Case Study
Telecom Churn

Case study telecom churn-Subash, Saloni, Sanjeev Kumar

Data Understanding	Data understanding involves accessing the data and exploring it using tables and graphics that can be organized in IBM® SPSS® Modeler using the CRISP-DM project tool
Dealing with Missing value	One way of handling missing values is the deletion of the rows or columns having null values.
EDA Univariate analysis	The objective of univariate analysis is to derive the data, define and summarize it, and analyze the pattern present in it
Analysis of the action phase's average revenue per customer (churn and not churn)	customer churn refers to the percentage of customers you lose over a given period, revenue churn refers to the percentage of revenue you lose because of lost customers over a given period.
#Analysis of churn rate by the decreasing recharge amount and volume based cost in the action phase	Companies with low churn rates are managing to retain customers.
removing a few derived columns that won't be used in the subsequent analysis	measurement errors, data entry or processing errors, or poor sampling.

Train-Test Split	when you split your data into a training set and a testing set
Dealing with data imbalance	down sample and up weight the majority class.
Feature Scaling	a method used to normalize the range of independent variables or features of data
Logistic regression	Logistic regression is used to obtain odds ratio in the presence of more than one explanatory variable.
Logistic regression with optimal C	C: float, default=1.0 Inverse of regularization strength; must be a positive float
Model summary **Random forest**	The random forest is a classification algorithm consisting of many decisions trees

Table of contents

Business Problem

Customers in the telecom sector have access to a variety of service providers and can actively switch from one operator to another. The telecoms business has an average annual churn rate of 15 to 25 percent in this fiercely competitive market. Customer retention has now surpassed customer acquisition in importance due to the fact that it is 5–10 times more expensive to gain new customers than to keep existing ones.

Data Prep

The following data preparation steps are important for this problem:

1. total_ic_mou_9Create new features The ability to distinguish between excellent and bad models using good features makes this one of the most crucial steps in the data preparation process. We will leverage our knowledge of business to derive characteristics that could serve as key churn indicators.
2. Eliminate high-value clients We simply need to forecast turnover for high-value customers, as was already mentioned. As an example of a high-value client, consider: Those who have recharged with a sum more than or equal to X, where X represents the 70th percentile of the typical recharge sum over the first two months (the favourable phase).
3. Tag churners and take away churn phase traits. Now, depending on the fourth month, tag the customers who churned (churn=1, otherwise 0) as follows: In the churn phase, those who have not placed any calls (incoming or outgoing) OR accessed mobile internet even once. We need to identify churners using the following attributes:

- total_og_mou_9
- vol_2g_mb_9
- vol_3g_mb_9

All the attributes pertaining to the churn phase (all attributes with "_9," "_10," etc.) must be removed after labelling churners.

Modelling

Create models to foresee churn. The predictive model we'll create will have two functions:

It will be used to forecast the churn phase, or whether a high-value customer would leave in the near future. Knowing this allows the business to take appropriate action, such as offering customised programmes or discounts on recharges.

It will be utilised to pinpoint crucial elements that are reliable churn predictors. These factors might also reveal the factors influencing customers' decisions to transfer networks.

In some circumstances, a single machine learning model can accomplish both of the a forementioned objectives.

But because there are so many features in this case, we should try a dimensionali ty reduction method like PCA before creating a predictive model.

We can apply any classification model after PCA.

We will try to use ways to tackle class imbalance because the rate of churn is nor mally modest (between 5 and 10%; this is known as class-imbalance).

To create the model, we might consider the following steps:

1. Perform data preprocessing (convert columns to the proper formats, deal with missing values, etc.).
2. Perform appropriate exploratory analysis to glean valuable insights (whether immediately applicable to business or ultimately applicable for modeling/feature engineering).
3. Derive new features.
4. Use PCA to lessen the number of variables.
5. Train several models, fine-tune model hyperparameters, etc. (adequate procedures should be used to handle class imbalance).
6. Use the right evaluation measures to assess the models. Remember that effectively identifying churners is more crucial than precisely identifying non-churners; use an evaluation metric that matches this business objective.
7. Finally, pick a model based on an assessment metric.

