	sachet_2g_8	monthly_3g_6	monthly_3g_7
\					
8	0	1	3	0	0
13	0	0	3	0	0
16	0	0	0	0	0
17	0	0	0	0	0
21	0	0	0	0	0
	monthly_3g_8	sachet_3g_6	sachet_3g_7	sachet_3g_8	aon
aug_vbc_3g	\				
8	0	0	0	0	315
21.03					
13	0	0	0	0	2607
0.00					
16	0	0	0	0	511
0.00					
17	0	0	0	0	667
0.00					
21	0	0	0	0	720
0.00					
	jul_vbc_3g	jun_vbc_3g	avg_rech_amt_6_7	churn	total_mou_good
\					
8	910.65	122.16	519.0	0	612.22
13	0.00	0.00	380.0	0	1875.70
16	2.45	21.89	459.0	0	711.67
17	0.00	0.00	408.0	0	1341.41
21	0.00	0.00	640.0	0	1067.43
	avg_mou_action	diff_mou	decrease_mou_action	avg_rech_num_action	
\					
8	324.125	-288.095		1	17.5
13	1262.390	-613.310		1	7.5
16	597.705	-113.965		1	4.0
17	1.560	-1339.850		1	3.0
21	1245.130	177.700		0	26.5

	diff_rech_num	decrease_rech_num_action
8	-1.5	1
13	1.5	0
16	-6.0	1
17	-16.0	1
21	4.5	0

Deriving new column decrease_rech_amt_action

This column indicates whether the amount of recharge of the customer has decreased in the action phase than the good phase.

```
# Avg rech_amt in action phase
data['avg_rech_amt_action'] = (data['total_rech_amt_7'] +
data['total_rech_amt_8'])/2

# Difference of action phase rech amt and good phase rech amt
data['diff_rech_amt'] = data['avg_rech_amt_action'] -
data['total_rech_amt_6']

# Checking if rech_amt has decreased in action phase
data['decrease_rech_amt_action'] = np.where((data['diff_rech_amt'] <
0), 1, 0)
```

```
data.head()
```

	mobile_number	loc_og_t2o_mou	std_og_t2o_mou	loc_ic_t2o_mou
arpu_6 \				
8	7001524846	0.0	0.0	0.0
378.721				
13	7002191713	0.0	0.0	0.0
492.846				
16	7000875565	0.0	0.0	0.0
430.975				
17	7000187447	0.0	0.0	0.0
690.008				
21	7002124215	0.0	0.0	0.0
514.453				

	arpu_7	arpu_8	onnet_mou_6	onnet_mou_7	onnet_mou_8
offnet_mou_6 \					
8	492.223	137.362	413.69	351.03	35.08
94.66					
13	205.671	593.260	501.76	108.39	534.24
413.31					
16	299.869	187.894	50.51	74.01	70.61
296.29					
17	18.980	25.499	1185.91	9.28	7.79

61.64
 21 597.753 637.760 102.41 132.11 85.14
 757.93

	offnet_mou_7	offnet_mou_8	roam_ic_mou_6	roam_ic_mou_7
roam_ic_mou_8 \				
8	80.63	136.48	0.00	0.00
0.00				
13	119.28	482.46	23.53	144.24
72.11				
16	229.74	162.76	0.00	2.83
0.00				
17	0.00	5.54	0.00	4.76
4.81				
21	896.68	983.39	0.00	0.00
0.00				

	roam_og_mou_6	roam_og_mou_7	roam_og_mou_8	loc_og_t2t_mou_6 \
8	0.00	0.00	0.00	297.13
13	7.98	35.26	1.44	49.63
16	0.00	17.74	0.00	42.61
17	0.00	8.46	13.34	38.99
21	0.00	0.00	0.00	4.48

	loc_og_t2t_mou_7	loc_og_t2t_mou_8	loc_og_t2m_mou_6
loc_og_t2m_mou_7 \			
8	217.59	12.49	80.96
70.58			
13	6.19	36.01	151.13
47.28			
16	65.16	67.38	273.29
145.99			
17	0.00	0.00	58.54
0.00			
21	6.16	23.34	91.81
87.93			

	loc_og_t2m_mou_8	loc_og_t2f_mou_6	loc_og_t2f_mou_7
loc_og_t2f_mou_8 \			
8	50.54	0.00	0.00
0.00			
13	294.46	4.54	0.00
23.51			
16	128.28	0.00	4.48
10.26			
17	0.00	0.00	0.00
0.00			
21	104.81	0.75	0.00
1.58			

	loc_og_t2c_mou_6	loc_og_t2c_mou_7	loc_og_t2c_mou_8	loc_og_mou_6	
\					
8	0.0	0.0	7.15	378.09	
13	0.0	0.0	0.49	205.31	
16	0.0	0.0	0.00	315.91	
17	0.0	0.0	0.00	97.54	
21	0.0	0.0	0.00	97.04	
	loc_og_mou_7	loc_og_mou_8	std_og_t2t_mou_6	std_og_t2t_mou_7	\
8	288.18	63.04	116.56	133.43	
13	53.48	353.99	446.41	85.98	
16	215.64	205.93	7.89	2.58	
17	0.00	0.00	1146.91	0.81	
21	94.09	129.74	97.93	125.94	
	std_og_t2t_mou_8	std_og_t2m_mou_6	std_og_t2m_mou_7		
std_og_t2m_mou_8	\				
8	22.58	13.69	10.04		
75.69					
13	498.23	255.36	52.94		
156.94					
16	3.23	22.99	64.51		
18.29					
17	0.00	1.55	0.00		
0.00					
21	61.79	665.36	808.74		
876.99					
	std_og_t2f_mou_6	std_og_t2f_mou_7	std_og_t2f_mou_8		
std_og_t2c_mou_6	\				
8	0.0	0.0	0.0		
0.0					
13	0.0	0.0	0.0		
0.0					
16	0.0	0.0	0.0		
0.0					
17	0.0	0.0	0.0		
0.0					
21	0.0	0.0	0.0		
0.0					
	std_og_t2c_mou_7	std_og_t2c_mou_8	std_og_mou_6	std_og_mou_7	\
8	0.0	0.0	130.26	143.48	
13	0.0	0.0	701.78	138.93	
16	0.0	0.0	30.89	67.09	

17	0.0	0.0	1148.46	0.81	
21	0.0	0.0	763.29	934.69	
std_og_mou_8	isd_og_mou_6	isd_og_mou_7	isd_og_mou_8		
spl_og_mou_6 \					
8	98.28	0.0	0.0	0.00	
0.00					
13	655.18	0.0	0.0	1.29	
0.00					
16	21.53	0.0	0.0	0.00	
0.00					
17	0.00	0.0	0.0	0.00	
2.58					
21	938.79	0.0	0.0	0.00	
0.00					
spl_og_mou_7	spl_og_mou_8	og_others_6	og_others_7	og_others_8	
\					
8	0.00	10.23	0.00	0.0	0.0
13	0.00	4.78	0.00	0.0	0.0
16	3.26	5.91	0.00	0.0	0.0
17	0.00	0.00	0.93	0.0	0.0
21	0.00	0.00	0.00	0.0	0.0
total_og_mou_6	total_og_mou_7	total_og_mou_8			
loc_ic_t2t_mou_6 \					
8	508.36	431.66	171.56		23.84
13	907.09	192.41	1015.26		67.88
16	346.81	286.01	233.38		41.33
17	1249.53	0.81	0.00		34.54
21	860.34	1028.79	1068.54		2.48
loc_ic_t2t_mou_7	loc_ic_t2t_mou_8	loc_ic_t2m_mou_6			
loc_ic_t2m_mou_7 \					
8	9.84	0.31	57.58		
13.98					
13	7.58	52.58	142.88		
18.53					
16	71.44	28.89	226.81		
149.69					
17	0.00	0.00	47.41		

2.31			
21	10.19	19.54	118.23
74.63			

	loc_ic_t2m_mou_8	loc_ic_t2f_mou_6	loc_ic_t2f_mou_7
loc_ic_t2f_mou_8 \			
8	15.48	0.00	0.00
0.00			
13	195.18	4.81	0.00
7.49			
16	150.16	8.71	8.68
32.71			
17	0.00	0.00	0.00
0.00			
21	129.16	4.61	2.84
10.39			

	loc_ic_mou_6	loc_ic_mou_7	loc_ic_mou_8	std_ic_t2t_mou_6 \
8	81.43	23.83	15.79	0.00
13	215.58	26.11	255.26	115.68
16	276.86	229.83	211.78	68.79
17	81.96	2.31	0.00	8.63
21	125.33	87.68	159.11	14.06

	std_ic_t2t_mou_7	std_ic_t2t_mou_8	std_ic_t2m_mou_6
std_ic_t2m_mou_7 \			
8	0.58	0.10	22.43
4.08			
13	38.29	154.58	308.13
29.79			
16	78.64	6.33	18.68
73.08			
17	0.00	0.00	1.28
0.00			
21	5.98	0.18	67.69
38.23			

	std_ic_t2m_mou_8	std_ic_t2f_mou_6	std_ic_t2f_mou_7
std_ic_t2f_mou_8 \			
8	0.65	0.00	0.0
0.00			
13	317.91	0.00	0.0
1.91			
16	73.93	0.51	0.0
2.18			
17	0.00	0.00	0.0
0.00			
21	101.74	0.00	0.0
0.00			

	std_ic_t2o_mou_6	std_ic_t2o_mou_7	std_ic_t2o_mou_8	std_ic_mou_6	
\					
8	0.0	0.0	0.0	22.43	
13	0.0	0.0	0.0	423.81	
16	0.0	0.0	0.0	87.99	
17	0.0	0.0	0.0	9.91	
21	0.0	0.0	0.0	81.76	
	std_ic_mou_7	std_ic_mou_8	total_ic_mou_6	total_ic_mou_7	\
8	4.66	0.75	103.86	28.49	
13	68.09	474.41	968.61	172.58	
16	151.73	82.44	364.86	381.56	
17	0.00	0.00	91.88	2.31	
21	44.21	101.93	207.09	131.89	
	total_ic_mou_8	spl_ic_mou_6	spl_ic_mou_7	spl_ic_mou_8	isd_ic_mou_6 \
8	16.54	0.00	0.0	0.0	0.00
13	1144.53	0.45	0.0	0.0	245.28
16	294.46	0.00	0.0	0.0	0.00
17	0.00	0.00	0.0	0.0	0.00
21	261.04	0.00	0.0	0.0	0.00
	isd_ic_mou_7	isd_ic_mou_8	ic_others_6	ic_others_7	ic_others_8
\					
8	0.00	0.00	0.00	0.00	0.00
13	62.11	393.39	83.48	16.24	21.44
16	0.00	0.23	0.00	0.00	0.00
17	0.00	0.00	0.00	0.00	0.00
21	0.00	0.00	0.00	0.00	0.00
	total_rech_num_6	total_rech_num_7	total_rech_num_8	total_rech_amt_6 \	
8	19	21	14	437	
13	6	4	11		

507					
16	10	6	2		
570					
17	19	2	4		
816					
21	22	26	27		
600					
	total_rech_amt_7	total_rech_amt_8	max_rech_amt_6	max_rech_amt_7	
\					
8	601	120	90	154	
13	253	717	110	110	
16	348	160	110	110	
17	0	30	110	0	
21	680	718	50	50	
	max_rech_amt_8	last_day_rch_amt_6	last_day_rch_amt_7	\	
8	30	50	0		
13	130	110	50		
16	130	100	100		
17	30	30	0		
21	50	30	20		
	last_day_rch_amt_8	vol_2g_mb_6	vol_2g_mb_7	vol_2g_mb_8	
vol_3g_mb_6	\				
8	10	0.0	356.0	0.03	
0.0					
13	0	0.0	0.0	0.02	
0.0					
16	130	0.0	0.0	0.00	
0.0					
17	0	0.0	0.0	0.00	
0.0					
21	50	0.0	0.0	0.00	
0.0					
	vol_3g_mb_7	vol_3g_mb_8	monthly_2g_6	monthly_2g_7	monthly_2g_8
\					
8	750.95	11.94	0	1	0
13	0.00	0.00	0	0	0
16	0.00	0.00	0	0	0
17	0.00	0.00	0	0	0

21	0.00	0.00	0	0	0
sachet_2g_6	sachet_2g_7	sachet_2g_8	monthly_3g_6	monthly_3g_7	
\					
8	0	1	3	0	0
13	0	0	3	0	0
16	0	0	0	0	0
17	0	0	0	0	0
21	0	0	0	0	0
monthly_3g_8	sachet_3g_6	sachet_3g_7	sachet_3g_8	aon	
aug_vbc_3g	\				
8	0	0	0	0	315
21.03					
13	0	0	0	0	2607
0.00					
16	0	0	0	0	511
0.00					
17	0	0	0	0	667
0.00					
21	0	0	0	0	720
0.00					
jul_vbc_3g	jun_vbc_3g	avg_rech_amt_6_7	churn	total_mou_good	\
8	910.65	122.16	519.0	0	612.22
13	0.00	0.00	380.0	0	1875.70
16	2.45	21.89	459.0	0	711.67
17	0.00	0.00	408.0	0	1341.41
21	0.00	0.00	640.0	0	1067.43
avg_mou_action	diff_mou	decrease_mou_action	avg_rech_num_action		
\					
8	324.125	-288.095	1		17.5
13	1262.390	-613.310	1		7.5
16	597.705	-113.965	1		4.0
17	1.560	-1339.850	1		3.0
21	1245.130	177.700	0		26.5
diff_rech_num	decrease_rech_num_action	avg_rech_amt_action	\		

8	-1.5	1	360.5
13	1.5	0	485.0
16	-6.0	1	254.0
17	-16.0	1	15.0
21	4.5	0	699.0

	diff_rech_amt	decrease_rech_amt_action
8	-76.5	1
13	-22.0	1
16	-316.0	1
17	-801.0	1
21	99.0	0

Deriving new column `decrease_arpu_action`

This column indicates whether the average revenue per customer has decreased in the action phase than the good phase.

```
# ARUP in action phase
data['avg_arpu_action'] = (data['arpu_7'] + data['arpu_8'])/2

# Difference of good and action phase ARPU
data['diff_arpu'] = data['avg_arpu_action'] - data['arpu_6']

# Checking whether the arpu has decreased on the action month
data['decrease_arpu_action'] = np.where(data['diff_arpu'] < 0, 1, 0)

data.head()
```

	mobile_number	loc_og_t2o_mou	std_og_t2o_mou	loc_ic_t2o_mou
arpu_6 \				
8	7001524846	0.0	0.0	0.0
378.721				
13	7002191713	0.0	0.0	0.0
492.846				
16	7000875565	0.0	0.0	0.0
430.975				
17	7000187447	0.0	0.0	0.0
690.008				
21	7002124215	0.0	0.0	0.0
514.453				

	arpu_7	arpu_8	onnet_mou_6	onnet_mou_7	onnet_mou_8
offnet_mou_6 \					
8	492.223	137.362	413.69	351.03	35.08
94.66					
13	205.671	593.260	501.76	108.39	534.24
413.31					
16	299.869	187.894	50.51	74.01	70.61
296.29					

17	18.980	25.499	1185.91	9.28	7.79
61.64					
21	597.753	637.760	102.41	132.11	85.14
757.93					

	offnet_mou_7	offnet_mou_8	roam_ic_mou_6	roam_ic_mou_7
roam_ic_mou_8 \				
8	80.63	136.48	0.00	0.00
0.00				
13	119.28	482.46	23.53	144.24
72.11				
16	229.74	162.76	0.00	2.83
0.00				
17	0.00	5.54	0.00	4.76
4.81				
21	896.68	983.39	0.00	0.00
0.00				

	roam_og_mou_6	roam_og_mou_7	roam_og_mou_8	loc_og_t2t_mou_6 \
8	0.00	0.00	0.00	297.13
13	7.98	35.26	1.44	49.63
16	0.00	17.74	0.00	42.61
17	0.00	8.46	13.34	38.99
21	0.00	0.00	0.00	4.48

	loc_og_t2t_mou_7	loc_og_t2t_mou_8	loc_og_t2m_mou_6
loc_og_t2m_mou_7 \			
8	217.59	12.49	80.96
70.58			
13	6.19	36.01	151.13
47.28			
16	65.16	67.38	273.29
145.99			
17	0.00	0.00	58.54
0.00			
21	6.16	23.34	91.81
87.93			

	loc_og_t2m_mou_8	loc_og_t2f_mou_6	loc_og_t2f_mou_7
loc_og_t2f_mou_8 \			
8	50.54	0.00	0.00
0.00			
13	294.46	4.54	0.00
23.51			
16	128.28	0.00	4.48
10.26			
17	0.00	0.00	0.00
0.00			
21	104.81	0.75	0.00
1.58			

	loc_og_t2c_mou_6	loc_og_t2c_mou_7	loc_og_t2c_mou_8	loc_og_mou_6
\				
8	0.0	0.0	7.15	378.09
13	0.0	0.0	0.49	205.31
16	0.0	0.0	0.00	315.91
17	0.0	0.0	0.00	97.54
21	0.0	0.0	0.00	97.04

	loc_og_mou_7	loc_og_mou_8	std_og_t2t_mou_6	std_og_t2t_mou_7	\
8	288.18	63.04	116.56	133.43	
13	53.48	353.99	446.41	85.98	
16	215.64	205.93	7.89	2.58	
17	0.00	0.00	1146.91	0.81	
21	94.09	129.74	97.93	125.94	

	std_og_t2t_mou_8	std_og_t2m_mou_6	std_og_t2m_mou_7
std_og_t2m_mou_8	\		
8	22.58	13.69	10.04
75.69			
13	498.23	255.36	52.94
156.94			
16	3.23	22.99	64.51
18.29			
17	0.00	1.55	0.00
0.00			
21	61.79	665.36	808.74
876.99			

	std_og_t2f_mou_6	std_og_t2f_mou_7	std_og_t2f_mou_8
std_og_t2c_mou_6	\		
8	0.0	0.0	0.0
0.0			
13	0.0	0.0	0.0
0.0			
16	0.0	0.0	0.0
0.0			
17	0.0	0.0	0.0
0.0			
21	0.0	0.0	0.0
0.0			

	std_og_t2c_mou_7	std_og_t2c_mou_8	std_og_mou_6	std_og_mou_7	\
8	0.0	0.0	130.26	143.48	
13	0.0	0.0	701.78	138.93	

16	0.0	0.0	30.89	67.09	
17	0.0	0.0	1148.46	0.81	
21	0.0	0.0	763.29	934.69	
std_og_mou_8	isd_og_mou_6	isd_og_mou_7	isd_og_mou_8		
spl_og_mou_6 \					
8	98.28	0.0	0.0	0.00	
0.00					
13	655.18	0.0	0.0	1.29	
0.00					
16	21.53	0.0	0.0	0.00	
0.00					
17	0.00	0.0	0.0	0.00	
2.58					
21	938.79	0.0	0.0	0.00	
0.00					
spl_og_mou_7	spl_og_mou_8	og_others_6	og_others_7	og_others_8	
\					
8	0.00	10.23	0.00	0.0	0.0
13	0.00	4.78	0.00	0.0	0.0
16	3.26	5.91	0.00	0.0	0.0
17	0.00	0.00	0.93	0.0	0.0
21	0.00	0.00	0.00	0.0	0.0
total_og_mou_6	total_og_mou_7	total_og_mou_8			
loc_ic_t2t_mou_6 \					
8	508.36	431.66	171.56	23.84	
13	907.09	192.41	1015.26	67.88	
16	346.81	286.01	233.38	41.33	
17	1249.53	0.81	0.00	34.54	
21	860.34	1028.79	1068.54	2.48	
loc_ic_t2t_mou_7	loc_ic_t2t_mou_8	loc_ic_t2m_mou_6			
loc_ic_t2m_mou_7 \					
8	9.84	0.31	57.58		
13.98					
13	7.58	52.58	142.88		
18.53					
16	71.44	28.89	226.81		
149.69					

17	0.00	0.00	47.41
2.31			
21	10.19	19.54	118.23
74.63			

	loc_ic_t2m_mou_8	loc_ic_t2f_mou_6	loc_ic_t2f_mou_7
loc_ic_t2f_mou_8 \			
8	15.48	0.00	0.00
0.00			
13	195.18	4.81	0.00
7.49			
16	150.16	8.71	8.68
32.71			
17	0.00	0.00	0.00
0.00			
21	129.16	4.61	2.84
10.39			

	loc_ic_mou_6	loc_ic_mou_7	loc_ic_mou_8	std_ic_t2t_mou_6 \
8	81.43	23.83	15.79	0.00
13	215.58	26.11	255.26	115.68
16	276.86	229.83	211.78	68.79
17	81.96	2.31	0.00	8.63
21	125.33	87.68	159.11	14.06

	std_ic_t2t_mou_7	std_ic_t2t_mou_8	std_ic_t2m_mou_6
std_ic_t2m_mou_7 \			
8	0.58	0.10	22.43
4.08			
13	38.29	154.58	308.13
29.79			
16	78.64	6.33	18.68
73.08			
17	0.00	0.00	1.28
0.00			
21	5.98	0.18	67.69
38.23			

	std_ic_t2m_mou_8	std_ic_t2f_mou_6	std_ic_t2f_mou_7
std_ic_t2f_mou_8 \			
8	0.65	0.00	0.0
0.00			
13	317.91	0.00	0.0
1.91			
16	73.93	0.51	0.0
2.18			
17	0.00	0.00	0.0
0.00			
21	101.74	0.00	0.0
0.00			

	std_ic_t2o_mou_6	std_ic_t2o_mou_7	std_ic_t2o_mou_8	std_ic_mou_6	
\					
8	0.0	0.0	0.0	22.43	
13	0.0	0.0	0.0	423.81	
16	0.0	0.0	0.0	87.99	
17	0.0	0.0	0.0	9.91	
21	0.0	0.0	0.0	81.76	
	std_ic_mou_7	std_ic_mou_8	total_ic_mou_6	total_ic_mou_7	\
8	4.66	0.75	103.86	28.49	
13	68.09	474.41	968.61	172.58	
16	151.73	82.44	364.86	381.56	
17	0.00	0.00	91.88	2.31	
21	44.21	101.93	207.09	131.89	
	total_ic_mou_8	spl_ic_mou_6	spl_ic_mou_7	spl_ic_mou_8	isd_ic_mou_6 \
8	16.54	0.00	0.0	0.0	0.00
13	1144.53	0.45	0.0	0.0	245.28
16	294.46	0.00	0.0	0.0	0.00
17	0.00	0.00	0.0	0.0	0.00
21	261.04	0.00	0.0	0.0	0.00
	isd_ic_mou_7	isd_ic_mou_8	ic_others_6	ic_others_7	ic_others_8
\					
8	0.00	0.00	0.00	0.00	0.00
13	62.11	393.39	83.48	16.24	21.44
16	0.00	0.23	0.00	0.00	0.00
17	0.00	0.00	0.00	0.00	0.00
21	0.00	0.00	0.00	0.00	0.00
	total_rech_num_6	total_rech_num_7	total_rech_num_8	total_rech_amt_6	\
8	19	21	14	437	

13	6	4	11
507			
16	10	6	2
570			
17	19	2	4
816			
21	22	26	27
600			

	total_rech_amt_7	total_rech_amt_8	max_rech_amt_6	max_rech_amt_7
\				
8	601	120	90	154
13	253	717	110	110
16	348	160	110	110
17	0	30	110	0
21	680	718	50	50

	max_rech_amt_8	last_day_rch_amt_6	last_day_rch_amt_7	\
8	30	50	0	
13	130	110	50	
16	130	100	100	
17	30	30	0	
21	50	30	20	

	last_day_rch_amt_8	vol_2g_mb_6	vol_2g_mb_7	vol_2g_mb_8
vol_3g_mb_6 \				
8	10	0.0	356.0	0.03
0.0				
13	0	0.0	0.0	0.02
0.0				
16	130	0.0	0.0	0.00
0.0				
17	0	0.0	0.0	0.00
0.0				
21	50	0.0	0.0	0.00
0.0				

	vol_3g_mb_7	vol_3g_mb_8	monthly_2g_6	monthly_2g_7	monthly_2g_8
\					
8	750.95	11.94	0	1	0
13	0.00	0.00	0	0	0
16	0.00	0.00	0	0	0