- Only one of the two objectives will be accomplished by the aforementioned model—predicting clients who will leave. The aforementioned methodology cannot be used to pinpoint the critical churn factors. Because PCA typically produces components that are difficult to read, this is the case.
- As a result, we will create a new model with the primary goal of identifying key predictor features that aid in the understanding of churn indicators by the business. A model from the tree family or one based on logistic regression is a solid option for determining important variables. When using logistic regression, we shall take multi-collinearity into account.
- Use plots, summary tables, or other visual representations to visually represent the most significant predictors after finding them.
- Finally, based on our observations, make suggestions for measures to control customer attrition.

01

Data understanding

Understanding Data

df = pd.read_csv('telecom_churn_data.csv')
df.head()

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3 (ipykernel)

Run Markdown

Data Understanding

In [3]:
```
## Understanding Data
df = pd.read_csv('telecom_churn_data.csv')
df.head()
```

Out[3]:

	mobile_number	circle_id	loc_og_t2o_mou	std_og_t2o_mou	loc_ic_t2o_mou	last_date_of_month_6	last_date_of_month_7	last_date_of_month_8	last_date_of
0	7000842753	109	0.0	0.0	0.0	6/30/2014	7/31/2014	8/31/2014	
1	7001865778	109	0.0	0.0	0.0	6/30/2014	7/31/2014	8/31/2014	
2	7001625959	109	0.0	0.0	0.0	6/30/2014	7/31/2014	8/31/2014	
3	7001204172	109	0.0	0.0	0.0	6/30/2014	7/31/2014	8/31/2014	
4	7000142493	109	0.0	0.0	0.0	6/30/2014	7/31/2014	8/31/2014	

In [4]:
```
df.shape
```

Out[4]:
```
(99999, 226)
```

In [5]:
```
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.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

Dealing with Missing value

Deaing with Missing value

In [9]:
```
## Handeling missing value
df_missing_columns = (round(((df.isnull().sum()/len(df.index))*100),2).to_frame('null')).sort_values('null', ascending=False)
df_missing_columns
```

Out[9]:

	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
...	...
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 × 1 columns

Dealing with Missing value

 # Columns with more than 30% missing values
col_list_missing_30 =

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

```
(99999, 226)
```

 # Delete the columns with more than 30% missing values

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

```
df.shape
(99999, 186)
```

#removing the date columns because they are not needed for 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)

```
df.shape
(99999, 178)
```

##Dropping the circle_id column.It only has one distinct value. So, this column will have no effect on the data analysis.
 # Drop circle_id column

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

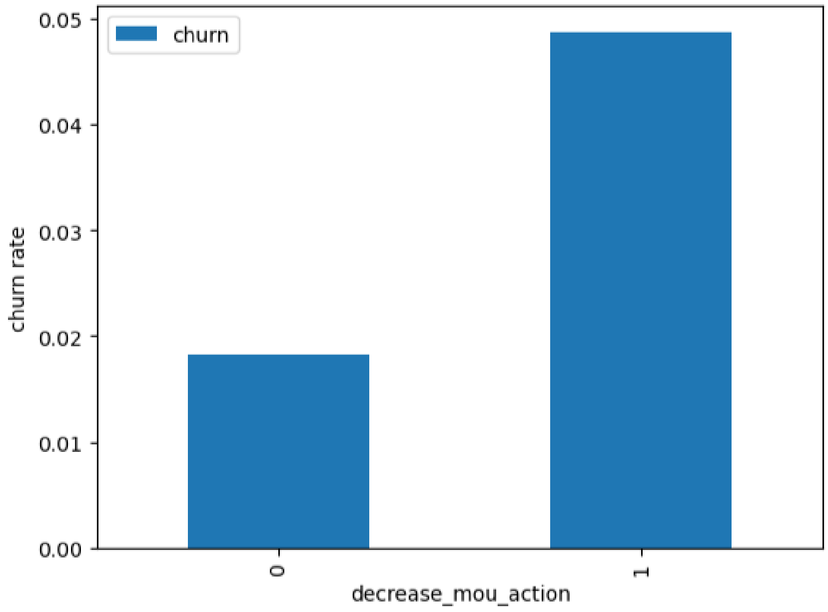
```
df.shape
(99999, 177)
```

#70th percentile of the avg_rech_amt_6_7
X =
_df['avg_rech_amt_6_7'].quantile(0.7)
X