17	0.00	0.00	0	0	0
21	0.00	0.00	0	0	0
sachet_2g_6 sachet_2g_7 sachet_2g_8 monthly_3g_6 monthly_3g_7					
\					
8	0	1	3	0	0
13	0	0	3	0	0
16	0	0	0	0	0
17	0	0	0	0	0
21	0	0	0	0	0
monthly_3g_8 sachet_3g_6 sachet_3g_7 sachet_3g_8 aon					
aug_vbc_3g	\				
8	0	0	0	0	315
21.03					
13	0	0	0	0	2607
0.00					
16	0	0	0	0	511
0.00					
17	0	0	0	0	667
0.00					
21	0	0	0	0	720
0.00					
jul_vbc_3g jun_vbc_3g avg_rech_amt_6_7 churn total_mou_good					
\					
8	910.65	122.16	519.0	0	612.22
13	0.00	0.00	380.0	0	1875.70
16	2.45	21.89	459.0	0	711.67
17	0.00	0.00	408.0	0	1341.41
21	0.00	0.00	640.0	0	1067.43
avg_mou_action diff_mou decrease_mou_action avg_rech_num_action					
\					
8	324.125	-288.095		1	17.5
13	1262.390	-613.310		1	7.5
16	597.705	-113.965		1	4.0
17	1.560	-1339.850		1	3.0
21	1245.130	177.700		0	26.5

	diff_rech_num	decrease_rech_num_action	avg_rech_amt_action	\
8	-1.5	1	360.5	
13	1.5	0	485.0	
16	-6.0	1	254.0	
17	-16.0	1	15.0	
21	4.5	0	699.0	

	diff_rech_amt	decrease_rech_amt_action	avg_arpu_action	
diff_arpu \				
8	-76.5	1	314.7925	-
63.9285				
13	-22.0	1	399.4655	-
93.3805				
16	-316.0	1	243.8815	-
187.0935				
17	-801.0	1	22.2395	-
667.7685				
21	99.0	0	617.7565	
103.3035				

	decrease_arpu_action
8	1
13	1
16	1
17	1
21	0

Deriving new column decrease_vbc_action

This column indicates whether the volume based cost of the customer has decreased in the action phase than the good phase.

```
# VBC in action phase
data['avg_vbc_3g_action'] = (data['jul_vbc_3g'] +
data['aug_vbc_3g'])/2

# Difference of good and action phase VBC
data['diff_vbc'] = data['avg_vbc_3g_action'] - data['jun_vbc_3g']

# Checking whether the VBC has decreased on the action month
data['decrease_vbc_action'] = np.where(data['diff_vbc'] < 0 , 1, 0)

data.head()
```

	mobile_number	loc_og_t2o_mou	std_og_t2o_mou	loc_ic_t2o_mou
arpu_6 \				
8	7001524846	0.0	0.0	0.0
378.721				
13	7002191713	0.0	0.0	0.0
492.846				

16	7000875565	0.0	0.0	0.0
430.975				
17	7000187447	0.0	0.0	0.0
690.008				
21	7002124215	0.0	0.0	0.0
514.453				

	arpu_7	arpu_8	onnet_mou_6	onnet_mou_7	onnet_mou_8
offnet_mou_6	\				
8	492.223	137.362	413.69	351.03	35.08
94.66					
13	205.671	593.260	501.76	108.39	534.24
413.31					
16	299.869	187.894	50.51	74.01	70.61
296.29					
17	18.980	25.499	1185.91	9.28	7.79
61.64					
21	597.753	637.760	102.41	132.11	85.14
757.93					

	offnet_mou_7	offnet_mou_8	roam_ic_mou_6	roam_ic_mou_7
roam_ic_mou_8	\			
8	80.63	136.48	0.00	0.00
0.00				
13	119.28	482.46	23.53	144.24
72.11				
16	229.74	162.76	0.00	2.83
0.00				
17	0.00	5.54	0.00	4.76
4.81				
21	896.68	983.39	0.00	0.00
0.00				

	roam_og_mou_6	roam_og_mou_7	roam_og_mou_8	loc_og_t2t_mou_6	\
8	0.00	0.00	0.00	297.13	
13	7.98	35.26	1.44	49.63	
16	0.00	17.74	0.00	42.61	
17	0.00	8.46	13.34	38.99	
21	0.00	0.00	0.00	4.48	

	loc_og_t2t_mou_7	loc_og_t2t_mou_8	loc_og_t2m_mou_6
loc_og_t2m_mou_7	\		
8	217.59	12.49	80.96
70.58			
13	6.19	36.01	151.13
47.28			
16	65.16	67.38	273.29
145.99			
17	0.00	0.00	58.54
0.00			

21	6.16	23.34	91.81	
87.93				
loc_og_t2m_mou_8	loc_og_t2f_mou_6	loc_og_t2f_mou_7		
loc_og_t2f_mou_8 \				
8	50.54	0.00	0.00	
0.00				
13	294.46	4.54	0.00	
23.51				
16	128.28	0.00	4.48	
10.26				
17	0.00	0.00	0.00	
0.00				
21	104.81	0.75	0.00	
1.58				
loc_og_t2c_mou_6	loc_og_t2c_mou_7	loc_og_t2c_mou_8	loc_og_mou_6	
\				
8	0.0	0.0	7.15	378.09
13	0.0	0.0	0.49	205.31
16	0.0	0.0	0.00	315.91
17	0.0	0.0	0.00	97.54
21	0.0	0.0	0.00	97.04
loc_og_mou_7	loc_og_mou_8	std_og_t2t_mou_6	std_og_t2t_mou_7	\
8	288.18	63.04	116.56	133.43
13	53.48	353.99	446.41	85.98
16	215.64	205.93	7.89	2.58
17	0.00	0.00	1146.91	0.81
21	94.09	129.74	97.93	125.94
std_og_t2t_mou_8	std_og_t2m_mou_6	std_og_t2m_mou_7		
std_og_t2m_mou_8 \				
8	22.58	13.69	10.04	
75.69				
13	498.23	255.36	52.94	
156.94				
16	3.23	22.99	64.51	
18.29				
17	0.00	1.55	0.00	
0.00				
21	61.79	665.36	808.74	
876.99				
std_og_t2f_mou_6	std_og_t2f_mou_7	std_og_t2f_mou_8		

std_og_t2c_mou_6 \			
8	0.0	0.0	0.0
0.0			
13	0.0	0.0	0.0
0.0			
16	0.0	0.0	0.0
0.0			
17	0.0	0.0	0.0
0.0			
21	0.0	0.0	0.0
0.0			

	std_og_t2c_mou_7	std_og_t2c_mou_8	std_og_mou_6	std_og_mou_7 \
8	0.0	0.0	130.26	143.48
13	0.0	0.0	701.78	138.93
16	0.0	0.0	30.89	67.09
17	0.0	0.0	1148.46	0.81
21	0.0	0.0	763.29	934.69

	std_og_mou_8	isd_og_mou_6	isd_og_mou_7	isd_og_mou_8
spl_og_mou_6 \				
8	98.28	0.0	0.0	0.00
0.00				
13	655.18	0.0	0.0	1.29
0.00				
16	21.53	0.0	0.0	0.00
0.00				
17	0.00	0.0	0.0	0.00
2.58				
21	938.79	0.0	0.0	0.00
0.00				

	spl_og_mou_7	spl_og_mou_8	og_others_6	og_others_7	og_others_8
\					
8	0.00	10.23	0.00	0.0	0.0
13	0.00	4.78	0.00	0.0	0.0
16	3.26	5.91	0.00	0.0	0.0
17	0.00	0.00	0.93	0.0	0.0
21	0.00	0.00	0.00	0.0	0.0

	total_og_mou_6	total_og_mou_7	total_og_mou_8	
loc_ic_t2t_mou_6 \				
8	508.36	431.66	171.56	23.84
13	907.09	192.41	1015.26	67.88

16	346.81	286.01	233.38	41.33
17	1249.53	0.81	0.00	34.54
21	860.34	1028.79	1068.54	2.48

	loc_ic_t2t_mou_7	loc_ic_t2t_mou_8	loc_ic_t2m_mou_6	
loc_ic_t2m_mou_7 \				
8	9.84	0.31	57.58	
13.98				
13	7.58	52.58	142.88	
18.53				
16	71.44	28.89	226.81	
149.69				
17	0.00	0.00	47.41	
2.31				
21	10.19	19.54	118.23	
74.63				

	loc_ic_t2m_mou_8	loc_ic_t2f_mou_6	loc_ic_t2f_mou_7	
loc_ic_t2f_mou_8 \				
8	15.48	0.00	0.00	
0.00				
13	195.18	4.81	0.00	
7.49				
16	150.16	8.71	8.68	
32.71				
17	0.00	0.00	0.00	
0.00				
21	129.16	4.61	2.84	
10.39				

	loc_ic_mou_6	loc_ic_mou_7	loc_ic_mou_8	std_ic_t2t_mou_6 \
8	81.43	23.83	15.79	0.00
13	215.58	26.11	255.26	115.68
16	276.86	229.83	211.78	68.79
17	81.96	2.31	0.00	8.63
21	125.33	87.68	159.11	14.06

	std_ic_t2t_mou_7	std_ic_t2t_mou_8	std_ic_t2m_mou_6	
std_ic_t2m_mou_7 \				
8	0.58	0.10	22.43	
4.08				
13	38.29	154.58	308.13	
29.79				
16	78.64	6.33	18.68	
73.08				
17	0.00	0.00	1.28	

0.00			
21	5.98	0.18	67.69
38.23			

	std_ic_t2m_mou_8	std_ic_t2f_mou_6	std_ic_t2f_mou_7
std_ic_t2f_mou_8 \			
8	0.65	0.00	0.0
0.00			
13	317.91	0.00	0.0
1.91			
16	73.93	0.51	0.0
2.18			
17	0.00	0.00	0.0
0.00			
21	101.74	0.00	0.0
0.00			

	std_ic_t2o_mou_6	std_ic_t2o_mou_7	std_ic_t2o_mou_8	std_ic_mou_6
\				
8	0.0	0.0	0.0	22.43
13	0.0	0.0	0.0	423.81
16	0.0	0.0	0.0	87.99
17	0.0	0.0	0.0	9.91
21	0.0	0.0	0.0	81.76

	std_ic_mou_7	std_ic_mou_8	total_ic_mou_6	total_ic_mou_7	\
8	4.66	0.75	103.86	28.49	
13	68.09	474.41	968.61	172.58	
16	151.73	82.44	364.86	381.56	
17	0.00	0.00	91.88	2.31	
21	44.21	101.93	207.09	131.89	

	total_ic_mou_8	spl_ic_mou_6	spl_ic_mou_7	spl_ic_mou_8
isd_ic_mou_6 \				
8	16.54	0.00	0.0	0.0
0.00				
13	1144.53	0.45	0.0	0.0
245.28				
16	294.46	0.00	0.0	0.0
0.00				
17	0.00	0.00	0.0	0.0
0.00				
21	261.04	0.00	0.0	0.0
0.00				

	isd_ic_mou_7	isd_ic_mou_8	ic_others_6	ic_others_7	ic_others_8
\					
8	0.00	0.00	0.00	0.00	0.00
13	62.11	393.39	83.48	16.24	21.44
16	0.00	0.23	0.00	0.00	0.00
17	0.00	0.00	0.00	0.00	0.00
21	0.00	0.00	0.00	0.00	0.00
	total_rech_num_6	total_rech_num_7	total_rech_num_8		
total_rech_amt_6	\				
8	19	21	14		
437					
13	6	4	11		
507					
16	10	6	2		
570					
17	19	2	4		
816					
21	22	26	27		
600					
	total_rech_amt_7	total_rech_amt_8	max_rech_amt_6	max_rech_amt_7	
\					
8	601	120	90	154	
13	253	717	110	110	
16	348	160	110	110	
17	0	30	110	0	
21	680	718	50	50	
	max_rech_amt_8	last_day_rch_amt_6	last_day_rch_amt_7	\	
8	30	50	0		
13	130	110	50		
16	130	100	100		
17	30	30	0		
21	50	30	20		
	last_day_rch_amt_8	vol_2g_mb_6	vol_2g_mb_7	vol_2g_mb_8	
vol_3g_mb_6	\				
8	10	0.0	356.0	0.03	
0.0					
13	0	0.0	0.0	0.02	

0.0					
16	130	0.0	0.0	0.00	
0.0					
17	0	0.0	0.0	0.00	
0.0					
21	50	0.0	0.0	0.00	
0.0					
	vol_3g_mb_7	vol_3g_mb_8	monthly_2g_6	monthly_2g_7	monthly_2g_8
\					
8	750.95	11.94	0	1	0
13	0.00	0.00	0	0	0
16	0.00	0.00	0	0	0
17	0.00	0.00	0	0	0
21	0.00	0.00	0	0	0
	sachet_2g_6	sachet_2g_7	sachet_2g_8	monthly_3g_6	monthly_3g_7
\					
8	0	1	3	0	0
13	0	0	3	0	0
16	0	0	0	0	0
17	0	0	0	0	0
21	0	0	0	0	0
	monthly_3g_8	sachet_3g_6	sachet_3g_7	sachet_3g_8	aon
aug_vbc_3g	\				
8	0	0	0	0	315
21.03					
13	0	0	0	0	2607
0.00					
16	0	0	0	0	511
0.00					
17	0	0	0	0	667
0.00					
21	0	0	0	0	720
0.00					
	jul_vbc_3g	jun_vbc_3g	avg_rech_amt_6_7	churn	total_mou_good
\					
8	910.65	122.16	519.0	0	612.22
13	0.00	0.00	380.0	0	1875.70
16	2.45	21.89	459.0	0	711.67

17	0.00	0.00	408.0	0	1341.41
21	0.00	0.00	640.0	0	1067.43
	avg_mou_action	diff_mou	decrease_mou_action	avg_rech_num_action	
\					
8	324.125	-288.095	1	17.5	
13	1262.390	-613.310	1	7.5	
16	597.705	-113.965	1	4.0	
17	1.560	-1339.850	1	3.0	
21	1245.130	177.700	0	26.5	
	diff_rech_num	decrease_rech_num_action	avg_rech_amt_action	\	
8	-1.5	1	360.5		
13	1.5	0	485.0		
16	-6.0	1	254.0		
17	-16.0	1	15.0		
21	4.5	0	699.0		
	diff_rech_amt	decrease_rech_amt_action	avg_arpu_action		
diff_arpu \					
8	-76.5	1	314.7925	-	
63.9285					
13	-22.0	1	399.4655	-	
93.3805					
16	-316.0	1	243.8815	-	
187.0935					
17	-801.0	1	22.2395	-	
667.7685					
21	99.0	0	617.7565		
103.3035					
	decrease_arpu_action	avg_vbc_3g_action	diff_vbc		
decrease_vbc_action					
8	1	465.840	343.680		
0					
13	1	0.000	0.000		
0					
16	1	1.225	-20.665		
1					
17	1	0.000	0.000		
0					
21	0	0.000	0.000		
0					

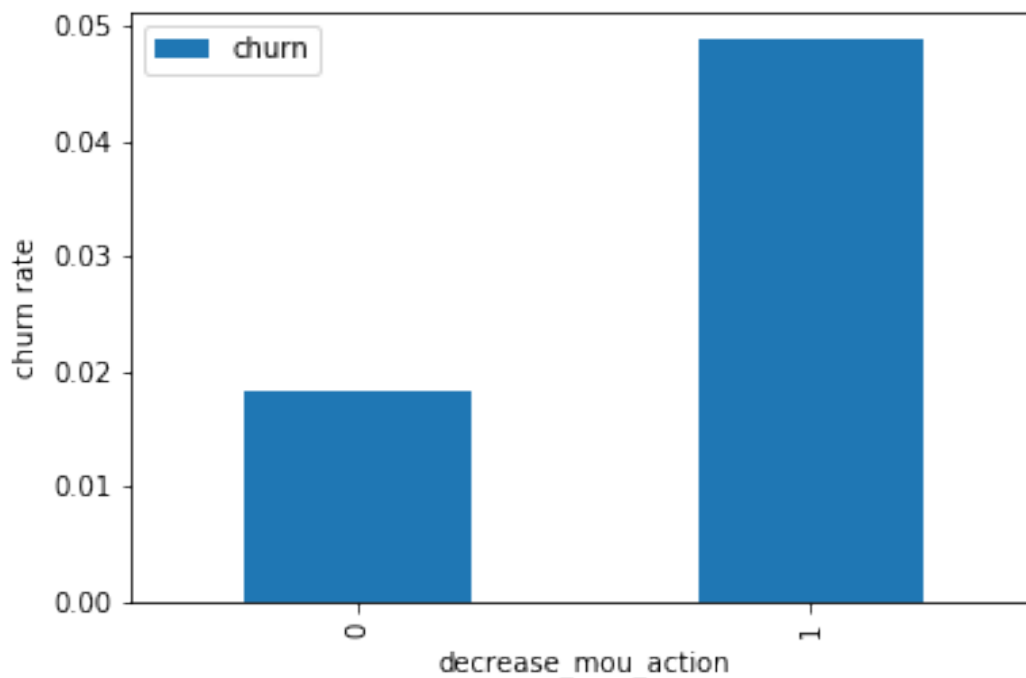
EDA

Univariate analysis

Churn rate on the basis whether the customer decreased her/his MOU in action month

```
# Converting churn column to int in order to do aggfunc in the pivot table
data['churn'] = data['churn'].astype('int64')

data.pivot_table(values='churn', index='decrease_mou_action',
aggfunc='mean').plot.bar()
plt.ylabel('churn rate')
plt.show()
```

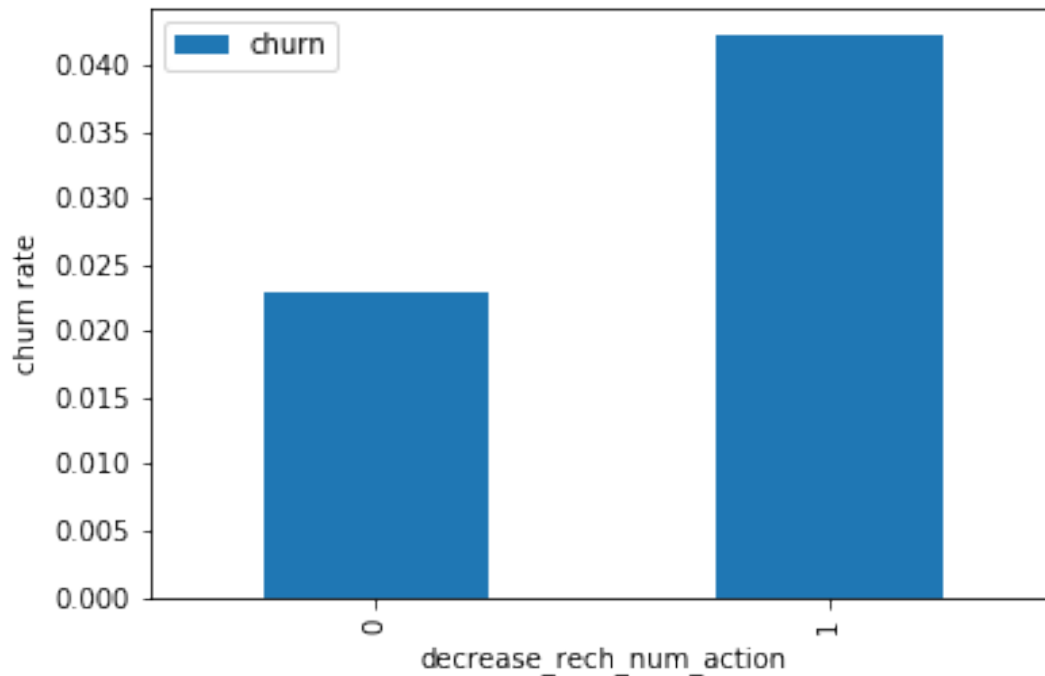


Analysis

We can see that the churn rate is more for the customers, whose minutes of usage(mou) decreased in the action phase than the good phase.

Churn rate on the basis whether the customer decreased her/his number of recharge in action month

```
data.pivot_table(values='churn', index='decrease_rech_num_action',
aggfunc='mean').plot.bar()
plt.ylabel('churn rate')
plt.show()
```

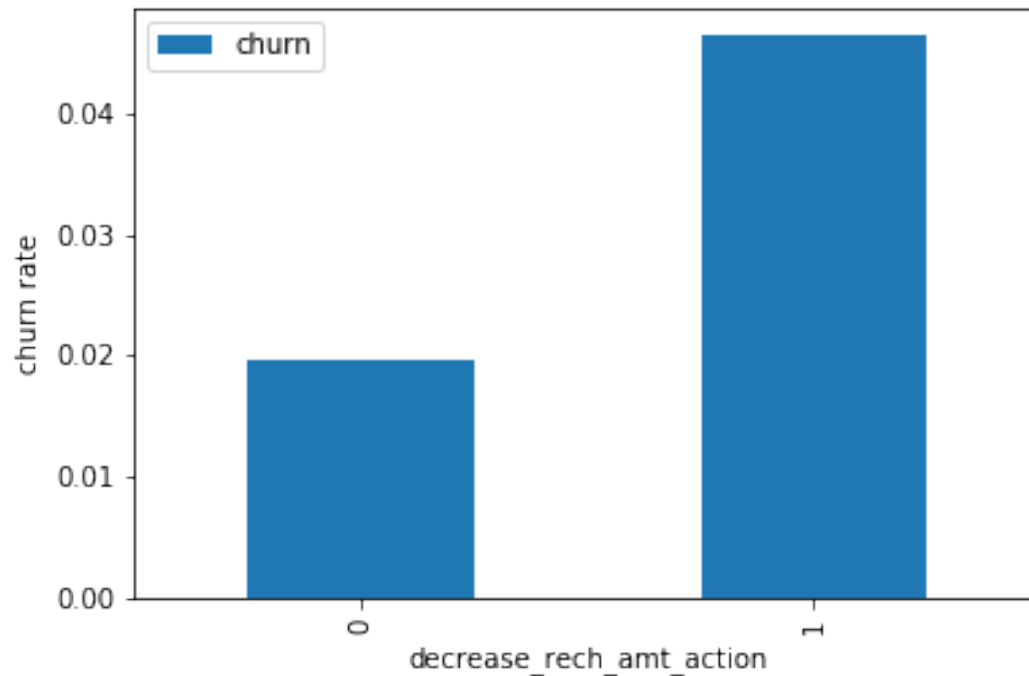


Analysis

As expected, the churn rate is more for the customers, whose number of recharge in the action phase is lesser than the number in good phase.

Churn rate on the basis whether the customer decreased her/his amount of recharge in action month

```
data.pivot_table(values='churn', index='decrease_rech_amt_action',  
aggfunc='mean').plot.bar()  
plt.ylabel('churn rate')  
plt.show()
```

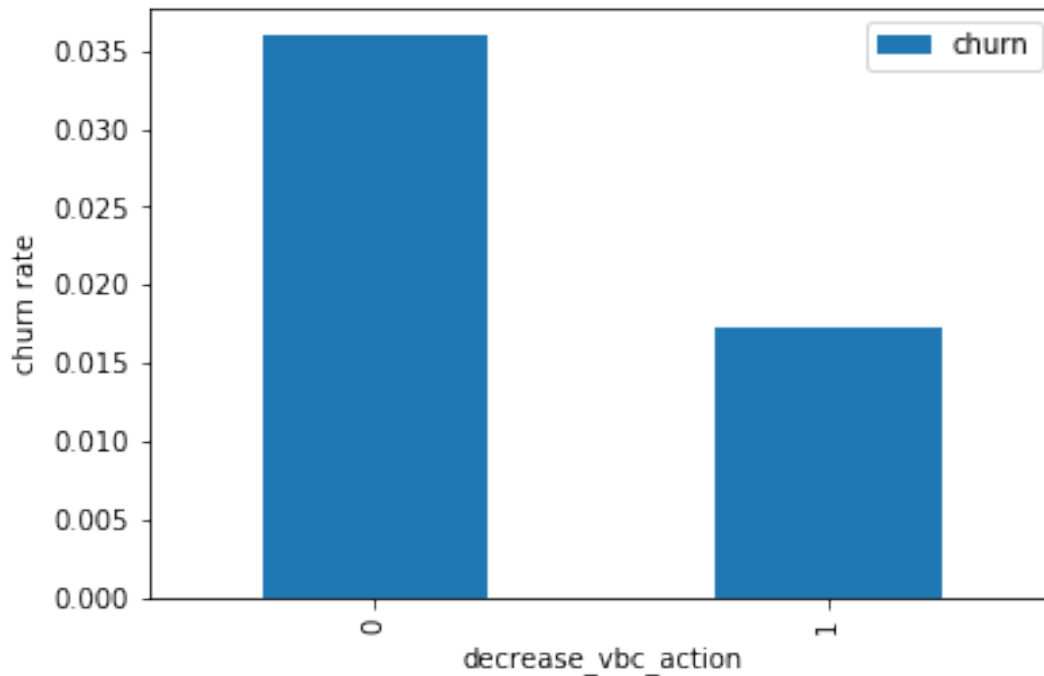


Analysis

Here also we see the same behaviour. The churn rate is more for the customers, whose amount of recharge in the action phase is lesser than the amount in good phase.

Churn rate on the basis whether the customer decreased her/his volume based cost in action month

```
data.pivot_table(values='churn', index='decrease_vbc_action',  
aggfunc='mean').plot.bar()  
plt.ylabel('churn rate')  
plt.show()
```



Analysis

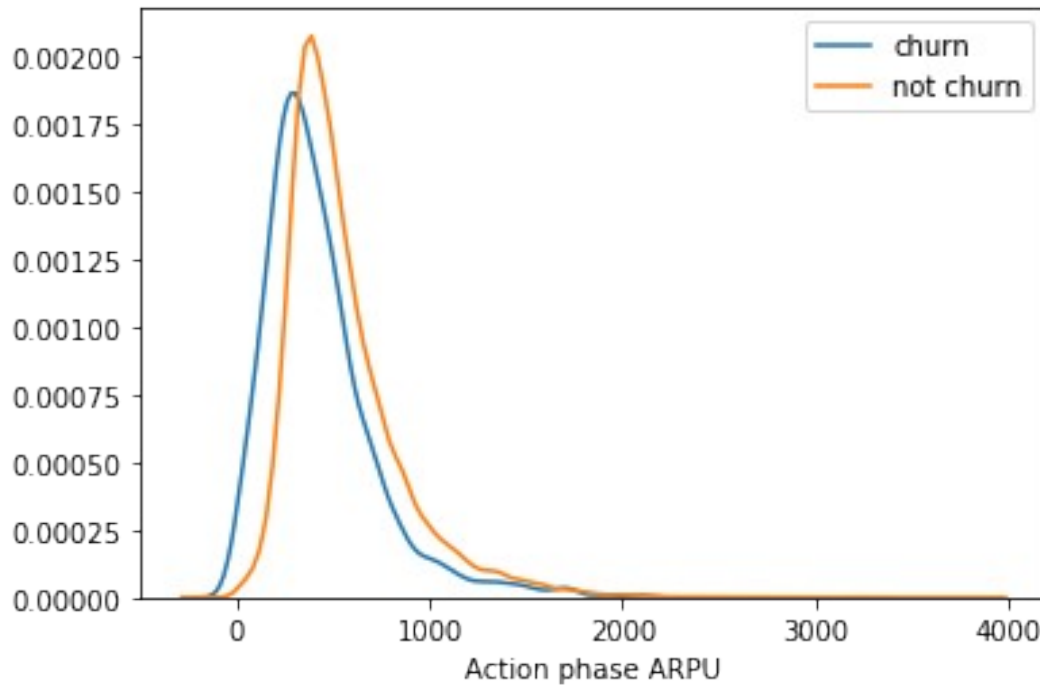
Here we see the expected result. The churn rate is more for the customers, whose volume based cost in action month is increased. That means the customers do not do the monthly recharge more when they are in the action phase.

Analysis of the average revenue per customer (churn and not churn) in the action phase

```
# Creating churn dataframe
data_churn = data[data['churn'] == 1]
# Creating not churn dataframe
data_non_churn = data[data['churn'] == 0]

# Distribution plot
ax =
sns.distplot(data_churn['avg_arpu_action'],label='churn',hist=False)
ax = sns.distplot(data_non_churn['avg_arpu_action'],label='not
churn',hist=False)
ax.set(xlabel='Action phase ARPU')

[Text(0.5, 0, 'Action phase ARPU')]
```



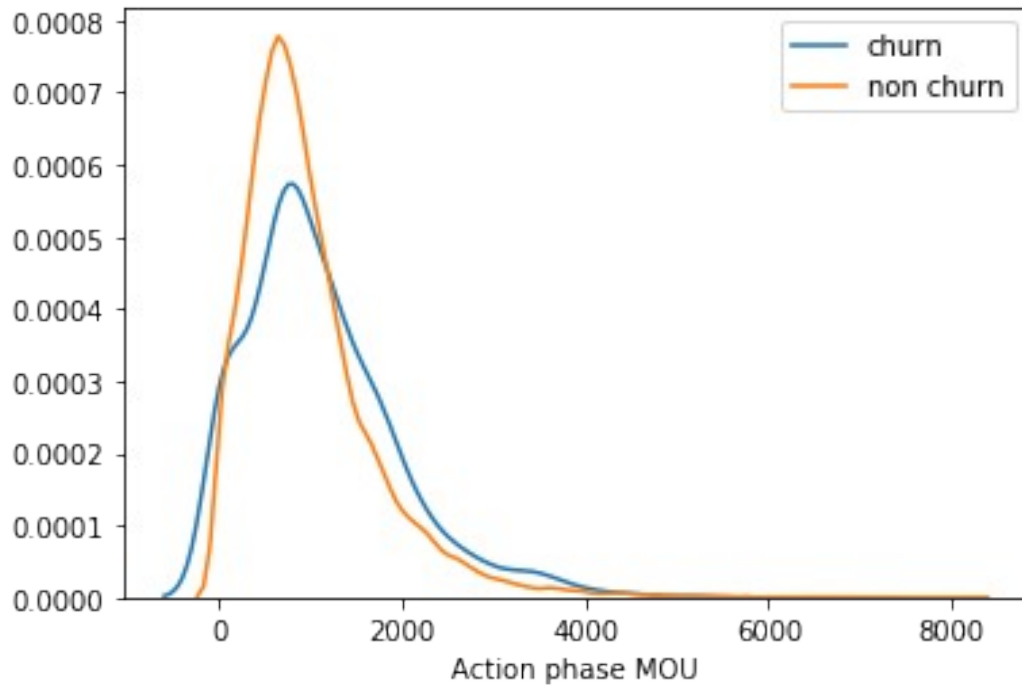
Average revenue per user (ARPU) for the churned customers is mostly dense on the 0 to 900. The higher ARPU customers are less likely to be churned.

ARPU for the not churned customers is mostly dense on the 0 to 1000.

Analysis of the minutes of usage MOU (churn and not churn) in the action phase

```
# Distribution plot
ax =
sns.distplot(data_churn['total_mou_good'], label='churn', hist=False)
ax = sns.distplot(data_non_churn['total_mou_good'], label='non
churn', hist=False)
ax.set(xlabel='Action phase MOU')

[Text(0.5, 0, 'Action phase MOU')]
```

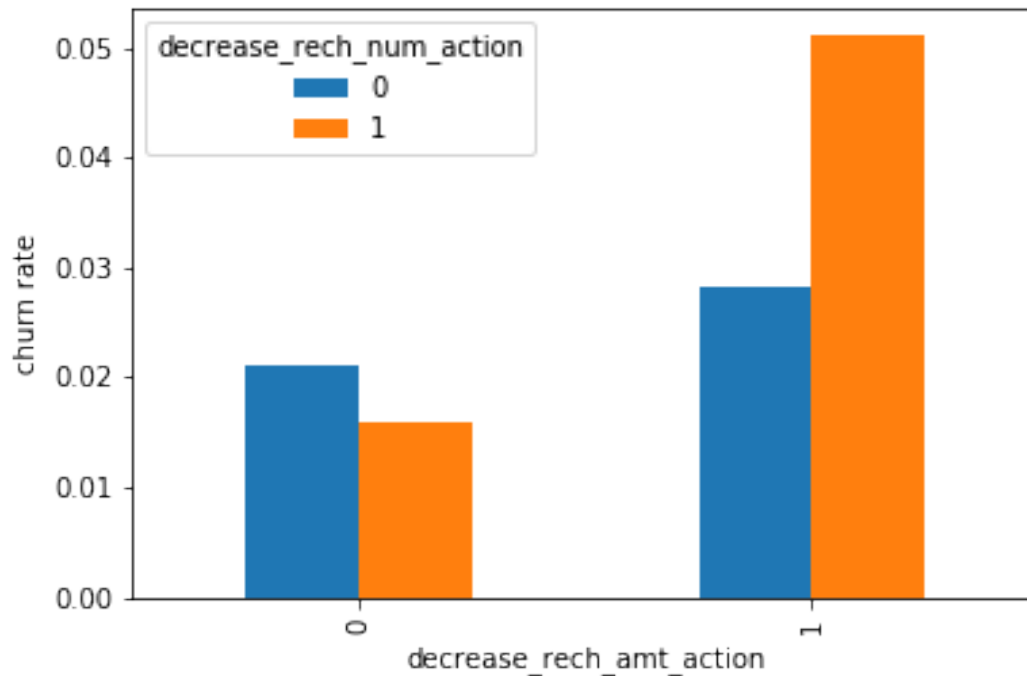


Minutes of usage(MOU) of the churn customers is mostly populated on the 0 to 2500 range. Higher the MOU, lesser the churn probability.

Bivariate analysis

Analysis of churn rate by the decreasing recharge amount and number of recharge in the action phase

```
data.pivot_table(values='churn', index='decrease_rech_amt_action',  
columns='decrease_rech_num_action', aggfunc='mean').plot.bar()  
plt.ylabel('churn rate')  
plt.show()
```

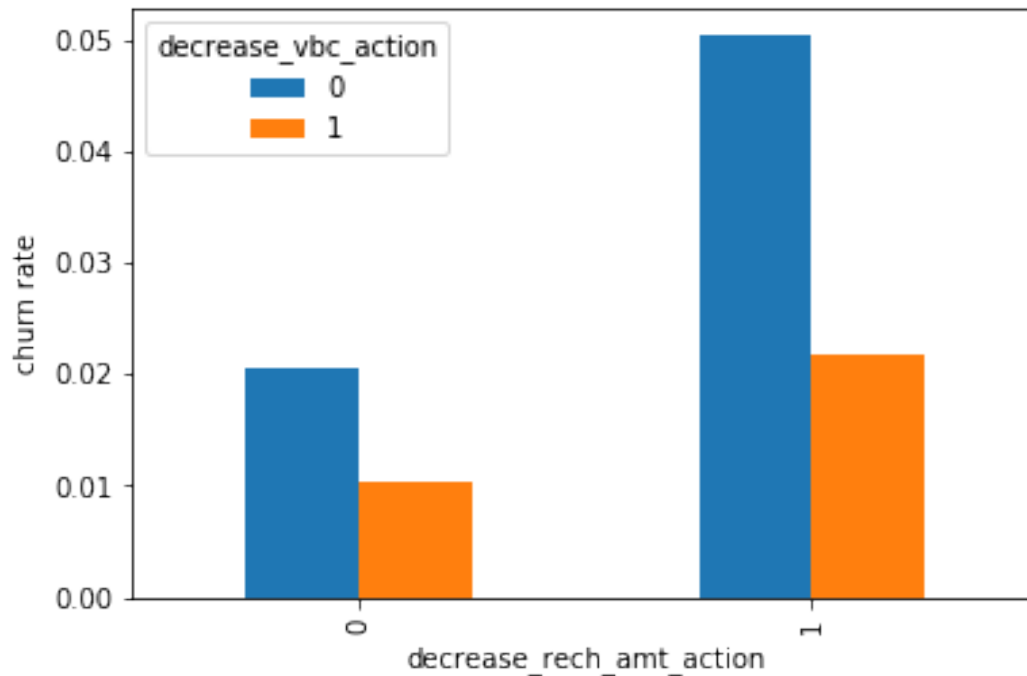



Analysis

We can see from the above plot, that the churn rate is more for the customers, whose recharge amount as well as number of recharge have decreased in the action phase than the good phase.

Analysis of churn rate by the decreasing recharge amount and volume based cost in the action phase

```
data.pivot_table(values='churn', index='decrease_rech_amt_action',  
columns='decrease_vbc_action', aggfunc='mean').plot.bar()  
plt.ylabel('churn rate')  
plt.show()
```

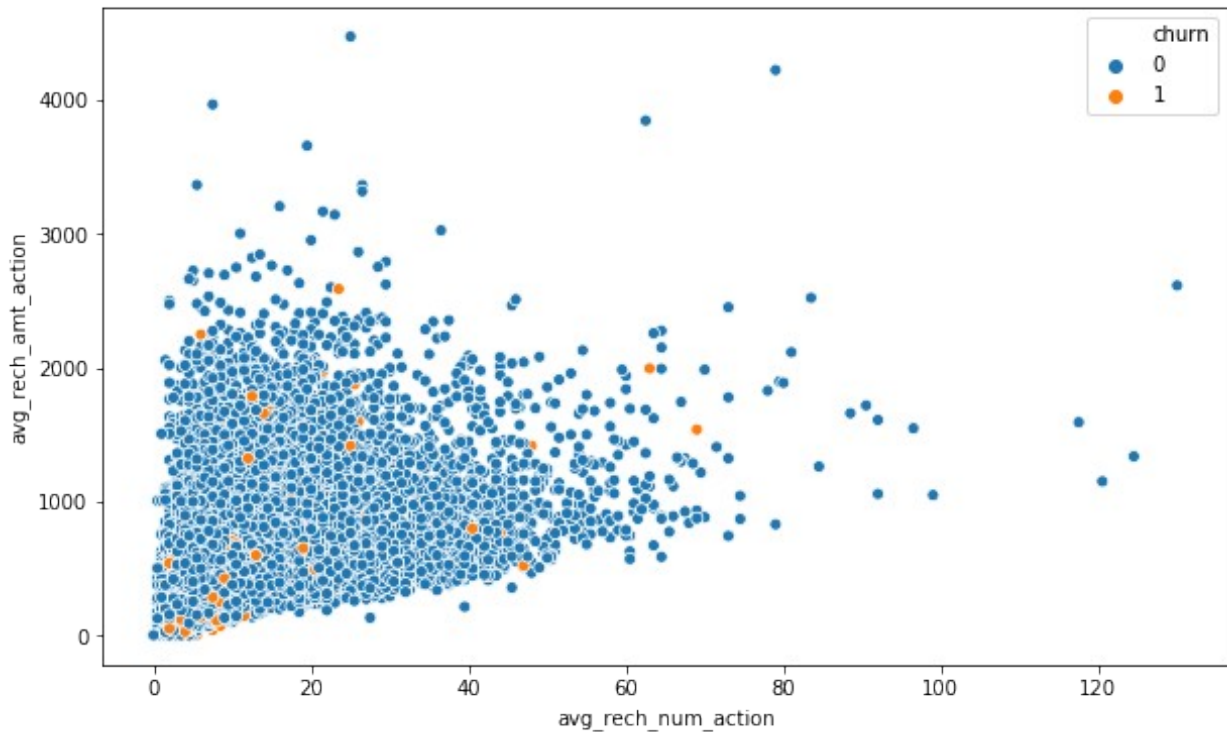


Analysis

Here, also we can see that the churn rate is more for the customers, whose recharge amount is decreased along with the volume based cost is increased in the action month.

Analysis of recharge amount and number of recharge in action month

```
plt.figure(figsize=(10,6))  
ax = sns.scatterplot('avg_rech_num_action', 'avg_rech_amt_action',  
hue='churn', data=data)
```



Analysis

We can see from the above pattern that the recharge number and the recharge amount are mostly proportional. More the number of recharge, more the amount of the recharge.

Dropping few derived columns, which are not required in further analysis

```
data =
data.drop(['total_mou_good', 'avg_mou_action', 'diff_mou', 'avg_rech_num_
action', 'diff_rech_num', 'avg_rech_amt_action',
'diff_rech_amt', 'avg_arpu_action', 'diff_arpu', 'avg_vbc_3g_action', 'dif
f_vbc', 'avg_rech_amt_6_7'], axis=1)
```

Train-Test Split

```
# Import library
from sklearn.model_selection import train_test_split

# Putting feature variables into X
X = data.drop(['mobile_number', 'churn'], axis=1)

# Putting target variable to y
y = data['churn']

# Splitting data into train and test set 80:20
X_train, X_test, y_train, y_test = train_test_split(X, y,
train_size=0.8, test_size=0.2, random_state=100)
```

Dealing with data imbalance

We are creating synthetic samples by doing upsampling using SMOTE(Synthetic Minority Oversampling Technique).

```
# Importing SMOTE
from imblearn.over_sampling import SMOTE

# Instantiate SMOTE
sm = SMOTE(random_state=27)

# Fitting SMOTE to the train set
X_train, y_train = sm.fit_sample(X_train, y_train)
```

Feature Scaling

```
# Standardization method
from sklearn.preprocessing import StandardScaler

# Instantiate the Scaler
scaler = StandardScaler()

# List of the numeric columns
cols_scale = X_train.columns.to_list()
# Removing the derived binary columns
cols_scale.remove('decrease_mou_action')
cols_scale.remove('decrease_rech_num_action')
cols_scale.remove('decrease_rech_amt_action')
cols_scale.remove('decrease_arpu_action')
cols_scale.remove('decrease_vbc_action')

# Fit the data into scaler and transform
X_train[cols_scale] = scaler.fit_transform(X_train[cols_scale])

X_train.head()
```

	loc_og_t2o_mou	std_og_t2o_mou	loc_ic_t2o_mou	arpu_6	arpu_7
0	0.0	0.0	0.0	0.140777	-0.522792
1	0.0	0.0	0.0	-1.427243	4.428047
2	0.0	0.0	0.0	-0.222751	0.543206
3	0.0	0.0	0.0	-0.911173	0.842273
4	0.0	0.0	0.0	0.271356	0.247684

	arpu_8	onnet_mou_6	onnet_mou_7	onnet_mou_8	offnet_mou_6	\
0	-0.276289	0.106540	-0.662084	-0.465777	-0.211202	

1	3.254270	-0.658491	-0.236590	-0.004450	-0.776075
2	0.809117	-0.601239	-0.599206	-0.331043	-0.363395
3	0.731302	-0.702232	-0.650471	-0.458464	-0.789784
4	1.256421	-0.356392	-0.180394	0.114727	0.899204

	offnet_mou_7	offnet_mou_8	roam_ic_mou_6	roam_ic_mou_7	
roam_ic_mou_8 \					
0	-0.636415	0.317224	-0.254996	-0.001208	-
0.235211					
1	2.523985	2.732154	-0.254996	-0.253231	-
0.304660					
2	-0.495976	-0.028236	-0.254996	-0.253231	-
0.304660					
3	-0.654483	-0.519047	-0.254996	-0.253231	-
0.304660					
4	0.904465	1.255807	-0.231882	-0.253231	-
0.304660					

	roam_og_mou_6	roam_og_mou_7	roam_og_mou_8	loc_og_t2t_mou_6	\
0	-0.300833	-0.374857	-0.412810	-0.263308	
1	-0.300833	-0.374857	-0.431026	-0.201396	
2	-0.300833	-0.374857	-0.431026	0.077694	
3	-0.300833	-0.374857	-0.431026	-0.192289	
4	-0.202644	-0.374857	-0.431026	0.128384	

	loc_og_t2t_mou_7	loc_og_t2t_mou_8	loc_og_t2m_mou_6	
loc_og_t2m_mou_7 \				
0	-0.311548	-0.251411	0.485770	-
0.190660				
1	0.270791	0.198344	-0.529474	
1.106670				
2	-0.095916	0.228431	0.605362	
0.258376				
3	-0.181513	-0.064925	-0.371787	-
0.205099				
4	0.784682	1.062326	1.423002	
0.996094				

	loc_og_t2m_mou_8	loc_og_t2f_mou_6	loc_og_t2f_mou_7	
loc_og_t2f_mou_8 \				
0	-0.399182	-0.256866	-0.267401	-
0.244832				
1	0.288951	-0.276320	-0.267401	-
0.244832				
2	0.908270	1.475098	0.451689	-
0.131562				
3	-0.251524	-0.157090	0.216496	-
0.244832				
4	1.845573	0.780430	1.055332	
0.519904				

	loc_og_t2c_mou_6	loc_og_t2c_mou_7	loc_og_t2c_mou_8	loc_og_mou_6
\				
0	-0.191587	-0.267368	-0.244432	0.129144
1	-0.191587	-0.267368	-0.244432	-0.477059
2	-0.191587	-0.267368	-0.244432	0.512549
3	1.002136	2.438345	2.557369	-0.364845
4	1.811266	-0.267368	0.843143	1.025297

	loc_og_mou_7	loc_og_mou_8	std_og_t2t_mou_6	std_og_t2t_mou_7	\
0	-0.335468	-0.418749	0.254982	-0.528622	
1	0.843930	0.290569	-0.570615	-0.320253	
2	0.121104	0.710496	-0.618738	-0.551860	
3	-0.233086	-0.212616	-0.619956	-0.570510	
4	1.183543	1.843624	-0.414192	-0.474892	

	std_og_t2t_mou_8	std_og_t2m_mou_6	std_og_t2m_mou_7	
std_og_t2m_mou_8 \				
0	-0.338018	-0.342394	-0.504282	
0.650664				
1	-0.041333	-0.512504	2.294191	
3.087483				
2	-0.420186	-0.617043	-0.571393	-
0.416795				
3	-0.420186	-0.621707	-0.578677	-
0.406309				
4	-0.327001	0.357250	0.585026	
0.555984				

	std_og_t2f_mou_6	std_og_t2f_mou_7	std_og_t2f_mou_8	
std_og_t2c_mou_6 \				
0	-0.143576	-0.139257	-0.119299	
0.0				
1	-0.143576	-0.139257	-0.119299	
0.0				
2	-0.143576	-0.139257	-0.067469	
0.0				
3	-0.143576	-0.139257	-0.119299	
0.0				
4	-0.143576	-0.139257	-0.119299	
0.0				

	std_og_t2c_mou_7	std_og_t2c_mou_8	std_og_mou_6	std_og_mou_7	\
0	0.0	0.0	-0.048161	-0.731560	
1	0.0	0.0	-0.771902	1.368343	