```
368.5
```

EDA Univariate analysis

```
In [87]: data.pivot_table(values='churn', index='decrease_mou_action', aggfunc='mean').plot.bar()
         plt.ylabel('churn rate')
         plt.show()
```

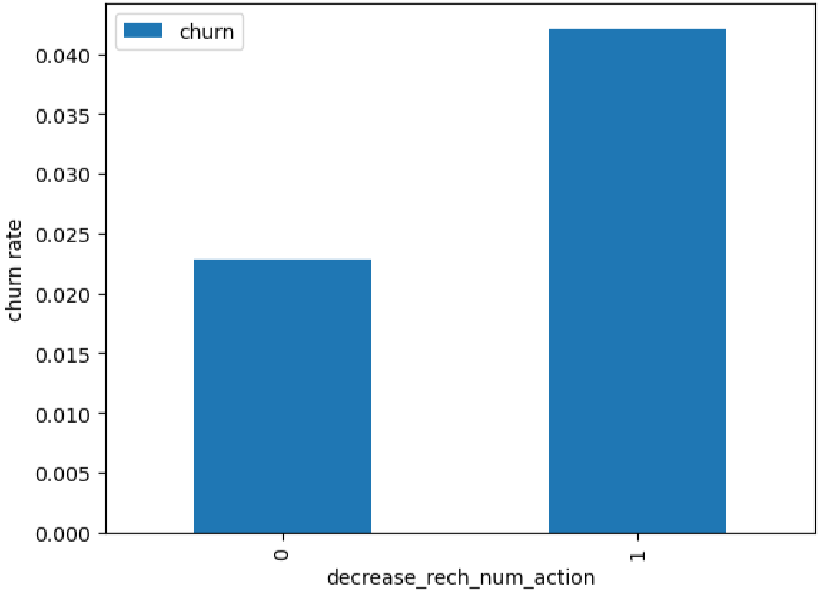


Finding

#We can see that the consumers with lower minutes of usage (mou) during the action phase as opposed to the good phase have a higher churn rate.

#Churn rate determined by the number of recharges a client made within a given month.

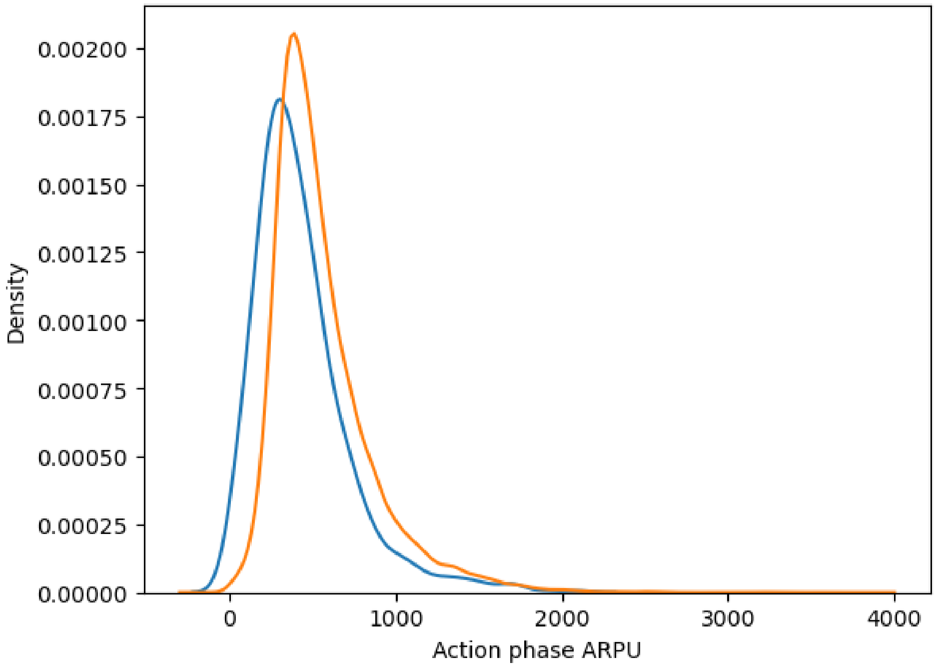
```
In [89]: data.pivot_table(values='churn', index='decrease_rech_num_action', aggfunc='mean').plot.bar()