2	0.0	0.0	-0.878705	-0.794939
3	0.0	0.0	-0.882786	-0.813361
4	0.0	0.0	-0.062970	0.066271

	std_og_mou_8	isd_og_mou_6	isd_og_mou_7	isd_og_mou_8	
spl_og_mou_6 \					
0	0.214243	-0.080803	-0.092449	-0.061631	-
0.347585					
1	2.063999	-0.080803	-0.092449	-0.061631	-
0.347585					
2	-0.563412	-0.080803	-0.031701	-0.061631	-
0.347585					
3	-0.557142	-0.080803	-0.092449	-0.061631	
0.299948					
4	0.157380	-0.080803	-0.092449	0.132387	
0.906003					

	spl_og_mou_7	spl_og_mou_8	og_others_6	og_others_7	
og_others_8 \					
0	-0.363159	-0.017165	-0.346191	-0.015583	-0.013735
1	-0.363159	-0.290355	-0.346191	-0.015583	-0.013735
2	-0.203140	0.151727	-0.346191	-0.015583	-0.013735
3	0.639325	0.743145	-0.244436	-0.015583	-0.013735
4	-0.251495	0.107282	-0.346191	-0.015583	-0.013735

	total_og_mou_6	total_og_mou_7	total_og_mou_8	loc_ic_t2t_mou_6 \
0	-0.000389	-0.860412	-0.011382	-0.203981
1	-0.970285	1.670188	1.938953	-0.410762
2	-0.637091	-0.716013	-0.155609	-0.073331
3	-1.012362	-0.864684	-0.569774	-0.262725
4	0.411184	0.572929	1.014123	-0.238919

	loc_ic_t2t_mou_7	loc_ic_t2t_mou_8	loc_ic_t2m_mou_6	
loc_ic_t2m_mou_7 \				
0	-0.266718	-0.242771	-0.380593	-
0.272733				
1	0.193158	0.156537	-0.481723	
0.744741				
2	-0.082299	0.189717	0.211940	
0.166326				
3	-0.287643	-0.150724	0.157353	
0.540086				
4	0.483606	0.750395	-0.281606	
0.384609				

	loc_ic_t2m_mou_8	loc_ic_t2f_mou_6	loc_ic_t2f_mou_7	
loc_ic_t2f_mou_8 \				
0	-0.437571	-0.290528	-0.270877	-
0.150060				
1	0.256589	-0.290528	-0.270877	-
0.257696				
2	0.542595	0.223523	-0.117519	
0.167136				
3	-0.095861	-0.290528	-0.268736	-
0.250781				
4	0.578588	0.220005	0.304016	
0.113318				

	loc_ic_mou_6	loc_ic_mou_7	loc_ic_mou_8	std_ic_t2t_mou_6	\
0	-0.409101	-0.363983	-0.440411	-0.175106	
1	-0.583307	0.570197	0.219470	-0.215496	
2	0.142990	0.054813	0.490068	-0.215496	
3	-0.061177	0.184367	-0.172152	-0.215496	
4	-0.286414	0.556564	0.777995	-0.215496	

	std_ic_t2t_mou_7	std_ic_t2t_mou_8	std_ic_t2m_mou_6	
std_ic_t2m_mou_7 \				
0	-0.159825	0.078711	-0.164347	
0.367474				
1	-0.200464	-0.112725	-0.355157	
0.100763				
2	-0.200464	-0.187265	-0.361304	-
0.256979				
3	-0.200464	-0.187265	-0.361304	-
0.343715				
4	-0.200464	0.108490	-0.066471	
1.099027				

	std_ic_t2m_mou_8	std_ic_t2f_mou_6	std_ic_t2f_mou_7	
std_ic_t2f_mou_8 \				
0	-0.117454	-0.135479	-0.137327	-
0.110642				
1	-0.034777	-0.135479	-0.137327	-
0.110642				
2	-0.217027	-0.135479	-0.137327	-
0.109904				
3	-0.229338	-0.135479	-0.137327	-
0.110642				
4	0.087246	0.078665	-0.072592	
0.078362				

	std_ic_t2o_mou_6	std_ic_t2o_mou_7	std_ic_t2o_mou_8	std_ic_mou_6
\				
0	0.0	0.0	0.0	-0.234904

1	0.0	0.0	0.0	-0.386264
2	0.0	0.0	0.0	-0.390310
3	0.0	0.0	0.0	-0.390310
4	0.0	0.0	0.0	-0.171917

	std_ic_mou_7	std_ic_mou_8	total_ic_mou_6	total_ic_mou_7	
total_ic_mou_8	\				
0	0.121332	-0.064154	-0.475564	-0.287010	-
0.420829					
1	-0.078694	-0.096335	-0.688082	0.417278	
0.125859					
2	-0.312284	-0.272135	-0.073424	-0.106086	
0.290685					
3	-0.368919	-0.281605	-0.248992	-0.029566	-
0.273184					
4	0.581141	0.130311	-0.108798	2.215081	
2.220000					

	spl_ic_mou_6	spl_ic_mou_7	spl_ic_mou_8	isd_ic_mou_6	
isd_ic_mou_7	\				
0	-0.366516	-0.089786	-0.192624	-0.151655	-
0.153778					
1	-0.366516	-0.089786	-0.192624	-0.151655	-
0.153778					
2	-0.366516	-0.089786	-0.192624	-0.132791	-
0.099984					
3	-0.366516	0.968997	-0.192624	-0.151655	-
0.153778					
4	-0.366516	-0.089786	-0.192624	1.103151	
4.262216					

	isd_ic_mou_8	ic_others_6	ic_others_7	ic_others_8	
total_rech_num_6	\				
0	-0.126576	-0.099745	-0.121704	-0.081491	
0.192736					
1	-0.126576	-0.099745	-0.121704	-0.081491	-
0.738325					
2	-0.126576	-0.099745	-0.121704	-0.081491	-
0.738325					
3	-0.126576	-0.099745	-0.121704	-0.081491	-
0.272794					
4	4.608310	0.746998	26.877658	25.149134	-
0.738325					

	total_rech_num_7	total_rech_num_8	total_rech_amt_6	
total_rech_amt_7	\			

0	-0.444988	0.305289	0.044172	-
0.726027				
1	4.142873	2.933529	-1.364090	
4.102682				
2	-0.327351	-0.195328	-0.110493	
0.728436				
3	0.378474	0.555597	-0.946224	
0.693869				
4	-0.209713	0.555597	-0.368267	
0.175368				

	total_rech_amt_8	max_rech_amt_6	max_rech_amt_7	max_rech_amt_8	\
0	-0.235478	0.054992	0.023937	0.029739	
1	3.350107	-0.748908	-0.386255	-0.054702	
2	0.772451	0.039532	0.023937	0.198621	
3	0.803137	0.054992	0.768359	0.966264	
4	1.225665	-0.207821	-0.158371	0.029739	

	last_day_rch_amt_6	last_day_rch_amt_7	last_day_rch_amt_8	
vol_2g_mb_6 \				
0	0.601511	-0.811577	-0.626096	-
0.094017				
1	-0.405085	-0.350629	-0.066907	-
0.245535				
2	0.272431	0.386888	0.565618	-
0.077862				
3	-0.598662	-0.535008	-0.351085	-
0.059437				
4	0.272431	0.386888	0.565618	-
0.245535				

	vol_2g_mb_7	vol_2g_mb_8	vol_3g_mb_6	vol_3g_mb_7	vol_3g_mb_8	\
0	0.696113	1.750783	0.510634	1.202971	-0.241652	
1	-0.235847	-0.207939	-0.262491	-0.274601	-0.249913	
2	-0.034247	0.104903	0.950720	1.994409	1.671342	
3	-0.040569	-0.027155	2.610032	5.767468	4.000137	
4	-0.235847	-0.207939	-0.262491	-0.274601	-0.249913	

	monthly_2g_6	monthly_2g_7	monthly_2g_8	sachet_2g_6	sachet_2g_7	\
0	3.236849	3.104207	-0.232664	4.023237	2.358097	
1	-0.246650	-0.251375	-0.232664	-0.255793	-0.269796	
2	-0.246650	3.104207	-0.232664	0.457379	1.044151	
3	3.236849	-0.251375	-0.232664	-0.255793	-0.269796	
4	-0.246650	-0.251375	-0.232664	-0.255793	-0.269796	

	sachet_2g_8	monthly_3g_6	monthly_3g_7	monthly_3g_8	sachet_3g_6
0	2.447476	-0.224183	-0.221779	-0.216364	-0.141182
1	-0.268245	-0.224183	-0.221779	-0.216364	-0.141182
2	1.089616	-0.224183	-0.221779	-0.216364	1.315163
3	-0.268245	2.171393	9.083717	4.618685	-0.141182
4	-0.268245	-0.224183	-0.221779	-0.216364	-0.141182

	sachet_3g_7	sachet_3g_8	aon	aug_vbc_3g	jul_vbc_3g	jun_vbc_3g
0	-0.136208	-0.113882	-0.361238	-0.236209	-0.265392	0.110582
1	-0.136208	-0.113882	-0.790173	-0.255884	-0.265392	0.259366
2	2.575301	2.526725	1.571302	3.307334	2.691063	1.700218
3	-0.136208	-0.113882	-0.951024	-0.255884	-0.265392	0.259366
4	-0.136208	-0.113882	-0.519757	-0.255884	-0.265392	0.259366

	decrease_mou_action	decrease_rech_num_action	decrease_rech_amt_action
0	1	1	1
1	0	0	0
2	1	0	0
3	0	0	0
4	0	0	0

	decrease_arpu_action	decrease_vbc_action
0	1	1
1	0	0
2	0	0
3	0	0
4	0	0

Scaling the test set

We don't fit scaler on the test set. We only transform the test set.

```
# Transform the test set
```

```
X_test[cols_scale] = scaler.transform(X_test[cols_scale])
```

```
X_test.head()
```

	loc_og_t2o_mou	std_og_t2o_mou	loc_ic_t2o_mou	arpu_6
arpu_7 \				
5704	0.0	0.0	0.0	0.244310 -
0.268832				
64892	0.0	0.0	0.0	0.048359 -
0.779609				
39613	0.0	0.0	0.0	0.545470
0.184388				
93118	0.0	0.0	0.0	0.641508
0.816632				
81235	0.0	0.0	0.0	3.878627
0.911619				

	arpu_8	onnet_mou_6	onnet_mou_7	onnet_mou_8
offnet_mou_6 \				
5704 1.005890	-0.725286	-0.690223	-0.476634	0.483540
64892 -0.157969	-0.734066	-0.698072	-0.502219	-0.358555
39613 1.403349	-0.537110	-0.521615	-0.206890	0.694901
93118 -0.211023	-0.058843	0.029897	-0.155872	-0.148197
81235 2.745295	4.117829	1.452446	2.809582	-0.002634

	offnet_mou_7	offnet_mou_8	roam_ic_mou_6	roam_ic_mou_7	\
5704	0.307300	2.323745	-0.077655	-0.253231	
64892	-0.577717	-0.256061	0.022864	-0.253231	
39613	0.435043	1.465067	-0.254996	-0.253231	
93118	-0.143451	-0.410827	-0.254996	-0.253231	
81235	-0.290323	0.029332	-0.254996	-0.253231	

	roam_ic_mou_8	roam_og_mou_6	roam_og_mou_7	roam_og_mou_8	\
5704	-0.304660	0.215992	-0.374857	-0.431026	
64892	-0.304660	-0.120122	-0.374857	-0.431026	
39613	-0.304660	-0.300833	-0.374857	-0.431026	
93118	-0.304660	-0.300833	-0.374857	-0.431026	
81235	-0.003778	-0.300833	-0.374857	1.456232	

	loc_og_t2t_mou_6	loc_og_t2t_mou_7	loc_og_t2t_mou_8
loc_og_t2m_mou_6 \			
5704	-0.278217	-0.282623	-0.106758
0.028192			
64892	-0.278380	-0.302589	-0.174571 -
0.300150			

39613	0.254268	0.146234	0.514266	
2.795255				
93118	0.871759	1.002772	0.222587	
0.871444				
81235	2.888120	0.289221	1.362336	
0.767176				
loc_og_t2m_mou_7 loc_og_t2m_mou_8 loc_og_t2f_mou_6				
loc_og_t2f_mou_7 \				
5704	0.006336	0.034141	-0.087435	-
0.267401				
64892	-0.204014	-0.295881	-0.261886	-
0.267401				
39613	2.186811	3.743713	0.011714	-
0.076422				
93118	0.713384	-0.116066	1.669630	
1.311405				
81235	0.540001	0.742988	0.005438	-
0.267401				
loc_og_t2f_mou_8 loc_og_t2c_mou_6 loc_og_t2c_mou_7				
loc_og_t2c_mou_8 \				
5704	-0.244832	0.037799	-0.267368	-
0.244432				
64892	-0.244832	-0.191587	-0.267368	-
0.244432				
39613	1.174644	-0.191587	-0.267368	-
0.244432				
93118	0.642996	-0.191587	-0.267368	-
0.244432				
81235	-0.244832	-0.191587	-0.267368	-
0.244432				
loc_og_mou_6 loc_og_mou_7 loc_og_mou_8 std_og_t2t_mou_6 \				
5704	-0.161248	-0.195270	-0.055078	-0.610819
64892	-0.379084	-0.337876	-0.306653	-0.619956
39613	1.932970	1.438327	2.763270	-0.619956
93118	1.189608	1.165755	0.093205	-0.358067
81235	2.297311	0.506959	1.278083	3.237049
std_og_t2t_mou_7 std_og_t2t_mou_8 std_og_t2m_mou_6				
std_og_t2m_mou_7 \				
5704	-0.570510	-0.420186	0.346789	
0.369671				
64892	-0.570510	-0.415897	-0.231854	-
0.437192				
39613	-0.570510	-0.420186	-0.394991	-
0.343132				
93118	-0.342850	-0.221957	-0.507976	-
0.406765				

81235	1.462701	2.078469	-0.263825	-
0.435005				

	std_og_t2m_mou_8	std_og_t2f_mou_6	std_og_t2f_mou_7	
std_og_t2f_mou_8 \				
5704	2.702104	-0.143576	-0.139257	-
0.119299				
64892	-0.040526	-0.143576	-0.139257	-
0.104326				
39613	-0.177784	1.244575	-0.139257	-
0.119299				
93118	-0.346116	-0.143576	-0.139257	-
0.119299				
81235	-0.400887	-0.143576	-0.139257	-
0.119299				

	std_og_t2c_mou_6	std_og_t2c_mou_7	std_og_t2c_mou_8	
std_og_mou_6 \				
5704	0.0	0.0	0.0	-
0.214836				
64892	0.0	0.0	0.0	-
0.616620				
39613	0.0	0.0	0.0	-
0.708089				
93118	0.0	0.0	0.0	-
0.612400				
81235	0.0	0.0	0.0	
2.200139				

	std_og_mou_7	std_og_mou_8	isd_og_mou_6	isd_og_mou_7	
isd_og_mou_8 \					
5704	-0.152215	1.550482	-0.080803	-0.092449	-
0.061631					
64892	-0.714724	-0.306010	-0.080803	-0.092449	-
0.061631					
39613	-0.649149	-0.402193	-0.080803	-0.092449	-
0.061631					
93118	-0.530795	-0.384371	-0.080803	-0.092449	-
0.061631					
81235	0.739892	1.109859	-0.080803	-0.092449	-
0.061631					

	spl_og_mou_6	spl_og_mou_7	spl_og_mou_8	og_others_6	
og_others_7 \					
5704	1.055196	0.774917	0.757960	0.315218	-
0.015583					
64892	-0.327156	-0.363159	-0.290355	-0.346191	-
0.015583					
39613	-0.347585	-0.363159	-0.290355	-0.346191	-
0.015583					

93118	-0.278869	-0.293867	-0.231687	-0.025662	-
0.015583					
81235	0.049230	-0.363159	-0.290355	-0.346191	-
0.015583					

	og_others_8	total_og_mou_6	total_og_mou_7	total_og_mou_8	\
5704	-0.013735	-0.254350	-0.209855	1.354152	
64892	-0.013735	-0.775847	-0.845314	-0.422452	
39613	-0.013735	0.155438	-0.005238	0.943279	
93118	-0.013735	-0.077192	-0.008896	-0.300672	
81235	-0.013735	3.147976	0.919824	1.568307	

	loc_ic_t2t_mou_6	loc_ic_t2t_mou_7	loc_ic_t2t_mou_8	
loc_ic_t2m_mou_6	\			
5704	-0.356975	-0.095026	0.281846	
0.089162				
64892	-0.107944	-0.347607	-0.187444	
0.377903				
39613	0.075275	-0.307553	-0.130965	-
0.113096				
93118	2.234897	1.680142	1.178931	
0.874402				
81235	0.238993	0.429108	0.113725	
1.182362				

	loc_ic_t2m_mou_7	loc_ic_t2m_mou_8	loc_ic_t2f_mou_6	
loc_ic_t2f_mou_7	\			
5704	-0.112790	0.515971	-0.290528	-
0.270877				
64892	0.199498	0.240935	-0.275866	-
0.257495				
39613	-0.328115	0.017490	-0.104613	-
0.247057				
93118	0.796818	0.524427	-0.043033	
0.080535				
81235	0.691388	0.502602	-0.182615	-
0.060511				

	loc_ic_t2f_mou_8	loc_ic_mou_6	loc_ic_mou_7	loc_ic_mou_8	\
5704	-0.194257	-0.156095	-0.166424	0.468259	
64892	-0.235146	0.172870	-0.078726	0.045944	
39613	-0.125105	-0.056143	-0.419352	-0.067426	
93118	0.274772	1.723994	1.417516	0.968096	
81235	-0.257696	0.923322	0.685320	0.369639	

	std_ic_t2t_mou_6	std_ic_t2t_mou_7	std_ic_t2t_mou_8
std_ic_t2m_mou_6	\		
5704	-0.215496	-0.200464	-0.187265
0.113370			
64892	-0.215496	-0.152024	0.151031

2.985768				
39613	-0.215496	-0.200464	-0.187265	-
0.190865				
93118	-0.141573	-0.121450	-0.104402	-
0.219794				
81235	3.676291	2.238646	4.587345	-
0.261982				

	std_ic_t2m_mou_7	std_ic_t2m_mou_8	std_ic_t2f_mou_6	
std_ic_t2f_mou_7 \				
5704	-0.185210	-0.166335	-0.135479	-
0.137327				
64892	2.167834	2.203886	0.947838	
2.883004				
39613	-0.281392	-0.252874	0.271115	-
0.137327				
93118	-0.280534	-0.238753	-0.135479	-
0.137327				
81235	-0.280902	-0.232114	-0.135479	-
0.137327				

	std_ic_t2f_mou_8	std_ic_t2o_mou_6	std_ic_t2o_mou_7
std_ic_t2o_mou_8 \			
5704	-0.110642	0.0	0.0
0.0			
64892	1.082447	0.0	0.0
0.0			
39613	-0.110642	0.0	0.0
0.0			
93118	-0.110642	0.0	0.0
0.0			
81235	-0.110642	0.0	0.0
0.0			

	std_ic_mou_6	std_ic_mou_7	std_ic_mou_8	total_ic_mou_6 \
5704	-0.077912	-0.265421	-0.233610	-0.194148
64892	1.935296	1.671948	1.888954	1.033317
39613	-0.231969	-0.328225	-0.299535	-0.176324
93118	-0.250056	-0.277357	-0.247586	1.336498
81235	2.151705	1.225035	2.089855	1.680641

	total_ic_mou_7	total_ic_mou_8	spl_ic_mou_6	spl_ic_mou_7 \
5704	-0.204469	0.286255	-0.366516	-0.089786
64892	0.758037	0.716003	-0.366516	-0.089786
39613	-0.522456	-0.189128	-0.366516	-0.089786
93118	1.048813	0.704124	-0.366516	-0.089786
81235	1.061830	1.050754	-0.366516	-0.089786

	spl_ic_mou_8	isd_ic_mou_6	isd_ic_mou_7	isd_ic_mou_8
ic_others_6 \				

5704	-0.192624	-0.151655	0.285066	-0.126576	-
0.099745					
64892	-0.192624	0.312051	-0.021464	-0.126576	-
0.050008					
39613	-0.192624	-0.151655	-0.153778	-0.126576	-
0.099745					
93118	-0.192624	-0.047189	-0.153778	-0.126576	-
0.099745					
81235	-0.192624	-0.151655	-0.153778	-0.126576	-
0.099745					

	ic_others_7	ic_others_8	total_rech_num_6	total_rech_num_7	\
5704	-0.121704	-0.081491	-0.156412	0.260837	
64892	4.574367	0.366640	-0.040029	-0.680263	
39613	-0.121704	-0.011433	-0.738325	-1.033175	
93118	-0.121704	-0.081491	-0.738325	0.025562	
81235	-0.121704	-0.081491	1.472945	0.025562	

	total_rech_num_8	total_rech_amt_6	total_rech_amt_7	
total_rech_amt_8 \				
5704	1.306523	0.087587	-0.236774	
0.817300				
64892	0.054980	0.060452	-0.688801	-
0.051360				
39613	-0.695945	0.421336	-0.518626	
1.256352				
93118	-0.445637	1.126824	0.638030	-
0.511656				
81235	0.555597	3.764262	0.688551	
3.201396				

	max_rech_amt_6	max_rech_amt_7	max_rech_amt_8	
last_day_rch_amt_6 \				
5704	0.054992	-0.173563	0.029739	
0.175643				
64892	0.054992	0.358167	0.551737	-
0.598662				
39613	2.412584	2.340760	2.555285	
3.553548				
93118	1.183544	0.753166	0.950911	
1.530677				
81235	0.812513	0.008745	1.718554	
0.175643				

	last_day_rch_amt_7	last_day_rch_amt_8	vol_2g_mb_6
vol_2g_mb_7 \			
5704	0.368450	-0.351085	3.313695
2.175444			
64892	-0.350629	-0.626096	3.855666
6.784445			

39613	3.419926	3.040717	-0.245535	-
0.235847				
93118	1.493164	-0.626096	-0.245535	-
0.235847				
81235	-0.350629	1.207310	5.471850	
2.782323				

	vol_2g_mb_8	vol_3g_mb_6	vol_3g_mb_7	vol_3g_mb_8
monthly_2g_6 \				
5704	-0.098306	-0.262491	-0.063995	0.506232
3.236849				
64892	4.402555	1.716008	0.276844	0.288491
6.720348				
39613	-0.207939	-0.262491	-0.274601	-0.249913
0.246650				
93118	-0.207939	-0.262491	-0.274601	-0.249913
0.246650				
81235	1.989567	-0.262491	-0.274601	-0.249913
0.246650				

	monthly_2g_7	monthly_2g_8	sachet_2g_6	sachet_2g_7
sachet_2g_8 \				
5704	-0.251375	-0.232664	0.457379	2.358097
2.447476				
64892	3.104207	3.421905	-0.255793	-0.269796
2.447476				
39613	-0.251375	-0.232664	-0.255793	-0.269796
0.268245				
93118	-0.251375	-0.232664	-0.255793	-0.269796
0.268245				
81235	-0.251375	-0.232664	1.883722	1.044151
0.410685				

	monthly_3g_6	monthly_3g_7	monthly_3g_8	sachet_3g_6
sachet_3g_7 \				
5704	-0.224183	-0.221779	-0.216364	1.315163
1.219546				
64892	-0.224183	-0.221779	-0.216364	-0.141182
0.136208				
39613	-0.224183	-0.221779	-0.216364	-0.141182
0.136208				
93118	-0.224183	-0.221779	-0.216364	-0.141182
0.136208				
81235	2.171393	-0.221779	2.201160	2.771508
2.575301				

	sachet_3g_8	aon	aug_vbc_3g	jul_vbc_3g	jun_vbc_3g \
5704	2.526725	0.225051	0.018023	0.194794	-0.259366
64892	-0.113882	0.622516	2.423668	2.357564	5.861151
39613	-0.113882	2.966507	-0.255884	-0.265392	-0.259366

93118	-0.113882	1.742643	-0.255884	-0.265392	-0.259366
81235	1.206422	-0.244679	-0.255884	-0.265392	-0.259366

	decrease_mou_action	decrease_rech_num_action	\
5704	0	0	
64892	1	1	
39613	1	1	
93118	1	0	
81235	1	1	

	decrease_rech_amt_action	decrease_arpu_action
decrease_vbc_action		
5704	1	1
0		
64892	1	1
1		
39613	1	0
0		
93118	1	1
0		
81235	1	1
0		