         plt.ylabel('churn rate')
         plt.show()
```



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

In [93]:
```
# 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')
```

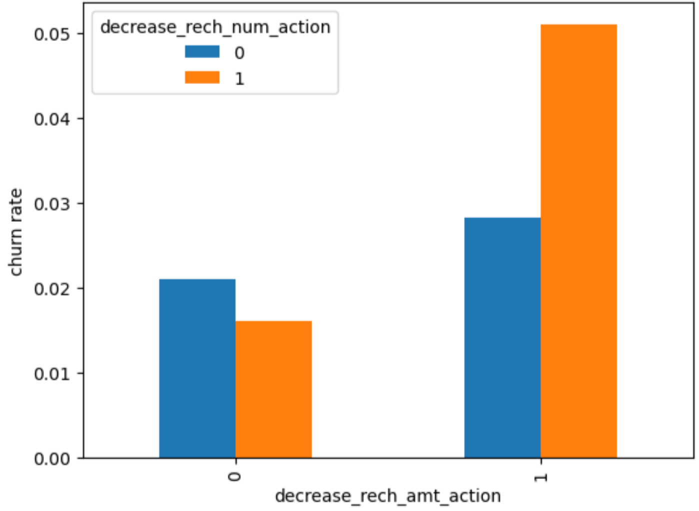
Out[93]:
```
[Text(0.5, 0, 'Action phase ARPU')]
```



Analysis (Above)

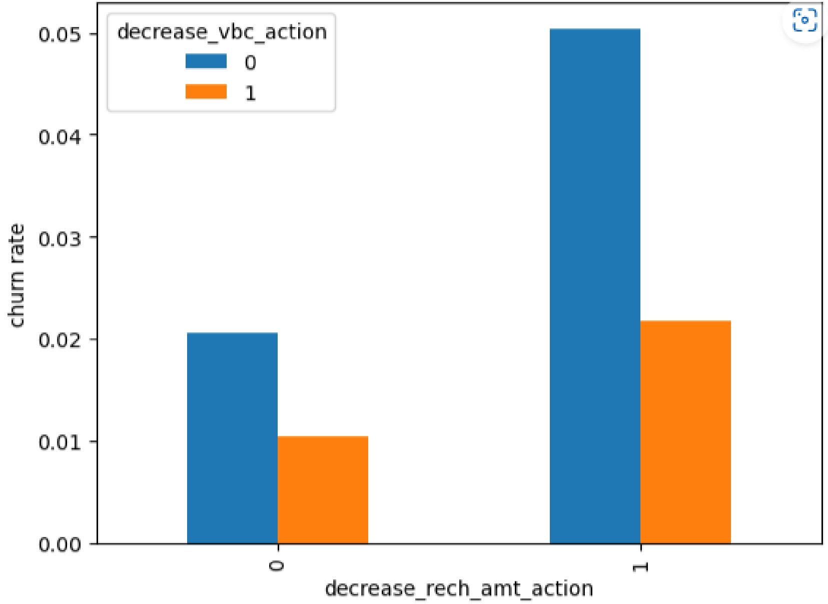
Average revenue per user (ARPU) for the churned customers is mostly densed on the 0 to 900. The higher ARPU customers are less likely to be churned. ARPU for the not churned customers is mostly densed on the 0 to 1000. Analysis of the minutes of usage MOU (churn and not churn) in the action phase