Model with PCA

```
#Import PCA
from sklearn.decomposition import PCA

# Instantiate PCA
pca = PCA(random_state=42)

# Fit train set on PCA
pca.fit(X_train)

PCA(copy=True, iterated_power='auto', n_components=None,
    random_state=42,
    svd_solver='auto', tol=0.0, whiten=False)

# Principal components
pca.components_

array([[ -7.50315936e-20,  4.16333634e-17,  1.11022302e-16, ...,
        -2.59799614e-02, -2.57740516e-02,  1.40032998e-02],
       [-1.61507486e-19, -5.55111512e-17,  0.00000000e+00, ...,
        -1.16737642e-02, -9.94022864e-03, -1.42598315e-02],
       [ 1.91332162e-19, -2.77555756e-17,  0.00000000e+00, ...,
        -4.18532955e-02, -4.28357226e-02,  2.46812846e-02],
       ...,
       [-0.00000000e+00, -3.78694731e-02, -3.56427844e-02, ...,
```

```

1.23056947e-16, -4.06575815e-17, -0.00000000e+00],
[ 0.00000000e+00,  2.32804774e-01,  3.95374959e-02, ...,
 6.41847686e-17,  3.12250226e-17,  8.32667268e-17],
[ 9.9999199e-01, -3.85782335e-04,  1.19512948e-03, ...,
 1.35525272e-20,  3.11708125e-19, -1.99086624e-17]])

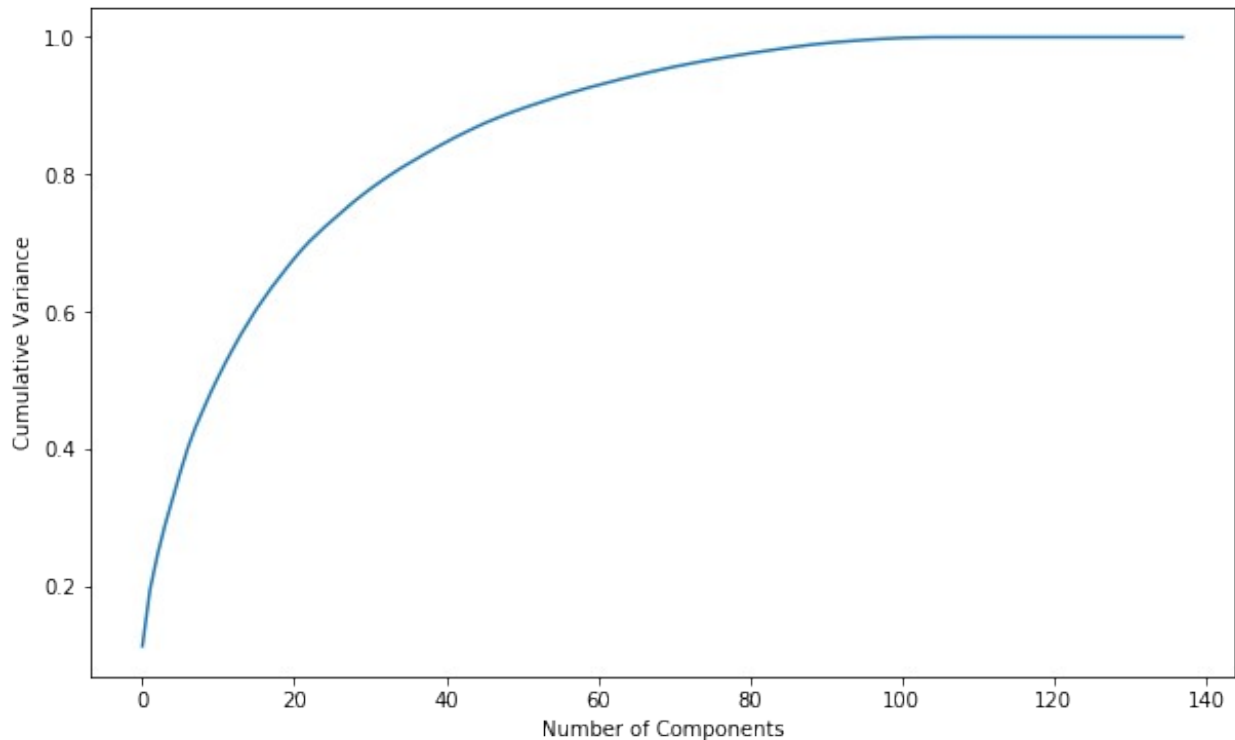
# Cumulative varinace of the PCs
variance_cumu = np.cumsum(pca.explained_variance_ratio_)
print(variance_cumu)

[0.11213256 0.19426234 0.24575583 0.28953571 0.32841891 0.36623473
 0.40173361 0.43144425 0.45702167 0.48194328 0.50480575 0.52673812
 0.54724457 0.5670202  0.58530008 0.60304258 0.6190213  0.63473458
 0.64927873 0.66341423 0.67712828 0.69025011 0.7020618  0.71278516
 0.72309435 0.73290234 0.74255604 0.75209676 0.76151565 0.77010093
 0.77861315 0.7866115  0.79429496 0.80173555 0.80878909 0.81538157
 0.82193734 0.8283476  0.83472622 0.84089758 0.84687761 0.85280024
 0.85840083 0.86374029 0.86901646 0.87418749 0.87891437 0.88341796
 0.887723    0.89186057 0.89588256 0.89966074 0.90339384 0.90704071
 0.91060084 0.91411689 0.91752343 0.92076319 0.92395413 0.92705111
 0.93001239 0.93296077 0.93580029 0.93862291 0.94138851 0.9441162
 0.94678675 0.94937767 0.95188405 0.95433786 0.95665036 0.95893735
 0.96116409 0.96323063 0.96526039 0.967203    0.96912626 0.97100138
 0.97284931 0.9746657  0.97639261 0.97806622 0.97972617 0.98133794
 0.98290963 0.98446566 0.98601222 0.98753485 0.98877905 0.98998795
 0.99114751 0.99224606 0.99321228 0.99407803 0.9949224  0.99573799
 0.99652652 0.99717502 0.99776401 0.99831985 0.99880793 0.99912289
 0.99942656 0.99969174 0.99985313 0.99994737 0.99998103 0.99999839
 0.99999963 0.99999989 1.         1.         1.         1.
 1.         1.         1.         1.         1.         1.
 1.         1.         1.         1.         1.         1.
 1.         1.         1.         1.         1.         1.
 1.         1.         1.         1.         1.         1.]

# Plotting scree plot
fig = plt.figure(figsize = (10,6))
plt.plot(variance_cumu)
plt.xlabel('Number of Components')
plt.ylabel('Cumulative Variance')

Text(0, 0.5, 'Cumulative Variance')

```



We can see that 60 components explain almost more than 90% variance of the data. So, we will perform PCA with 60 components.

Performing PCA with 60 components

```
# Importing incremental PCA
from sklearn.decomposition import IncrementalPCA

# Instantiate PCA with 60 components
pca_final = IncrementalPCA(n_components=60)

# Fit and transform the X_train
X_train_pca = pca_final.fit_transform(X_train)
```

Applying transformation on the test set

We are only doing Transform in the test set not the Fit-Transform. Because the Fitting is already done on the train set. So, we just have to do the transformation with the already fitted data on the train set.

```
X_test_pca = pca_final.transform(X_test)
```

Emphasize Sensitivity/Recall than Accuracy

We are more focused on higher Sensitivity/Recall score than the accuracy.

Because we need to care more about churn cases than the not churn cases. The main goal is to retain the customers, who have the possibility to churn. There should not be a problem, if we

consider few not churn customers as churn customers and provide them some incentives for retaining them. Hence, the sensitivity score is more important here.

Logistic regression with PCA

```
# Importing scikit logistic regression module
from sklearn.linear_model import LogisticRegression

# Impoting metrics
from sklearn import metrics
from sklearn.metrics import confusion_matrix
```

Tuning hyperparameter C

C is the the inverse of regularization strength in Logistic Regression. Higher values of C correspond to less regularization.

```
# Importing libraries for cross validation
from sklearn.model_selection import KFold
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import GridSearchCV

# Creating KFold object with 5 splits
folds = KFold(n_splits=5, shuffle=True, random_state=4)

# Specify params
params = {"C": [0.01, 0.1, 1, 10, 100, 1000]}

# Specifying score as recall as we are more focused on achieving the
higher sensitivity than the accuracy
model_cv = GridSearchCV(estimator = LogisticRegression(),
                        param_grid = params,
                        scoring= 'recall',
                        cv = folds,
                        verbose = 1,
                        return_train_score=True)
```

```
# Fit the model
model cv.fit(X train pca, y train)
```

Fitting 5 folds for each of 6 candidates, totalling 30 fits

```
[Parallel(n_jobs=1)]: Using backend SequentialBackend with 1
concurrent workers.
```

```
[Parallel(n jobs=1)]: Done 30 out of 30 | elapsed: 21.6s finished
```

```
GridSearchCV(cv=KFold(n_splits=5, random_state=4, shuffle=True),
             error_score=nan,
             estimator=LogisticRegression(C=1.0, class_weight=None,
             dual=False, fit_intercept=True,
```

```

l1_ratio=None, intercept_scaling=1,
max_iter=100,
multi_class='auto', n_jobs=None, penalty='l2',
random_state=None,
solver='lbfgs', tol=0.0001, verbose=0,
warm_start=False),
iid='deprecated', n_jobs=None,
param_grid={'C': [0.01, 0.1, 1, 10, 100, 1000]},
pre_dispatch='2*n_jobs', refit=True,
return_train_score=True,
scoring='recall', verbose=1)

```

results of grid search CV

```

cv_results = pd.DataFrame(model_cv.cv_results_)
cv_results

```

	mean_fit_time	std_fit_time	mean_score_time	std_score_time
param_C \				
0 0.01	0.478627	0.060932	0.007600	1.200167e-03
1 0.1	0.731842	0.021868	0.006801	3.999949e-04
2 1	0.743043	0.008100	0.007000	6.325605e-04
3 10	0.754643	0.024106	0.007200	1.469782e-03
4 100	0.720841	0.015716	0.007000	1.784161e-07
5 1000	0.719441	0.008778	0.006600	4.899208e-04

	params	split0_test_score	split1_test_score
split2_test_score \			
0 {'C': 0.01}		0.900071	0.897759
0.895814			
1 {'C': 0.1}		0.898177	0.896359
0.894651			
2 {'C': 1}		0.898650	0.898693
0.895581			
3 {'C': 10}		0.898887	0.898459
0.896744			
4 {'C': 100}		0.899597	0.898226
0.896977			
5 {'C': 1000}		0.899597	0.898226
0.896977			

split3_test_score	split4_test_score	mean_test_score
-------------------	-------------------	-----------------

	std_test_score \		
0	0.906425	0.887552	0.897524
0.006134			
1	0.905959	0.889403	0.896910
0.005390			
2	0.905028	0.890329	0.897656
0.004783			
3	0.904562	0.889866	0.897704
0.004719			
4	0.904330	0.890329	0.897892
0.004528			
5	0.904330	0.890329	0.897892
0.004528			

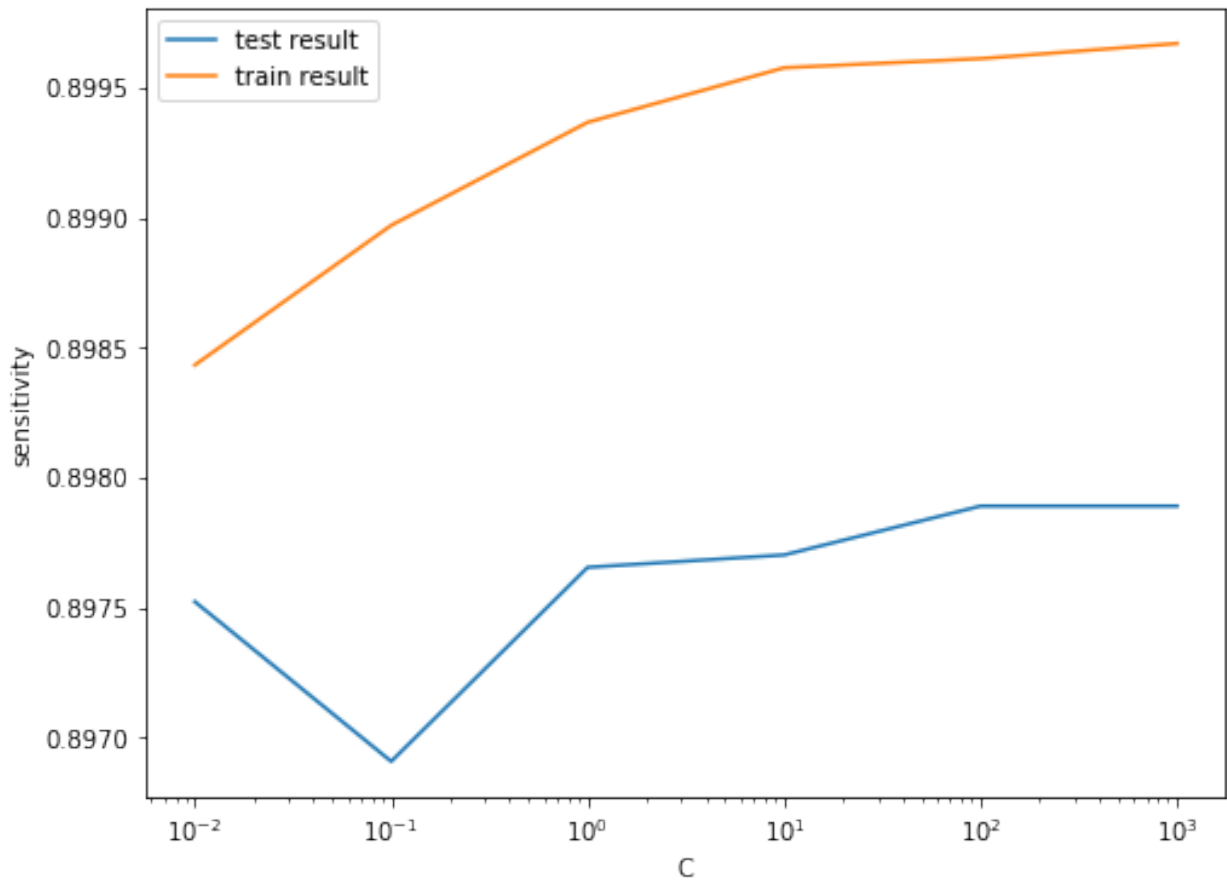
	rank_test_score	split0_train_score	split1_train_score \
0	5	0.901116	0.898256
1	6	0.901174	0.898431
2	4	0.901988	0.898606
3	3	0.902511	0.898956
4	1	0.902628	0.898722
5	1	0.902628	0.898839

	split2_train_score	split3_train_score	split4_train_score \
0	0.899387	0.895440	0.897971
1	0.899270	0.896725	0.899257
2	0.898861	0.898184	0.899199
3	0.898394	0.898476	0.899550
4	0.898569	0.898593	0.899550
5	0.898686	0.898593	0.899608

	mean_train_score	std_train_score
0	0.898434	0.001861
1	0.898971	0.001440
2	0.899368	0.001351
3	0.899577	0.001524
4	0.899612	0.001550
5	0.899671	0.001521

plot of C versus train and validation scores

```
plt.figure(figsize=(8, 6))
plt.plot(cv_results['param_C'], cv_results['mean_test_score'])
plt.plot(cv_results['param_C'], cv_results['mean_train_score'])
plt.xlabel('C')
plt.ylabel('sensitivity')
plt.legend(['test result', 'train result'], loc='upper left')
plt.xscale('log')
```

```
# Best score with best C
best_score = model_cv.best_score_
best_C = model_cv.best_params_['C']

print(" The highest test sensitivity is {0} at C =
{1}").format(best_score, best_C)
```

The highest test sensitivity is 0.8978916608693863 at C = 100

Logistic regression with optimal C

```
# Instantiate the model with best C
logistic_pca = LogisticRegression(C=best_C)

# Fit the model on the train set
log_pca_model = logistic_pca.fit(X_train_pca, y_train)
```

Prediction on the train set

```
# Predictions on the train set
y_train_pred = log_pca_model.predict(X_train_pca)
```

```

# Confusion matrix
confusion = metrics.confusion_matrix(y_train, y_train_pred)
print(confusion)

[[17908  3517]
 [ 2154 19271]]

TP = confusion[1,1] # true positive
TN = confusion[0,0] # true negatives
FP = confusion[0,1] # false positives
FN = confusion[1,0] # false negatives

# Accuracy
print("Accuracy:-",metrics.accuracy_score(y_train, y_train_pred))

# Sensitivity
print("Sensitivity:-",TP / float(TP+FN))

# Specificity
print("Specificity:-", TN / float(TN+FP))

Accuracy:- 0.8676546091015169
Sensitivity:- 0.899463243873979
Specificity:- 0.8358459743290548

```

Prediction on the test set

```

# Prediction on the test set
y_test_pred = log_pca_model.predict(X_test_pca)

# Confusion matrix
confusion = metrics.confusion_matrix(y_test, y_test_pred)
print(confusion)

[[4452  896]
 [  36 157]]

TP = confusion[1,1] # true positive
TN = confusion[0,0] # true negatives
FP = confusion[0,1] # false positives
FN = confusion[1,0] # false negatives

# Accuracy
print("Accuracy:-",metrics.accuracy_score(y_test, y_test_pred))

# Sensitivity
print("Sensitivity:-",TP / float(TP+FN))

# Specificity
print("Specificity:-", TN / float(TN+FP))

```

```
Accuracy:- 0.8317993142032124
Sensitivity:- 0.8134715025906736
Specificity:- 0.8324607329842932
```

Model summary

- Train set
 - Accuracy = 0.86
 - Sensitivity = 0.89
 - Specificity = 0.83
- Test set
 - Accuracy = 0.83
 - Sensitivity = 0.81
 - Specificity = 0.83

Overall, the model is performing well in the test set, what it had learnt from the train set.

Support Vector Machine(SVM) with PCA

```
# Importing SVC
from sklearn.svm import SVC
```

Hyperparameter tuning

C:- Regularization parameter.

gamma:- Handles non linear classifications.

```
# specify range of hyperparameters

hyper_params = [ {'gamma': [1e-2, 1e-3, 1e-4],
                    'C': [1, 10, 100, 1000]}]

# specify model with RBF kernel
model = SVC(kernel="rbf")

# set up GridSearchCV()
model_cv = GridSearchCV(estimator = model,
                        param_grid = hyper_params,
                        scoring= 'accuracy',
                        cv = 3,
                        verbose = 1,
                        return_train_score=True)

# fit the model
model_cv.fit(X_train_pca, y_train)
```

Fitting 3 folds for each of 12 candidates, totalling 36 fits

```
[Parallel(n_jobs=1)]: Using backend SequentialBackend with 1
concurrent workers.
[Parallel(n_jobs=1)]: Done 36 out of 36 | elapsed: 95.4min finished
```

```
GridSearchCV(cv=3, error_score=nan,
             estimator=SVC(C=1.0, break_ties=False, cache_size=200,
                           class_weight=None, coef0=0.0,
                           decision_function_shape='ovr', degree=3,
                           gamma='scale', kernel='rbf', max_iter=-1,
                           probability=False, random_state=None,
                           shrinking=True,
                           tol=0.001, verbose=False),
             iid='deprecated', n_jobs=None,
             param_grid=[{'C': [1, 10, 100, 1000],
                           'gamma': [0.01, 0.001, 0.0001]}],
             pre_dispatch='2*n_jobs', refit=True,
             return_train_score=True,
             scoring='accuracy', verbose=1)
```

```
# cv results
```

```
cv_results = pd.DataFrame(model_cv.cv_results_)
cv_results
```

	mean_fit_time	std_fit_time	mean_score_time	std_score_time
param_C \				
0	42.032071	0.126973	12.777397	0.027355
1				
1	53.846413	0.407173	17.470999	0.045588
1				
2	67.914884	0.491901	22.857974	0.176785
1				
3	39.426255	0.483549	8.479152	0.292587
10				
4	47.964410	0.411481	14.921520	0.169904
10				
5	57.650298	0.850001	19.027755	0.122431
10				
6	57.674299	6.235520	670.884914	941.456985
100				
7	56.490898	0.578504	11.192307	0.035219
100				
8	54.180099	0.697826	16.342601	0.187994
100				
9	93.312670	2.057615	3.813552	0.053278
1000				
10	127.868314	3.560431	8.287141	0.121411
1000				
11	79.973241	1.123954	14.635170	0.255735
1000				

	param_gamma	params	split0_test_score	\
0	0.01	{'C': 1, 'gamma': 0.01}	0.944903	
1	0.001	{'C': 1, 'gamma': 0.001}	0.883366	
2	0.0001	{'C': 1, 'gamma': 0.0001}	0.858513	
3	0.01	{'C': 10, 'gamma': 0.01}	0.967096	
4	0.001	{'C': 10, 'gamma': 0.001}	0.910459	
5	0.0001	{'C': 10, 'gamma': 0.0001}	0.870414	
6	0.01	{'C': 100, 'gamma': 0.01}	0.973397	
7	0.001	{'C': 100, 'gamma': 0.001}	0.935662	
8	0.0001	{'C': 100, 'gamma': 0.0001}	0.884906	
9	0.01	{'C': 1000, 'gamma': 0.01}	0.972277	
10	0.001	{'C': 1000, 'gamma': 0.001}	0.955965	
11	0.0001	{'C': 1000, 'gamma': 0.0001}	0.908499	
	split1_test_score	split2_test_score	mean_test_score	std_test_score
0	0.941679	0.940699	0.942427	0.001796
1	0.884268	0.884058	0.883897	0.000385
2	0.858433	0.859133	0.858693	0.000313
3	0.965413	0.965063	0.965858	0.000887
4	0.911433	0.908073	0.909988	0.001412
5	0.869355	0.872786	0.870852	0.001434
6	0.976686	0.975775	0.975286	0.001387
7	0.935518	0.934608	0.935263	0.000467
8	0.886438	0.886718	0.886021	0.000797
9	0.977876	0.976336	0.975496	0.002362
10	0.955121	0.955472	0.955519	0.000346
11	0.910943	0.907302	0.908915	0.001515
	rank_test_score	split0_train_score	split1_train_score	\
0	5	0.947210	0.947247	
1	10	0.883813	0.886757	
2	12	0.858993	0.859908	
3	3	0.975040	0.973536	
4	7	0.913709	0.911891	
5	11	0.871421	0.873875	
6	2	0.991318	0.990444	

7	6	0.941819	0.942066
8	9	0.886368	0.888893
9	1	0.998425	0.998495
10	4	0.965623	0.965345
11	8	0.912868	0.910806

	split2_train_score	mean_train_score	std_train_score
0	0.947702	0.947386	0.000224
1	0.885707	0.885426	0.001218
2	0.859173	0.859358	0.000396
3	0.974306	0.974294	0.000614
4	0.912381	0.912660	0.000768
5	0.870690	0.871995	0.001362
6	0.990198	0.990653	0.000481
7	0.941681	0.941855	0.000159
8	0.888543	0.887935	0.001117
9	0.998495	0.998471	0.000033
10	0.966465	0.965811	0.000476
11	0.911996	0.911890	0.000845

Plotting the accuracy with various C and gamma values

```
# converting C to numeric type for plotting on x-axis
cv_results['param_C'] = cv_results['param_C'].astype('int')

# # plotting
plt.figure(figsize=(16,6))

# subplot 1/3
plt.subplot(131)
gamma_01 = cv_results[cv_results['param_gamma']==0.01]

plt.plot(gamma_01["param_C"], gamma_01["mean_test_score"])
plt.plot(gamma_01["param_C"], gamma_01["mean_train_score"])
plt.xlabel('C')
plt.ylabel('Accuracy')
plt.title("Gamma=0.01")
plt.ylim([0.80, 1])
plt.legend(['test accuracy', 'train accuracy'], loc='upper left')
plt.xscale('log')

# subplot 2/3
plt.subplot(132)
gamma_001 = cv_results[cv_results['param_gamma']==0.001]

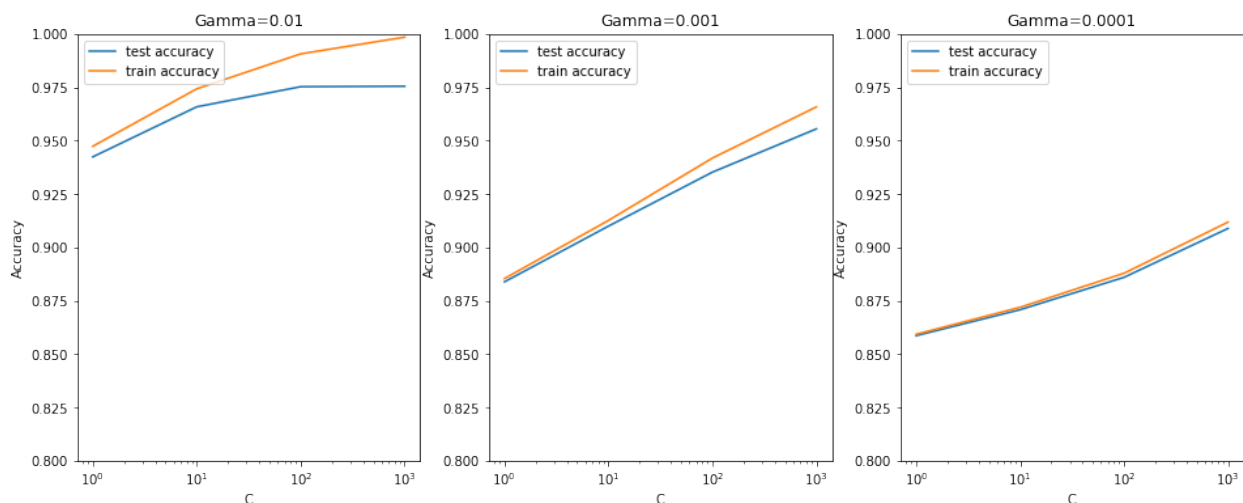
plt.plot(gamma_001["param_C"], gamma_001["mean_test_score"])
plt.plot(gamma_001["param_C"], gamma_001["mean_train_score"])
plt.xlabel('C')
plt.ylabel('Accuracy')
plt.title("Gamma=0.001")
```

```
plt.ylim([0.80, 1])
plt.legend(['test accuracy', 'train accuracy'], loc='upper left')
plt.xscale('log')
```

subplot 3/3

```
plt.subplot(133)
gamma_0001 = cv_results[cv_results['param_gamma']==0.0001]

plt.plot(gamma_0001["param_C"], gamma_0001["mean_test_score"])
plt.plot(gamma_0001["param_C"], gamma_0001["mean_train_score"])
plt.xlabel('C')
plt.ylabel('Accuracy')
plt.title("Gamma=0.0001")
plt.ylim([0.80, 1])
plt.legend(['test accuracy', 'train accuracy'], loc='upper left')
plt.xscale('log')
```



Printing the best score

```
best_score = model_cv.best_score_
best_hyperparams = model_cv.best_params_
```

```
print("The best test score is {0} corresponding to hyperparameters  
{1}".format(best_score, best_hyperparams))
```

```
The best test score is 0.9754959911159373 corresponding to  
hyperparameters {'C': 1000, 'gamma': 0.01}
```

From the above plot, we can see that higher value of gamma leads to overfitting the model. With the lowest value of gamma (0.0001) we have train and test accuracy almost same.

Also, at C=100 we have a good accuracy and the train and test scores are comparable.

Though sklearn suggests the optimal scores mentioned above ($\gamma=0.01$, $C=1000$), one could argue that it is better to choose a simpler, more non-linear model with $\gamma=0.0001$. This is because the optimal values mentioned here are calculated based on the average test accuracy (but not considering subjective parameters such as model complexity).

We can achieve comparable average test accuracy (~90%) with $\gamma=0.0001$ as well, though we'll have to increase the cost C for that. So to achieve high accuracy, there's a tradeoff between:

- High γ (i.e. high non-linearity) and average value of C
- Low γ (i.e. less non-linearity) and high value of C

We argue that the model will be simpler if it has as less non-linearity as possible, so we choose $\gamma=0.0001$ and a high $C=100$.

Build the model with optimal hyperparameters

```
# Building the model with optimal hyperparameters
svm_pca_model = SVC(C=100, gamma=0.0001, kernel="rbf")

svm_pca_model.fit(X_train_pca, y_train)

SVC(C=100, break_ties=False, cache_size=200, class_weight=None,
    coef0=0.0,
    decision_function_shape='ovr', degree=3, gamma=0.0001,
    kernel='rbf',
    max_iter=-1, probability=False, random_state=None, shrinking=True,
    tol=0.001, verbose=False)
```

Prediction on the train set

```
# Predictions on the train set
y_train_pred = svm_pca_model.predict(X_train_pca)

# Confusion matrix
confusion = metrics.confusion_matrix(y_train, y_train_pred)
print(confusion)

[[18376  3049]
 [ 1585 19840]]

TP = confusion[1,1] # true positive
TN = confusion[0,0] # true negatives
FP = confusion[0,1] # false positives
FN = confusion[1,0] # false negatives

# Accuracy
print("Accuracy:-", metrics.accuracy_score(y_train, y_train_pred))

# Sensitivity
print("Sensitivity:-", TP / float(TP+FN))
```



```
# Specificity
print("Specificity:-", TN / float(TN+FP))
```

```
Accuracy:- 0.891855309218203
Sensitivity:- 0.9260210035005835
Specificity:- 0.8576896149358226
```

Prediction on the test set

```
# Prediction on the test set
y_test_pred = svm_pca_model.predict(X_test_pca)
```

```
# Confusion matrix
confusion = metrics.confusion_matrix(y_test, y_test_pred)
print(confusion)
```

```
[[4557  791]
 [  36  157]]
```

```
TP = confusion[1,1] # true positive
TN = confusion[0,0] # true negatives
FP = confusion[0,1] # false positives
FN = confusion[1,0] # false negatives
```

```
# Accuracy
print("Accuracy:-", metrics.accuracy_score(y_test, y_test_pred))
```

```
# Sensitivity
print("Sensitivity:-", TP / float(TP+FN))
```

```
# Specificity
print("Specificity:-", TN / float(TN+FP))
```

```
Accuracy:- 0.8507489622811767
Sensitivity:- 0.8134715025906736
Specificity:- 0.8520942408376964
```

Model summary

- Train set
 - Accuracy = 0.89
 - Sensitivity = 0.92
 - Specificity = 0.85
- Test set
 - Accuracy = 0.85
 - Sensitivity = 0.81
 - Specificity = 0.85

Decision tree with PCA

```
# Importing decision tree classifier
from sklearn.tree import DecisionTreeClassifier
```

Hyperparameter tuning

```
# Create the parameter grid
param_grid = {
    'max_depth': range(5, 15, 5),
    'min_samples_leaf': range(50, 150, 50),
    'min_samples_split': range(50, 150, 50),
}

# Instantiate the grid search model
dtree = DecisionTreeClassifier()

grid_search = GridSearchCV(estimator = dtree,
                           param_grid = param_grid,
                           scoring= 'recall',
                           cv = 5,
                           verbose = 1)
```

```
# Fit the grid search to the data
grid_search.fit(X_train_pca,y_train)
```

Fitting 5 folds for each of 8 candidates, totalling 40 fits

[Parallel(n_jobs=1)]: Using backend SequentialBackend with 1 concurrent workers.

[Parallel(n_jobs=1)]: Done 40 out of 40 | elapsed: 1.7min finished

```
GridSearchCV(cv=5, error_score=nan,
             estimator=DecisionTreeClassifier(ccp_alpha=0.0,
             class_weight=None,
             criterion='gini',
             max_depth=None,
             max_features=None,
             max_leaf_nodes=None,
             min_impurity_decrease=0.0,
             min_impurity_split=None,
             min_samples_leaf=1,
             min_samples_split=2,
             min_weight_fraction_leaf=0.0,
             presort='deprecated',
             random_state=None,
             splitter='best'),
             iid='deprecated', n_jobs=None,
```

```

        param_grid={'max_depth': range(5, 15, 5),
                    'min_samples_leaf': range(50, 150, 50),
                    'min_samples_split': range(50, 150, 50)},
        pre_dispatch='2*n_jobs', refit=True,
return_train_score=False,
        scoring='recall', verbose=1)

```

cv results

```

cv_results = pd.DataFrame(grid_search.cv_results_)
cv_results

```

	mean_fit_time	std_fit_time	mean_score_time	std_score_time	\
0	1.948911	0.023829	0.008601	0.00049	
1	1.941111	0.010277	0.008400	0.00049	
2	1.925310	0.003188	0.008400	0.00049	
3	1.925510	0.002871	0.008600	0.00049	
4	3.343991	0.015459	0.008601	0.00049	
5	3.370193	0.083993	0.008601	0.00049	
6	3.199783	0.044874	0.008801	0.00040	
7	3.186582	0.025967	0.008801	0.00040	

	param_max_depth	param_min_samples_leaf	param_min_samples_split	\
0	5	50	50	
1	5	50	100	
2	5	100	50	
3	5	100	100	
4	10	50	50	
5	10	50	100	
6	10	100	50	
7	10	100	100	

	split0_test_score	\	params
0	0.862310		{'max_depth': 5, 'min_samples_leaf': 50, 'min...
1	0.862310		{'max_depth': 5, 'min_samples_leaf': 50, 'min...
2	0.858110		{'max_depth': 5, 'min_samples_leaf': 100, 'min...
3	0.858110		{'max_depth': 5, 'min_samples_leaf': 100, 'min...
4	0.886114		{'max_depth': 10, 'min_samples_leaf': 50, 'min...
5	0.886114		{'max_depth': 10, 'min_samples_leaf': 50, 'min...
6	0.889615		{'max_depth': 10, 'min_samples_leaf': 100, 'mi...
7	0.889615		{'max_depth': 10, 'min_samples_leaf': 100, 'mi...


```

                                max_depth=10,
                                min_samples_leaf=50,
                                min_samples_split=50)

dt_pca_model.fit(X_train_pca, y_train)

DecisionTreeClassifier(ccp_alpha=0.0, class_weight=None,
criterion='gini',
                        max_depth=10, max_features=None,
max_leaf_nodes=None,
                        min_impurity_decrease=0.0,
min_impurity_split=None,
                        min_samples_leaf=50, min_samples_split=50,
                        min_weight_fraction_leaf=0.0,
presort='deprecated',
                        random_state=100, splitter='best')

```

Prediction on the train set

```

# Predictions on the train set
y_train_pred = dt_pca_model.predict(X_train_pca)

# Confusion matrix
confusion = metrics.confusion_matrix(y_train, y_train_pred)
print(confusion)

[[18913  2512]
 [ 1763 19662]]

TP = confusion[1,1] # true positive
TN = confusion[0,0] # true negatives
FP = confusion[0,1] # false positives
FN = confusion[1,0] # false negatives

# Accuracy
print("Accuracy:-",metrics.accuracy_score(y_train, y_train_pred))

# Sensitivity
print("Sensitivity:-",TP / float(TP+FN))

# Specificity
print("Specificity:-", TN / float(TN+FP))

Accuracy:- 0.9002333722287048
Sensitivity:- 0.9177129521586931
Specificity:- 0.8827537922987164

```

Prediction on the test set

```

# Prediction on the test set
y_test_pred = dt_pca_model.predict(X_test_pca)

```

```

# Confusion matrix
confusion = metrics.confusion_matrix(y_test, y_test_pred)
print(confusion)

[[4632  716]
 [  58 135]]

TP = confusion[1,1] # true positive
TN = confusion[0,0] # true negatives
FP = confusion[0,1] # false positives
FN = confusion[1,0] # false negatives

# Accuracy
print("Accuracy:-", metrics.accuracy_score(y_test, y_test_pred))

# Sensitivity
print("Sensitivity:-", TP / float(TP+FN))

# Specificity
print("Specificity:-", TN / float(TN+FP))

Accuracy:- 0.8603140227395777
Sensitivity:- 0.6994818652849741
Specificity:- 0.8661181750186986

```

Model summary

- Train set
 - Accuracy = 0.90
 - Sensitivity = 0.91
 - Specificity = 0.88
- Test set
 - Accuracy = 0.86
 - Sensitivity = 0.70
 - Specificity = 0.87

We can see from the model performance that the Sensitivity has been decreased while evaluating the model on the test set. However, the accuracy and specificity is quite good in the test set.

Random forest with PCA

```

# Importing random forest classifier
from sklearn.ensemble import RandomForestClassifier

```

Hyperparameter tuning

```

param_grid = {
    'max_depth': range(5, 10, 5),
    'min_samples_leaf': range(50, 150, 50),
    'min_samples_split': range(50, 150, 50),

```

```

    'n_estimators': [100, 200, 300],
    'max_features': [10, 20]
}
# Create a based model
rf = RandomForestClassifier()
# Instantiate the grid search model
grid_search = GridSearchCV(estimator = rf,
                           param_grid = param_grid,
                           cv = 3,
                           n_jobs = -1,
                           verbose = 1,
                           return_train_score=True)

# Fit the model
grid_search.fit(X_train_pca, y_train)

Fitting 3 folds for each of 24 candidates, totalling 72 fits

[Parallel(n_jobs=-1)]: Using backend LokyBackend with 4 concurrent
workers.
[Parallel(n_jobs=-1)]: Done 42 tasks      | elapsed: 13.6min
[Parallel(n_jobs=-1)]: Done 72 out of 72 | elapsed: 132.9min
finished

GridSearchCV(cv=3, error_score=nan,
             estimator=RandomForestClassifier(bootstrap=True,
             ccp_alpha=0.0,
             class_weight=None,
             criterion='gini',
             max_depth=None,
             max_features='auto',
             max_leaf_nodes=None,
             max_samples=None,
             min_impurity_decrease=0.0,
             min_impurity_split=None,
             min_samples_leaf=1,
             min_samples_split=2,
             min_weight_fraction_leaf=0.0,
             n_estimators=100,
             n_jobs=None,
             oob_score=False,
             random_state=None,
             verbose=0,
             warm_start=False),
             iid='deprecated', n_jobs=-1,
             param_grid={'max_depth': range(5, 10, 5), 'max_features':
[10, 20],
             'min_samples_leaf': range(50, 150, 50),

```

```

        'min_samples_split': range(50, 150, 50),
        'n_estimators': [100, 200, 300]},
        pre_dispatch='2*n_jobs', refit=True,
return_train_score=True,
        scoring=None, verbose=1)

# printing the optimal accuracy score and hyperparameters
print('We can get accuracy
of', grid_search.best_score_, 'using', grid_search.best_params_)

We can get accuracy of 0.8449241538567582 using {'max_depth': 5,
'max_features': 20, 'min_samples_leaf': 50, 'min_samples_split': 100,
'n_estimators': 300}

```

Model with optimal hyperparameters

```

# model with the best hyperparameters

rfc_model = RandomForestClassifier(bootstrap=True,
                                max_depth=5,
                                min_samples_leaf=50,
                                min_samples_split=100,
                                max_features=20,
                                n_estimators=300)

# Fit the model
rfc_model.fit(X_train_pca, y_train)

RandomForestClassifier(bootstrap=True, ccp_alpha=0.0,
class_weight=None,
                        criterion='gini', max_depth=5, max_features=20,
                        max_leaf_nodes=None, max_samples=None,
                        min_impurity_decrease=0.0,
min_impurity_split=None,
                        min_samples_leaf=50, min_samples_split=100,
                        min_weight_fraction_leaf=0.0, n_estimators=300,
                        n_jobs=None, oob_score=False,
random_state=None,
                        verbose=0, warm_start=False)

```

Prediction on the train set

```

# Predictions on the train set
y_train_pred = rfc_model.predict(X_train_pca)

# Confusion matrix
confusion = metrics.confusion_matrix(y_train, y_train_pred)
print(confusion)

[[17363  4062]
 [ 2419 19006]]

```



```

TP = confusion[1,1] # true positive
TN = confusion[0,0] # true negatives
FP = confusion[0,1] # false positives
FN = confusion[1,0] # false negatives

# Accuracy
print("Accuracy:-", metrics.accuracy_score(y_train, y_train_pred))

# Sensitivity
print("Sensitivity:-", TP / float(TP+FN))

# Specificity
print("Specificity:-", TN / float(TN+FP))

Accuracy:- 0.8487514585764294
Sensitivity:- 0.8870945157526254
Specificity:- 0.8104084014002334

```

Prediction on the test set

```

# Prediction on the test set
y_test_pred = rfc_model.predict(X_test_pca)

# Confusion matrix
confusion = metrics.confusion_matrix(y_test, y_test_pred)
print(confusion)

[[4294 1054]
 [ 47  146]]

TP = confusion[1,1] # true positive
TN = confusion[0,0] # true negatives
FP = confusion[0,1] # false positives
FN = confusion[1,0] # false negatives

# Accuracy
print("Accuracy:-", metrics.accuracy_score(y_test, y_test_pred))

# Sensitivity
print("Sensitivity:-", TP / float(TP+FN))

# Specificity
print("Specificity:-", TN / float(TN+FP))

Accuracy:- 0.8012994044396319
Sensitivity:- 0.7564766839378239
Specificity:- 0.8029169783096485

```

Model summary

- Train set
 - Accuracy = 0.84

- Sensitivity = 0.88
 - Specificity = 0.80
- Test set
 - Accuracy = 0.80
 - Sensitivity = 0.75
 - Specificity = 0.80

We can see from the model performance that the Sensitivity has been decreased while evaluating the model on the test set. However, the accuracy and specificity is quite good in the test set.

Final conclusion with PCA

After trying several models we can see that for achieving the best sensitivity, which was our ultimate goal, the classic Logistic regression or the SVM models performs well. For both the models the sensitivity was approx 81%. Also we have good accuracy of approx 85%.

Without PCA

Logistic regression with No PCA

```
##### Importing stats model
import statsmodels.api as sm

# Instantiate the model
# Adding the constant to X_train
log_no_pca = sm.GLM(y_train, (sm.add_constant(X_train)),
family=sm.families.Binomial())

# Fit the model
log_no_pca = log_no_pca.fit().summary()

# Summary
log_no_pca
```

```
<class 'statsmodels.iolib.summary.Summary'>
"""
                                Generalized Linear Model Regression Results
=====
=====
Dep. Variable:                  churn    No. Observations:
42850
Model:                          GLM      Df Residuals:
42720
Model Family:                   Binomial  Df Model:
129
Link Function:                   logit     Scale:
1.0000
```

Method: IRLS Log-Likelihood:
nan
Date: Sat, 16 May 2020 Deviance:
nan
Time: 17:56:38 Pearson chi2:
3.70e+05
No. Iterations: 100
Covariance Type: nonrobust

		coef	std err	z	P> z
[0.025 0.975]					

const		-73.1903	4452.100	-0.016	0.987
-8799.145	8652.765				
loc_og_t2o_mou		7.163e-07	4.32e-05	0.017	0.987
-8.39e-05	8.53e-05				
std_og_t2o_mou		2.572e-07	1.42e-05	0.018	0.986
-2.75e-05	2.81e-05				
loc_ic_t2o_mou		1.441e-06	6.09e-05	0.024	0.981
-0.000	0.000				
arpu_6		-0.0338	0.081	-0.418	0.676
-0.192	0.124				
arpu_7		0.0855	0.086	0.994	0.320
-0.083	0.254				
arpu_8		0.0909	0.110	0.828	0.407
-0.124	0.306				
onnet_mou_6		15.5129	3.573	4.342	0.000
8.510	22.516				
onnet_mou_7		-4.3251	1.817	-2.380	0.017
-7.886	-0.764				
onnet_mou_8		2.3528	1.825	1.289	0.197
-1.225	5.930				
offnet_mou_6		15.0874	3.361	4.489	0.000
8.500	21.674				
offnet_mou_7		-1.7629	1.721	-1.024	0.306
-5.136	1.611				
offnet_mou_8		-0.5496	1.883	-0.292	0.770
-4.240	3.141				
roam_ic_mou_6		0.1622	0.036	4.473	0.000
0.091	0.233				
roam_ic_mou_7		-0.0099	0.052	-0.189	0.850
-0.112	0.093				
roam_ic_mou_8		0.2041	0.044	4.663	0.000
0.118	0.290				
roam_og_mou_6		-5.1505	1.131	-4.554	0.000

-7.367	-2.934				
roam_og_mou_7		0.8855	0.474	1.867	0.062
-0.044	1.815				
roam_og_mou_8		0.0927	0.531	0.175	0.861
-0.948	1.133				
loc_og_t2t_mou_6		-3302.8216	656.377	-5.032	0.000
-4589.297	-2016.346				
loc_og_t2t_mou_7		-1474.6175	679.783	-2.169	0.030
-2806.968	-142.267				
loc_og_t2t_mou_8		5516.1251	628.160	8.781	0.000
4284.953	6747.297				
loc_og_t2m_mou_6		-3342.4429	664.131	-5.033	0.000
-4644.116	-2040.770				
loc_og_t2m_mou_7		-1392.1079	641.104	-2.171	0.030
-2648.649	-135.567				
loc_og_t2m_mou_8		5887.3829	670.271	8.784	0.000
4573.677	7201.089				
loc_og_t2f_mou_6		-285.2245	56.710	-5.030	0.000
-396.373	-174.076				
loc_og_t2f_mou_7		-123.0165	56.677	-2.171	0.030
-234.101	-11.933				
loc_og_t2f_mou_8		487.3991	55.519	8.779	0.000
378.584	596.214				
loc_og_t2c_mou_6		0.0433	0.022	1.982	0.048
0.000	0.086				
loc_og_t2c_mou_7		0.0099	0.021	0.463	0.643
-0.032	0.052				
loc_og_t2c_mou_8		0.0673	0.023	2.988	0.003
0.023	0.111				
loc_og_mou_6		3756.2132	1269.036	2.960	0.003
1268.947	6243.479				
loc_og_mou_7		5686.5575	1330.512	4.274	0.000
3078.802	8294.313				
loc_og_mou_8		-265.7885	1351.069	-0.197	0.844
-2913.835	2382.258				
std_og_t2t_mou_6		-1.309e+04	1867.084	-7.011	0.000
-1.67e+04	-9430.445				
std_og_t2t_mou_7		-9674.1364	1822.022	-5.310	0.000
-1.32e+04	-6103.040				
std_og_t2t_mou_8		5854.8113	1510.088	3.877	0.000
2895.093	8814.530				
std_og_t2m_mou_6		-1.214e+04	1732.071	-7.011	0.000
-1.55e+04	-8749.312				
std_og_t2m_mou_7		-9438.8725	1777.374	-5.311	0.000
-1.29e+04	-5955.284				
std_og_t2m_mou_8		5966.0664	1538.126	3.879	0.000
2951.394	8980.739				
std_og_t2f_mou_6		-255.4281	36.393	-7.019	0.000
-326.757	-184.099				

std_og_t2f_mou_7	-213.5957	40.259	-5.306	0.000
-292.502 -134.689				
std_og_t2f_mou_8	142.4571	36.758	3.876	0.000
70.412 214.502				
std_og_t2c_mou_6	-3.536e-06	0.000	-0.018	0.986
-0.000 0.000				
std_og_t2c_mou_7	-2.353e-06	0.000	-0.014	0.989
-0.000 0.000				
std_og_t2c_mou_8	-2.486e-06	0.000	-0.017	0.986
-0.000 0.000				
std_og_mou_6	1.446e+04	2966.600	4.875	0.000
8646.469 2.03e+04				
std_og_mou_7	2.105e+04	3103.817	6.783	0.000
1.5e+04 2.71e+04				
std_og_mou_8	7815.2524	2767.836	2.824	0.005
2390.393 1.32e+04				
isd_og_mou_6	-51.5636	29.686	-1.737	0.082
-109.747 6.620				
isd_og_mou_7	94.2299	27.788	3.391	0.001
39.767 148.693				
isd_og_mou_8	320.5622	34.133	9.392	0.000
253.663 387.462				
spl_og_mou_6	-83.1552	47.782	-1.740	0.082
-176.805 10.495				
spl_og_mou_7	229.8600	67.719	3.394	0.001
97.134 362.586				
spl_og_mou_8	523.1539	55.587	9.411	0.000
414.206 632.102				
og_others_6	-10.0916	5.818	-1.734	0.083
-21.496 1.312				
og_others_7	15.6443	4.903	3.191	0.001
6.034 25.255				
og_others_8	-5276.7831	3.24e+05	-0.016	0.987
-6.41e+05 6.3e+05				
total_og_mou_6	3406.4402	1971.608	1.728	0.084
-457.840 7270.720				
total_og_mou_7	-7829.5225	2307.277	-3.393	0.001
-1.24e+04 -3307.343				
total_og_mou_8	-1.894e+04	2011.752	-9.413	0.000
-2.29e+04 -1.5e+04				
loc_ic_t2t_mou_6	-471.9694	401.748	-1.175	0.240
-1259.382 315.443				
loc_ic_t2t_mou_7	2043.4379	441.193	4.632	0.000
1178.715 2908.161				
loc_ic_t2t_mou_8	6411.9241	417.700	15.351	0.000
5593.247 7230.601				
loc_ic_t2m_mou_6	-662.3378	563.889	-1.175	0.240
-1767.540 442.865				
loc_ic_t2m_mou_7	2751.5586	593.997	4.632	0.000

1587.346	3915.772				
loc_ic_t2m_mou_8		9239.7305	601.928	15.350	0.000
8059.973	1.04e+04				
loc_ic_t2f_mou_6		-130.8717	111.320	-1.176	0.240
-349.055	87.312				
loc_ic_t2f_mou_7		595.9951	128.711	4.630	0.000
343.725	848.265				
loc_ic_t2f_mou_8		1755.6262	114.390	15.348	0.000
1531.426	1979.826				
loc_ic_mou_6		-1472.7581	1056.667	-1.394	0.163
-3543.788	598.272				
loc_ic_mou_7		-2703.9336	1115.928	-2.423	0.015
-4891.113	-516.754				
loc_ic_mou_8		3460.7823	1136.224	3.046	0.002
1233.824	5687.740				
std_ic_t2t_mou_6		-2047.5989	316.623	-6.467	0.000
-2668.169	-1427.028				
std_ic_t2t_mou_7		-414.5246	317.544	-1.305	0.192
-1036.900	207.851				
std_ic_t2t_mou_8		-551.5337	227.041	-2.429	0.015
-996.526	-106.541				
std_ic_t2m_mou_6		-2117.6774	327.457	-6.467	0.000
-2759.481	-1475.874				
std_ic_t2m_mou_7		-425.3418	325.663	-1.306	0.192
-1063.630	212.946				
std_ic_t2m_mou_8		-844.5218	347.926	-2.427	0.015
-1526.445	-162.599				
std_ic_t2f_mou_6		-364.6880	56.406	-6.465	0.000
-475.242	-254.134				
std_ic_t2f_mou_7		-79.5903	61.102	-1.303	0.193
-199.347	40.167				
std_ic_t2f_mou_8		-138.8260	56.895	-2.440	0.015
-250.338	-27.314				
std_ic_t2o_mou_6		-5.826e-07	2.86e-05	-0.020	0.984
-5.67e-05	5.55e-05				
std_ic_t2o_mou_7		1.092e-06	7.95e-05	0.014	0.989
-0.000	0.000				
std_ic_t2o_mou_8		1.37e-06	8.38e-05	0.016	0.987
-0.000	0.000				
std_ic_mou_6		1980.5760	602.710	3.286	0.001
799.287	3161.865				
std_ic_mou_7		1297.3541	611.724	2.121	0.034
98.398	2496.310				
std_ic_mou_8		8343.2863	569.153	14.659	0.000
7227.768	9458.805				
total_ic_mou_6		2863.2928	942.847	3.037	0.002
1015.348	4711.238				
total_ic_mou_7		-1538.9468	1008.240	-1.526	0.127
-3515.061	437.167				

total_ic_mou_8	-1.982e+04	1035.030	-19.153	0.000
-2.19e+04	-1.78e+04			
spl_ic_mou_6	-1.5369	0.562	-2.733	0.006
-2.639	-0.435			
spl_ic_mou_7	0.5833	0.518	1.127	0.260
-0.431	1.598			
spl_ic_mou_8	5.2078	0.295	17.671	0.000
4.630	5.785			
isd_ic_mou_6	-483.9815	159.328	-3.038	0.002
-796.259	-171.704			
isd_ic_mou_7	274.3329	179.671	1.527	0.127
-77.816	626.482			
isd_ic_mou_8	3507.8222	183.152	19.153	0.000
3148.851	3866.794			
ic_others_6	-81.0624	26.650	-3.042	0.002
-133.295	-28.829			
ic_others_7	42.4590	27.835	1.525	0.127
-12.096	97.014			
ic_others_8	550.1144	28.772	19.120	0.000
493.723	606.506			
total_rech_num_6	0.0224	0.035	0.638	0.523
-0.046	0.091			
total_rech_num_7	0.0726	0.040	1.804	0.071
-0.006	0.151			
total_rech_num_8	-0.6403	0.041	-15.682	0.000
-0.720	-0.560			
total_rech_amt_6	0.6131	0.082	7.465	0.000
0.452	0.774			
total_rech_amt_7	-0.2171	0.080	-2.701	0.007
-0.375	-0.060			
total_rech_amt_8	0.2182	0.114	1.913	0.056
-0.005	0.442			
max_rech_amt_6	-0.2237	0.037	-6.053	0.000
-0.296	-0.151			
max_rech_amt_7	-0.0587	0.036	-1.645	0.100
-0.129	0.011			
max_rech_amt_8	0.1475	0.043	3.398	0.001
0.062	0.233			
last_day_rch_amt_6	-0.1756	0.029	-6.043	0.000
-0.233	-0.119			
last_day_rch_amt_7	0.0028	0.029	0.098	0.922
-0.054	0.059			
last_day_rch_amt_8	-0.5102	0.033	-15.449	0.000
-0.575	-0.445			
vol_2g_mb_6	0.1398	0.030	4.724	0.000
0.082	0.198			
vol_2g_mb_7	0.0299	0.032	0.927	0.354
-0.033	0.093			
vol_2g_mb_8	0.0882	0.034	2.580	0.010

0.021	0.155				
vol_3g_mb_6		0.3625	0.044	8.158	0.000
0.275	0.450				
vol_3g_mb_7		0.4089	0.056	7.289	0.000
0.299	0.519				
vol_3g_mb_8		-0.1838	0.068	-2.700	0.007
-0.317	-0.050				
monthly_2g_6		-0.6068	0.045	-13.514	0.000
-0.695	-0.519				
monthly_2g_7		-0.4095	0.042	-9.834	0.000
-0.491	-0.328				
monthly_2g_8		-0.6419	0.059	-10.961	0.000
-0.757	-0.527				
sachet_2g_6		-0.0239	0.031	-0.773	0.439
-0.085	0.037				
sachet_2g_7		-0.2143	0.033	-6.464	0.000
-0.279	-0.149				
sachet_2g_8		-0.2391	0.032	-7.513	0.000
-0.301	-0.177				
monthly_3g_6		-0.3220	0.046	-6.989	0.000
-0.412	-0.232				
monthly_3g_7		-0.5808	0.052	-11.100	0.000
-0.683	-0.478				
monthly_3g_8		-0.8649	0.078	-11.083	0.000
-1.018	-0.712				
sachet_3g_6		-0.0281	0.032	-0.871	0.384
-0.091	0.035				
sachet_3g_7		-0.0829	0.042	-1.964	0.050
-0.166	-0.000				
sachet_3g_8		-0.1553	0.048	-3.222	0.001
-0.250	-0.061				
aon		-0.1564	0.022	-7.269	0.000
-0.199	-0.114				
aug_vbc_3g		-0.1965	0.057	-3.441	0.001
-0.308	-0.085				
jul_vbc_3g		-0.0522	0.047	-1.118	0.264
-0.144	0.039				
jun_vbc_3g		0.2364	0.047	5.007	0.000
0.144	0.329				
decrease_mou_action		-0.4989	0.053	-9.461	0.000
-0.602	-0.396				
decrease_rech_num_action		-1.0229	0.048	-21.429	0.000
-1.116	-0.929				
decrease_rech_amt_action		-0.3065	0.065	-4.720	0.000
-0.434	-0.179				
decrease_arpu_action		-0.1797	0.067	-2.701	0.007
-0.310	-0.049				
decrease_vbc_action		-1.7537	0.130	-13.538	0.000
-2.008	-1.500				


```
=====
=====
" " "
```

Model analysis

1. We can see that there are few features have positive coefficients and few have negative.
2. Many features have higher p-values and hence became insignificant in the model.

Coarse tuning (Auto+Manual)

We'll first eliminate a few features using Recursive Feature Elimination (RFE), and once we have reached a small set of variables to work with, we can then use manual feature elimination (i.e. manually eliminating features based on observing the p-values and VIFs).

Feature Selection Using RFE

```
# Importing logistic regression from sklearn
from sklearn.linear_model import LogisticRegression
# Intantiate the logistic regression
logreg = LogisticRegression()
```

RFE with 15 columns

```
# Importing RFE
from sklearn.feature_selection import RFE

# Intantiate RFE with 15 columns
rfe = RFE(logreg, 15)

# Fit the rfe model with train set
rfe = rfe.fit(X_train, y_train)

# RFE selected columns
rfe_cols = X_train.columns[rfe.support_]
print(rfe_cols)

Index(['offnet_mou_7', 'offnet_mou_8', 'roam_og_mou_8',
      'std_og_t2m_mou_8',
      'isd_og_mou_8', 'og_others_7', 'og_others_8',
      'loc_ic_t2f_mou_8',
      'loc_ic_mou_8', 'std_ic_t2f_mou_8', 'ic_others_8',
      'total_rech_num_8',
      'monthly_2g_8', 'monthly_3g_8', 'decrease_vbc_action'],
      dtype='object')
```

Model-1 with RFE selected columns

```
# Adding constant to X_train
X_train_sm_1 = sm.add_constant(X_train[rfe_cols])
```

```
#Instantiate the model
```

```
log_no_pca_1 = sm.GLM(y_train, X_train_sm_1,  
family=sm.families.Binomial())
```

```
# Fit the model
```

```
log_no_pca_1 = log_no_pca_1.fit()
```

```
log_no_pca_1.summary()
```

```
<class 'statsmodels.iolib.summary.Summary'>
```

```
"""
```

Generalized Linear Model Regression Results

```
=====
```

```
Dep. Variable:                churn    No. Observations:  
42850
```

```
Model:                        GLM      Df Residuals:  
42834
```

```
Model Family:                Binomial  Df Model:  
15
```

```
Link Function:                logit     Scale:  
1.0000
```

```
Method:                       IRLS     Log-Likelihood:  
nan
```

```
Date:                         Sat, 16 May 2020  Deviance:  
nan
```

```
Time:                         18:04:18    Pearson chi2:  
4.49e+06
```

```
No. Iterations:                100
```

```
Covariance Type:              nonrobust
```

```
=====
```

```
=====
```

```
[0.025    0.975]
```

```
-----
```

```
-----
```

```
const                -58.6610    4419.624    -0.013    0.989    -
```

```
8720.965    8603.643
```

```
offnet_mou_7         0.6096     0.026    23.449    0.000
```

```
0.559    0.661
```

```
offnet_mou_8        -3.2532     0.106   -30.548    0.000    -
```

```
3.462    -3.045
```

```
roam_og_mou_8        1.2482     0.032    39.496    0.000
```

```
1.186    1.310
```

```
std_og_t2m_mou_8     2.4408     0.094    26.101    0.000
```

```
2.258    2.624
```

```
isd_og_mou_8        -1.0212     0.194    -5.271    0.000    -
```

```

1.401      -0.641
og_others_7      -1.1915      0.862      -1.382      0.167      -
2.881      0.498
og_others_8      -4191.9652      3.22e+05      -0.013      0.990      -
6.35e+05      6.27e+05
loc_ic_t2f_mou_8      -0.7547      0.072      -10.487      0.000      -
0.896      -0.614
loc_ic_mou_8      -1.9744      0.066      -30.078      0.000      -
2.103      -1.846
std_ic_t2f_mou_8      -0.7922      0.075      -10.607      0.000      -
0.939      -0.646
ic_others_8      -1.4913      0.132      -11.305      0.000      -
1.750      -1.233
total_rech_num_8      -0.4840      0.018      -26.977      0.000      -
0.519      -0.449
monthly_2g_8      -0.9031      0.043      -20.851      0.000      -
0.988      -0.818
monthly_3g_8      -0.9871      0.043      -22.711      0.000      -
1.072      -0.902
decrease_vbc_action      -1.3078      0.073      -17.956      0.000      -
1.451      -1.165
=====
=====
"""

```

Checking VIFs

```

# Check for the VIF values of the feature variables.
from statsmodels.stats.outliers_influence import
variance_inflation_factor

# Create a dataframe that will contain the names of all the feature
variables and their respective VIFs
vif = pd.DataFrame()
vif['Features'] = X_train[rfe_cols].columns
vif['VIF'] = [variance_inflation_factor(X_train[rfe_cols].values, i)
for i in range(X_train[rfe_cols].shape[1])]
vif['VIF'] = round(vif['VIF'], 2)
vif = vif.sort_values(by = "VIF", ascending = False)
vif

```

	Features	VIF
1	offnet_mou_8	7.45
3	std_og_t2m_mou_8	6.27
0	offnet_mou_7	1.92
8	loc_ic_mou_8	1.68
7	loc_ic_t2f_mou_8	1.21
11	total_rech_num_8	1.19
2	roam_og_mou_8	1.16
14	decrease_vbc_action	1.08

13	monthly_3g_8	1.06
6	og_others_8	1.05
12	monthly_2g_8	1.05
5	og_others_7	1.04
9	std_ic_t2f_mou_8	1.02
10	ic_others_8	1.02
4	isd_og_mou_8	1.01

Removing column og_others_8, which is insignificant as it has the highest p-value 0.99

```
# Removing og_others_8 column
log_cols = rfe_cols.to_list()
log_cols.remove('og_others_8')
print(log_cols)

['offnet_mou_7', 'offnet_mou_8', 'roam_og_mou_8', 'std_og_t2m_mou_8',
'isd_og_mou_8', 'og_others_7', 'loc_ic_t2f_mou_8', 'loc_ic_mou_8',
'std_ic_t2f_mou_8', 'ic_others_8', 'total_rech_num_8', 'monthly_2g_8',
'monthly_3g_8', 'decrease_vbc_action']
```

Model-2

Building the model after removing og_others_8 variable.

```
# Adding constant to X_train
X_train_sm_2 = sm.add_constant(X_train[log_cols])

#Instantiate the model
log_no_pca_2 = sm.GLM(y_train, X_train_sm_2,
family=sm.families.Binomial())

# Fit the model
log_no_pca_2 = log_no_pca_2.fit()

log_no_pca_2.summary()

<class 'statsmodels.iolib.summary.Summary'>
"""
                    Generalized Linear Model Regression Results

=====
=====
Dep. Variable:                churn    No. Observations:
42850
Model:                        GLM      Df Residuals:
42835
Model Family:                Binomial  Df Model:
14
Link Function:                logit     Scale:
1.0000
```

Method: IRLS Log-Likelihood:
-15034.
Date: Sat, 16 May 2020 Deviance:
30068.
Time: 18:06:36 Pearson chi2:
4.51e+06
No. Iterations: 11
Covariance Type: nonrobust

```
=====
=====
              coef      std err          z      P>|z|
[0.025      0.975]
-----
const          -1.1052      0.031    -35.342      0.000      -
1.167      -1.044
offnet_mou_7      0.6081      0.026     23.427      0.000
0.557      0.659
offnet_mou_8     -3.2557      0.106    -30.603      0.000      -
3.464     -3.047
roam_og_mou_8      1.2491      0.031     39.747      0.000
1.188      1.311
std_og_t2m_mou_8      2.4428      0.093     26.146      0.000
2.260      2.626
isd_og_mou_8     -1.0982      0.196     -5.590      0.000      -
1.483     -0.713
og_others_7     -1.8793      0.818     -2.299      0.022      -
3.482     -0.277
loc_ic_t2f_mou_8     -0.7548      0.072    -10.491      0.000      -
0.896     -0.614
loc_ic_mou_8     -1.9714      0.066    -30.058      0.000      -
2.100     -1.843
std_ic_t2f_mou_8     -0.8020      0.075    -10.727      0.000      -
0.949     -0.655
ic_others_8     -1.4871      0.132    -11.278      0.000      -
1.746     -1.229
total_rech_num_8     -0.4864      0.018    -27.146      0.000      -
0.522     -0.451
monthly_2g_8     -0.9066      0.043    -20.866      0.000      -
0.992     -0.821
monthly_3g_8     -0.9862      0.043    -22.700      0.000      -
1.071     -0.901
decrease_vbc_action -1.3097      0.073    -17.994      0.000      -
1.452     -1.167
=====
=====
"" ""
```

Checking VIF for Model-2

```
# Create a dataframe that will contain the names of all the feature
variables and their respective VIFs
vif = pd.DataFrame()
vif['Features'] = X_train[log_cols].columns
vif['VIF'] = [variance_inflation_factor(X_train[log_cols].values, i)
for i in range(X_train[log_cols].shape[1])]
vif['VIF'] = round(vif['VIF'], 2)
vif = vif.sort_values(by = "VIF", ascending = False)
vif
```

	Features	VIF
1	offnet_mou_8	7.45
3	std_og_t2m_mou_8	6.27
0	offnet_mou_7	1.92
7	loc_ic_mou_8	1.68
6	loc_ic_t2f_mou_8	1.21
10	total_rech_num_8	1.19
2	roam_og_mou_8	1.16
13	decrease_vbc_action	1.08
12	monthly_3g_8	1.06
11	monthly_2g_8	1.05
8	std_ic_t2f_mou_8	1.02
4	isd_og_mou_8	1.01
9	ic_others_8	1.01
5	og_others_7	1.00

As we can see from the model summary that all the variables p-values are significant and offnet_mou_8 column has the highest VIF 7.45. Hence, deleting offnet_mou_8 column.

```
# Removing offnet_mou_8 column
log_cols.remove('offnet_mou_8')
```

Model-3

Model after removing offnet_mou_8 column.

```
# Adding constant to X_train
X_train_sm_3 = sm.add_constant(X_train[log_cols])

#Instantiate the model
log_no_pca_3 = sm.GLM(y_train, X_train_sm_3,
family=sm.families.Binomial())

# Fit the model
log_no_pca_3 = log_no_pca_3.fit()

log_no_pca_3.summary()
```

```
<class 'statsmodels.iolib.summary.Summary'>
```

```
"""
```

Generalized Linear Model Regression Results

```
=====
Dep. Variable:          churn    No. Observations:
42850
Model:                  GLM      Df Residuals:
42836
Model Family:          Binomial  Df Model:
13
Link Function:          logit    Scale:
1.0000
Method:                 IRLS     Log-Likelihood:
-15720.
Date:                   Sat, 16 May 2020    Deviance:
31440.
Time:                   18:07:30    Pearson chi2:
3.92e+06
No. Iterations:         11

Covariance Type:        nonrobust
=====
```

```
=====
[0.025    0.975]
-----
coef      std err      z      P>|z|
-----
const      -1.2058      0.032    -37.536      0.000      -
1.269      -1.143
offnet_mou_7      0.3665      0.022     16.456      0.000
0.323       0.410
roam_og_mou_8      0.7135      0.024     29.260      0.000
0.666       0.761
std_og_t2m_mou_8     -0.2474      0.022    -11.238      0.000      -
0.291      -0.204
isd_og_mou_8     -1.3811      0.212     -6.511      0.000      -
1.797      -0.965
og_others_7     -2.4711      0.872     -2.834      0.005      -
4.180      -0.762
loc_ic_t2f_mou_8     -0.7102      0.075     -9.532      0.000      -
0.856      -0.564
loc_ic_mou_8     -3.3287      0.057    -58.130      0.000      -
3.441     -3.216
std_ic_t2f_mou_8     -0.9503      0.078    -12.181      0.000      -
1.103      -0.797
ic_others_8     -1.5131      0.129    -11.771      0.000      -
1.765     -1.261
=====
```

```

total_rech_num_8      -0.5060      0.018      -28.808      0.000      -
0.540      -0.472
monthly_2g_8          -0.9279      0.044      -21.027      0.000      -
1.014      -0.841
monthly_3g_8          -1.0943      0.046      -23.615      0.000      -
1.185      -1.004
decrease_vbc_action    -1.3293      0.072      -18.478      0.000      -
1.470      -1.188
=====
=====
"""

```

VIF Model-3

```

vif = pd.DataFrame()
vif['Features'] = X_train[log_cols].columns
vif['VIF'] = [variance_inflation_factor(X_train[log_cols].values, i)
for i in range(X_train[log_cols].shape[1])]
vif['VIF'] = round(vif['VIF'], 2)
vif = vif.sort_values(by = "VIF", ascending = False)
vif

```

	Features	VIF
2	std_og_t2m_mou_8	1.87
0	offnet_mou_7	1.72
6	loc_ic_mou_8	1.33
5	loc_ic_t2f_mou_8	1.21
9	total_rech_num_8	1.17
12	decrease_vbc_action	1.07
1	roam_og_mou_8	1.06
11	monthly_3g_8	1.06
10	monthly_2g_8	1.05
7	std_ic_t2f_mou_8	1.02
3	isd_og_mou_8	1.01
8	ic_others_8	1.01
4	og_others_7	1.00

Now from the model summary and the VIF list we can see that all the variables are significant and there is no multicollinearity among the variables.

Hence, we can concluded that ***Model-3 log_no_pca_3 will be the final model.***

Model performance on the train set

```

# Getting the predicted value on the train set
y_train_pred_no_pca = log_no_pca_3.predict(X_train_sm_3)
y_train_pred_no_pca.head()

0      2.687411e-01
1      7.047483e-02

```



```
3    0
4    0
```

Now let's calculate the accuracy sensitivity and specificity for various probability cutoffs.

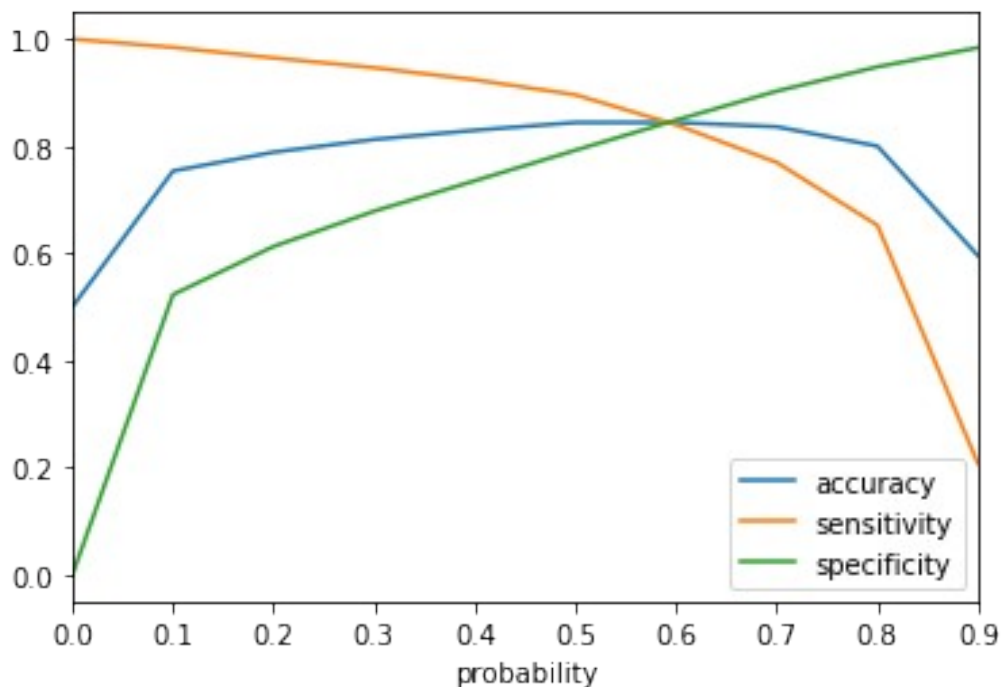
```
# Creating a dataframe
cutoff_df = pd.DataFrame(columns=['probability', 'accuracy',
'sensitivity', 'specificity'])

for i in prob_cutoff:
    cm1 = metrics.confusion_matrix(y_train_pred_final['churn'],
y_train_pred_final[i] )
    total1=sum(sum(cm1))
    accuracy = (cm1[0,0]+cm1[1,1])/total1

    speci = cm1[0,0]/(cm1[0,0]+cm1[0,1])
    sensi = cm1[1,1]/(cm1[1,0]+cm1[1,1])
    cutoff_df.loc[i] =[ i ,accuracy,sensi,speci]
print(cutoff_df)
```

	probability	accuracy	sensitivity	specificity
0.0	0.0	0.500000	1.000000	0.000000
0.1	0.1	0.753629	0.984411	0.522847
0.2	0.2	0.788751	0.964714	0.612789
0.3	0.3	0.812509	0.946371	0.678646
0.4	0.4	0.829638	0.923874	0.735403
0.5	0.5	0.844131	0.895823	0.792439
0.6	0.6	0.844271	0.839860	0.848681
0.7	0.7	0.836173	0.769522	0.902824
0.8	0.8	0.800163	0.652275	0.948051
0.9	0.9	0.595426	0.207001	0.983851

```
# Plotting accuracy, sensitivity and specificity for different
probabilities.
cutoff_df.plot('probability',
['accuracy','sensitivity','specificity'])
plt.show()
```



Analysis of the above curve

Accuracy - Becomes stable around 0.6

Sensitivity - Decreases with the increased probability.

Specificity - Increases with the increasing probability.

At point 0.6 where the three parameters cut each other, we can see that there is a balance between sensitivity and specificity with a good accuracy.

Here we are intended to achieve better sensitivity than accuracy and specificity. Though as per the above curve, we should take 0.6 as the optimum probability cutoff, we are taking **0.5** for achieving higher sensitivity, which is our main goal.

```
# Creating a column with name "predicted", which is the predicted
value for 0.5 cutoff
y_train_pred_final['predicted'] =
y_train_pred_final['churn_prob'].map(lambda x: 1 if x > 0.5 else 0)
y_train_pred_final.head()
```

	churn	churn_prob	CustID	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7
0.8 \											
0	0	2.687411e-01	0	1	1	1	0	0	0	0	0
0											
1	0	7.047483e-02	1	1	0	0	0	0	0	0	0
0											
2	0	8.024370e-02	2	1	0	0	0	0	0	0	0
0											
3	0	3.439222e-03	3	1	0	0	0	0	0	0	0

0											
4	0	5.253815e-19		4	1	0	0	0	0	0	0
0											

	0.9	predicted
0	0	0
1	0	0
2	0	0
3	0	0
4	0	0

Metrics

```
# Confusion metrics
confusion = metrics.confusion_matrix(y_train_pred_final['churn'],
y_train_pred_final['predicted'])
print(confusion)

[[16978  4447]
 [ 2232 19193]]

TP = confusion[1,1] # true positive
TN = confusion[0,0] # true negatives
FP = confusion[0,1] # false positives
FN = confusion[1,0] # false negatives

# Accuracy
print("Accuracy:-",metrics.accuracy_score(y_train_pred_final['churn'],
y_train_pred_final['predicted']))

# Sensitivity
print("Sensitivity:-",TP / float(TP+FN))

# Specificity
print("Specificity:-", TN / float(TN+FP))

Accuracy:- 0.8441306884480747
Sensitivity:- 0.8958226371061844
Specificity:- 0.792438739789965
```

We have got good accuracy, sensitivity and specificity on the train set prediction.

Plotting the ROC Curve (Trade off between sensitivity & specificity)

```
# ROC Curve function

def draw_roc( actual, probs ):
    fpr, tpr, thresholds = metrics.roc_curve( actual, probs,
                                              drop_intermediate =
False )
    auc_score = metrics.roc_auc_score( actual, probs )
    plt.figure(figsize=(5, 5))
```

```

plt.plot( fpr, tpr, label='ROC curve (area = %0.2f)' % auc_score )
plt.plot([0, 1], [0, 1], 'k--')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate or [1 - True Negative Rate]')
plt.ylabel('True Positive Rate')
plt.title('Receiver operating characteristic example')
plt.legend(loc="lower right")
plt.show()

```

```

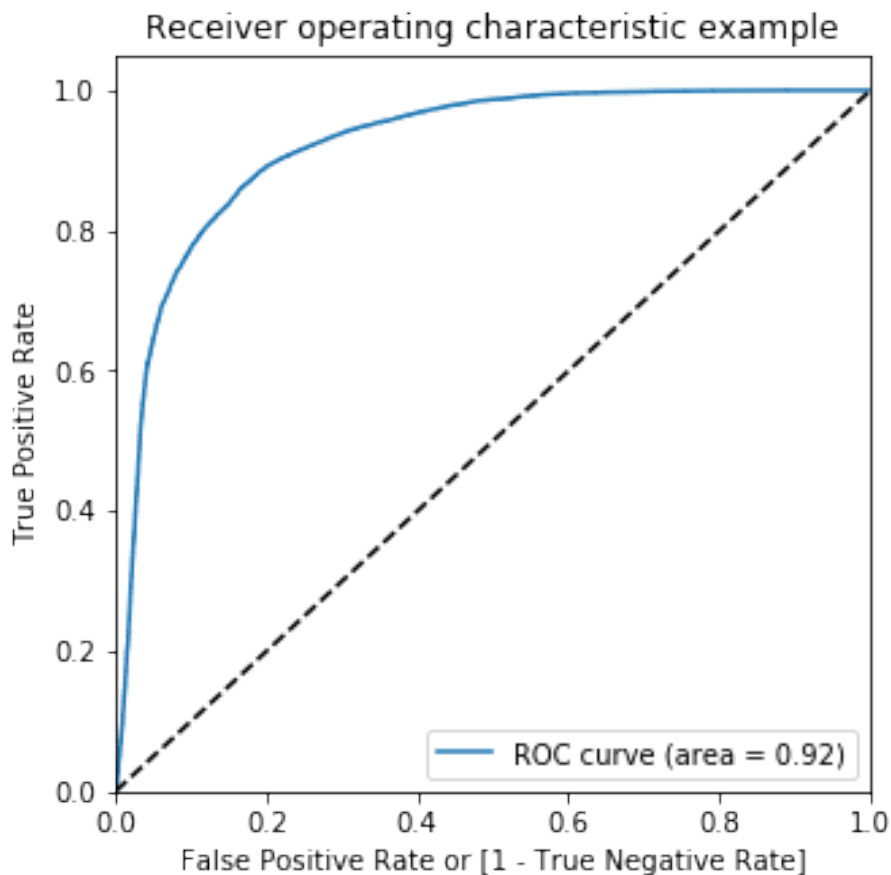
return None

```

```

draw_roc(y_train_pred_final['churn'],
y_train_pred_final['churn_prob'])

```



We can see the area of the ROC curve is closer to 1, which is the Gini of the model.

Testing the model on the test set

```

# Taking a copy of the test set
X_test_log = X_test.copy()

```

```

# Taking only the columns, which are selected in the train set after
removing insignificant and multicollinear variables
X_test_log = X_test_log[log_cols]

# Adding constant on the test set
X_test_sm = sm.add_constant(X_test_log)

```

Predictions on the test set with final model

```

# Predict on the test set
y_test_pred = log_no_pca_3.predict(X_test_sm)

y_test_pred.head()

5704      0.034015
64892     0.000578
39613     0.513564
93118     0.020480
81235     0.034115
dtype: float64

# Converting y_test_pred to a dataframe because y_test_pred is an
array
y_pred_1 = pd.DataFrame(y_test_pred)
y_pred_1.head()

              0
5704      0.034015
64892     0.000578
39613     0.513564
93118     0.020480
81235     0.034115

# Convetting y_test to a dataframe
y_test_df = pd.DataFrame(y_test)
y_test_df.head()

      churn
5704      0
64892      0
39613      0
93118      0
81235      0

# Putting index to Customer ID
y_test_df['CustID'] = y_test_df.index

# Removing index form the both dataframes for merging them side by
side
y_pred_1.reset_index(drop=True, inplace=True)
y_test_df.reset_index(drop=True, inplace=True)

```

```
# Appending y_pred_1 and y_test_df
y_test_pred_final = pd.concat([y_test_df, y_pred_1], axis=1)

y_test_pred_final.head()
```

	churn	CustID	0
0	0	5704	0.034015
1	0	64892	0.000578
2	0	39613	0.513564
3	0	93118	0.020480
4	0	81235	0.034115

```
# Renaming the '0' column as churn probablity
y_test_pred_final = y_test_pred_final.rename(columns={0: 'churn_prob'})

# Rearranging the columns
y_test_pred_final =
y_test_pred_final.reindex_axis(['CustID', 'churn', 'churn_prob'],
axis=1)

y_test_pred_final.head()
```

	CustID	churn	churn_prob
0	5704	0	0.034015
1	64892	0	0.000578
2	39613	0	0.513564
3	93118	0	0.020480
4	81235	0	0.034115

```
# In the test set using probablity cutoff 0.5, what we got in the
train set
y_test_pred_final['test_predicted'] =
y_test_pred_final['churn_prob'].map(lambda x: 1 if x > 0.5 else 0)

y_test_pred_final.head()
```

	CustID	churn	churn_prob	test_predicted
0	5704	0	0.034015	0
1	64892	0	0.000578	0
2	39613	0	0.513564	1
3	93118	0	0.020480	0
4	81235	0	0.034115	0

Metrics

```
# Confusion matrix
confusion = metrics.confusion_matrix(y_test_pred_final['churn'],
y_test_pred_final['test_predicted'])
print(confusion)
```

```
[[4190 1158]
 [ 34 159]]
```

```

TP = confusion[1,1] # true positive
TN = confusion[0,0] # true negatives
FP = confusion[0,1] # false positives
FN = confusion[1,0] # false negatives

# Accuracy
print("Accuracy:-",metrics.accuracy_score(y_test_pred_final['churn'],
y_test_pred_final['test_predicted']))

# Sensitivity
print("Sensitivity:-",TP / float(TP+FN))

# Specificity
print("Specificity:-", TN / float(TN+FP))

Accuracy:- 0.7848763761053962
Sensitivity:- 0.8238341968911918
Specificity:- 0.7834704562453254

```

Model summary

- Train set
 - Accuracy = 0.84
 - Sensitivity = 0.81
 - Specificity = 0.83
- Test set
 - Accuracy = 0.78
 - Sensitivity = 0.82
 - Specificity = 0.78

Overall, the model is performing well in the test set, what it had learnt from the train set.

Final conclusion with no PCA

We can see that the logistic model with no PCA has good sensitivity and accuracy, which are comparable to the models with PCA. So, we can go for the more simplistic model such as logistic regression with PCA as it explains the important predictor variables as well as the significance of each variable. The model also helps us to identify the variables which should be acted upon for making the decision of the to be churned customers. Hence, the model is more relevant in terms of explaining to the business.

Business recommendation

Top predictors

Below are few top variables selected in the logistic regression model.

Variables	Coefficients
loc_ic_mou_8	-3.3287

Variables	Coefficients
og_others_7	-2.4711
ic_others_8	-1.5131
isd_og_mou_8	-1.3811
decrease_vbc_action	-1.3293
monthly_3g_8	-1.0943
std_ic_t2f_mou_8	-0.9503
monthly_2g_8	-0.9279
loc_ic_t2f_mou_8	-0.7102
roam_og_mou_8	0.7135

We can see most of the top variables have negative coefficients. That means, the variables are inversely correlated with the churn probability.

E.g.:-

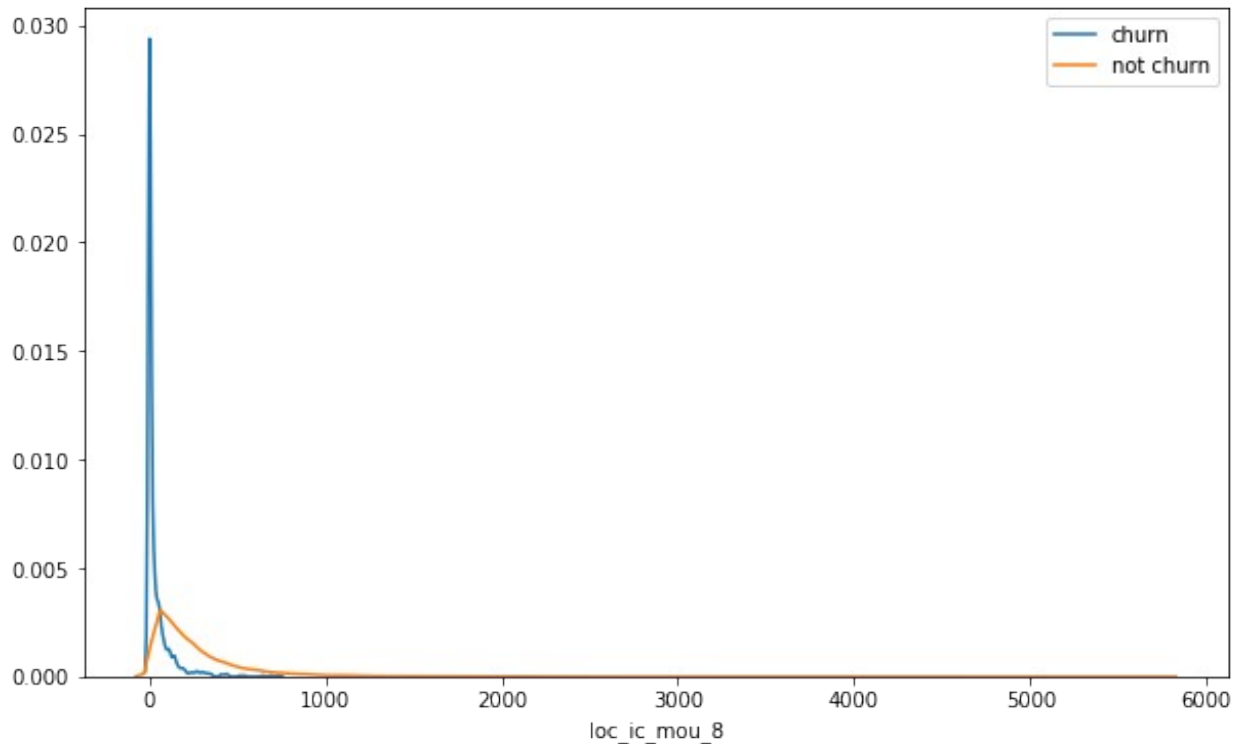
If the local incoming minutes of usage (loc_ic_mou_8) is lesser in the month of August than any other month, then there is a higher chance that the customer is likely to churn.

Recommendations

1. Target the customers, whose minutes of usage of the incoming local calls and outgoing ISD calls are less in the action phase (mostly in the month of August).
2. Target the customers, whose outgoing others charge in July and incoming others on August are less.
3. Also, the customers having value based cost in the action phase increased are more likely to churn than the other customers. Hence, these customers may be a good target to provide offer.
4. Cutomers, whose monthly 3G recharge in August is more, are likely to be churned.
5. Customers having decreasing STD incoming minutes of usage for operators T to fixed lines of T for the month of August are more likely to churn.
6. Cutomers decreasing monthly 2g usage for August are most probable to churn.
7. Customers having decreasing incoming minutes of usage for operators T to fixed lines of T for August are more likely to churn.
8. roam_og_mou_8 variables have positive coefficients (0.7135). That means for the customers, whose roaming outgoing minutes of usage is increasing are more likely to churn.

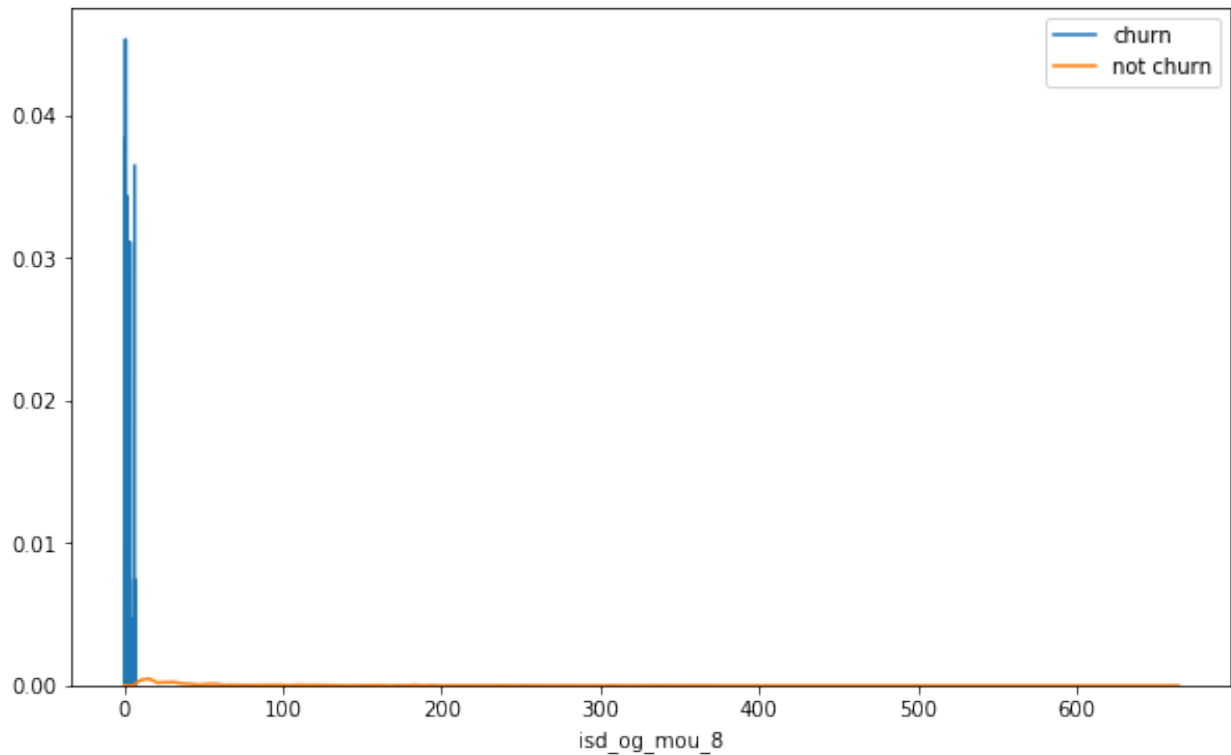
Plots of important predictors for churn and non churn customers

```
# Plotting loc_ic_mou_8 predictor for churn and not churn customers
fig = plt.figure(figsize=(10,6))
sns.distplot(data_churn['loc_ic_mou_8'],label='churn',hist=False)
sns.distplot(data_non_churn['loc_ic_mou_8'],label='not
churn',hist=False)
plt.show()
```



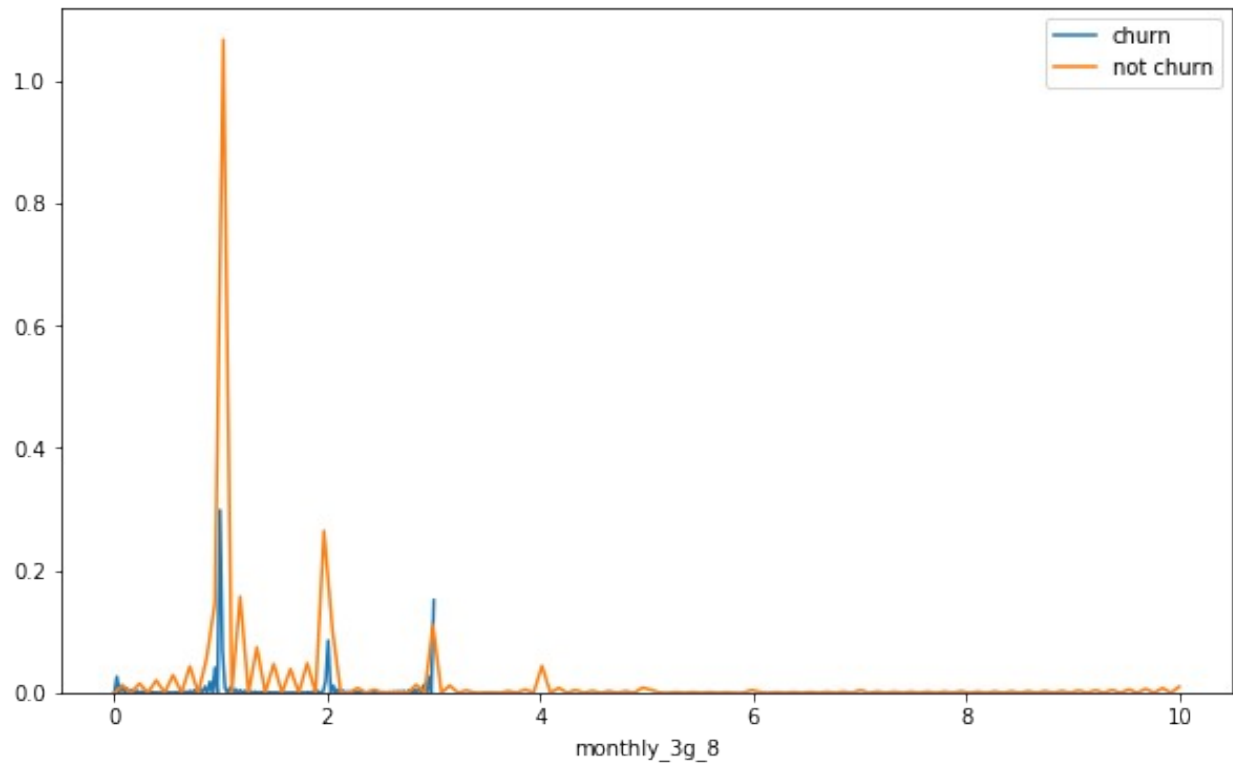
We can see that for the churn customers the minutes of usage for the month of August is mostly populated on the lower side than the non churn customers.

```
# Plotting isd_og_mou_8 predictor for churn and not churn customers
fig = plt.figure(figsize=(10,6))
sns.distplot(data_churn['isd_og_mou_8'],label='churn',hist=False)
sns.distplot(data_non_churn['isd_og_mou_8'],label='not
churn',hist=False)
plt.show()
```



We can see that the ISD outgoing minutes of usage for the month of August for churn customers is densed approximately to zero. On the onther hand for the non churn customers it is little more than the churn customers.

```
# Plotting monthly_3g_8 predictor for churn and not churn customers
fig = plt.figure(figsize=(10,6))
sns.distplot(data_churn['monthly_3g_8'],label='churn',hist=False)
sns.distplot(data_non_churn['monthly_3g_8'],label='not
churn',hist=False)
plt.show()
```



The number of monthly 3g data for August for the churn customers are very much populated around 1, whereas for non churn customers it is spread across various numbers.

Similarly, we can plot each variable, which has higher coefficients, to see the churn distribution.