```
In [96]: 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 of churn rate by the decreasing recharge amount and volume based cost in the action phase

```
In [98]: data.pivot_table(values='churn', index='decrease_rech_amt_action', columns='decrease_vbc_action', aggfunc='mean').plot.bar()
plt.ylabel('churn rate')
plt.show()
```



data.head()

In [99]:
```
#Analysis (Above)
#We can see that the churn rate is higher in this instance as well for consumers whose recharge amounts are reduced as the volume
#Analysis of the recharge amount and recharge rate in the current month
```

In [100]:
```
data.head()
```

Out[100]:

	mobile_number	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	of
8	7001524846	0.0	0.0	0.0	378.721	492.223	137.362	413.69	351.03	35.08	94.66	
13	7002191713	0.0	0.0	0.0	492.846	205.671	593.260	501.76	108.39	534.24	413.31	
16	7000875565	0.0	0.0	0.0	430.975	299.869	187.894	50.51	74.01	70.61	296.29	
17	7000187447	0.0	0.0	0.0	690.008	18.980	25.499	1185.91	9.28	7.79	61.64	
21	7002124215	0.0	0.0	0.0	514.453	597.753	637.760	102.41	132.11	85.14	757.93	

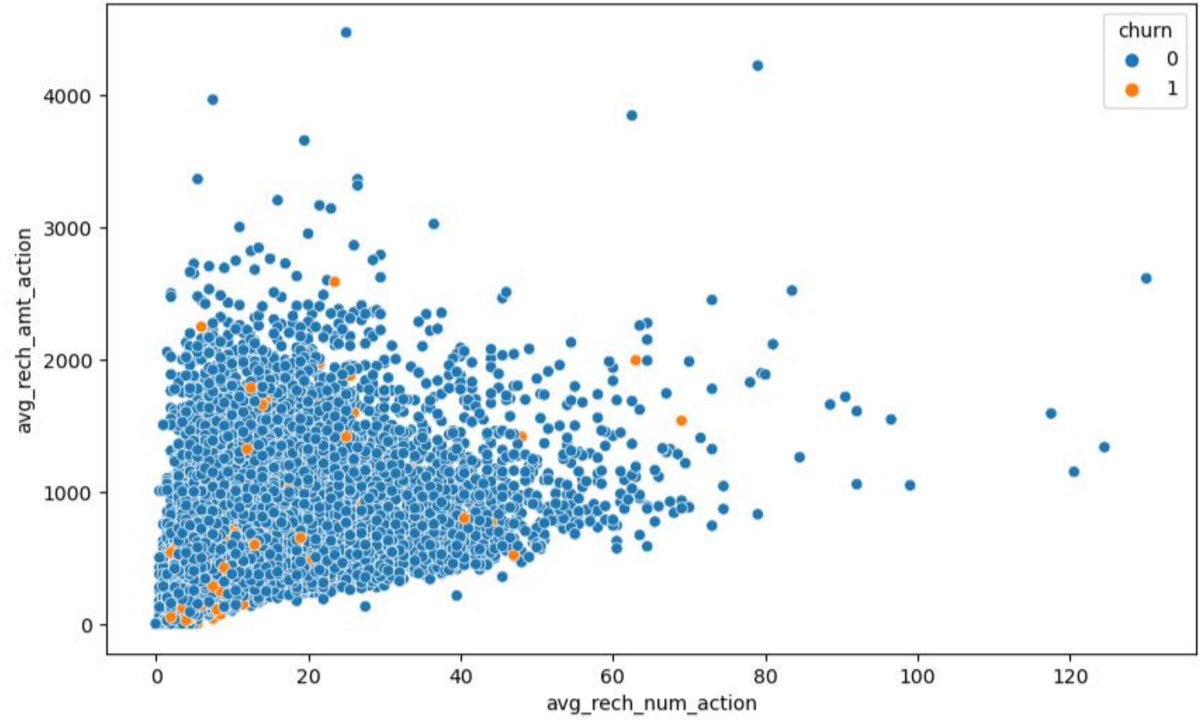
#Analysis (Above)
#We can see that the churn rate is higher in this instance as well for consumers whose recharge amounts are reduced as the volume-based costs rise during the action month.
#Analysis of the recharge amount and recharge rate in the current month

data.head()

```
In [100]: data.head()
```

Out[100]:

	mobile_number	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	of
8	7001524846	0.0	0.0	0.0	378.721	492.223	137.362	413.69	351.03	35.08	94.66	
13	7002191713	0.0	0.0	0.0	492.846	205.671	593.260	501.76	108.39	534.24	413.31	
16	7000875565	0.0	0.0	0.0	430.975	299.869	187.894	50.51	74.01	70.61	296.29	
17	7000187447	0.0	0.0	0.0	690.008	18.980	25.499	1185.91	9.28	7.79	61.64	
21	7002124215	0.0	0.0	0.0	514.453	597.753	637.760	102.41	132.11	85.14	757.93	



Analysis

The pattern demonstrates that the recharge quantity and amount are primarily propotional. The amount of the recharge increases with the number of recharge

Output after from x_train.head()

In [124]:
```
X_train.head()
```

Out[124]:

	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	offnet_mou_7
0	0.0	0.0	0.0	0.140777	-0.522792	-0.276289	0.106540	-0.662084	-0.465777	-0.211202	-0.636415
1	0.0	0.0	0.0	-1.427243	4.428047	3.254270	-0.658491	-0.236590	-0.004450	-0.776075	2.523985
2	0.0	0.0	0.0	-0.222751	0.543206	0.809117	-0.601239	-0.599206	-0.331043	-0.363395	-0.495976
3	0.0	0.0	0.0	-0.911173	0.842273	0.731302	-0.702232	-0.650471	-0.458464	-0.789784	-0.654483
4	0.0	0.0	0.0	0.271356	0.247684	1.256421	-0.356392	-0.180394	0.114727	0.899204	0.904465

In [125]:
```
##Scaling the test set
#We don't fit scaler on the test set. We only transform the test set.
```

In [126]:
```
# Transform the test set
X_test[cols_scale] = scaler.transform(X_test[cols_scale])
X_test.head()
```

Out[126]:

	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	offnet_mou
5704	0.0	0.0	0.0	0.244310	-0.268832	1.005890	-0.725286	-0.690223	-0.476634	0.483540	0.3073
64892	0.0	0.0	0.0	0.048359	-0.779609	-0.157969	-0.734066	-0.698072	-0.502219	-0.358555	-0.5777
39613	0.0	0.0	0.0	0.545470	0.184388	1.403349	-0.537110	-0.521615	-0.206890	0.694901	0.4350
93118	0.0	0.0	0.0	0.641508	0.816632	-0.211023	-0.058843	0.029897	-0.155872	-0.148197	-0.1434
81235	0.0	0.0	0.0	3.878627	0.911619	2.745295	4.117829	1.452446	2.809582	-0.002634	-0.2903

Logistic regression

Logistic regression

```
In [129]: # Importing scikit logistic regression module
          from sklearn.linear_model import LogisticRegression

In [130]: # Impoting metrics
          from sklearn import metrics
          from sklearn.metrics import confusion_matrix

In [131]: #Tuning hyperparameter C
          #C is the the inverse of regularization strength in Logistic Regression. Higher values of C correspond to less regularization.

In [132]: # 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
```

A high value of C tells the model to give more weight to the training data. **A lower value of C will indicate the model to give complexity more weight at the cost of fitting the data.**

Logistic regression with optimal C

Logistic regression with optimal C

In [137]:
```
# Instantiate the model with best C
logistic = LogisticRegression(C=best_C)
```

In [138]:
```
# Fit the model on the train set
log_model = logistic.fit(X_train, y_train)
```

In [139]:
```
# Predictions on the train set
y_train_pred = log_model.predict(X_train)
```

In [140]:
```
# Confusion matrix
confusion = metrics.confusion_matrix(y_train, y_train_pred)
print(confusion)
```

```
[[18241  3184]
 [ 1771 19654]]
```

In [141]:
```
TP = confusion[1,1] # true positive
TN = confusion[0,0] # true negatives
FP = confusion[0,1] # false positives
FN = confusion[1,0] # false negatives
```

The standard cutoff is 0.5, which means that if the predicted probability is greater than 0.5, that observation is classified as a "positive" (or simply as a

```
In [143]: # Prediction on the test set
          y_test_pred = log_model.predict(X_test)

In [144]: # Confusion matrix
          confusion = metrics.confusion_matrix(y_test, y_test_pred)
          print(confusion)

          [[4552  796]
           [  43  150]]

In [145]: TP = confusion[1,1] # true positive
          TN = confusion[0,0] # true negatives
          FP = confusion[0,1] # false positives
          FN = confusion[1,0] # false negatives

In [146]: # 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:- 0.8485832882151236
          Sensitivity:- 0.7772020725388601
          Specificity:- 0.8511593118922962
```

The last code line of logistic regression with optimal c

Model summary

Model summary. The model summary table reports the strength of the relationship between the model and the dependent variable. R, the multiple correlation coefficient, is the linear correlation between the observed and model-predicted values of the dependent variable. Its large value indicates a strong relationship.

Model summary

Train set
Accuracy = 0.88
Sensitivity = 0.92
Specificity = 0.85

Test set
Accuracy = 0.84
Sensitivity = 0.77
Specificity = 0.85

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

Model summary

Train set

Accuracy = 0.84 Sensitivity = 0.88 Specificity = 0.80

Test set

Accuracy = 0.95 Sensitivity = 0.98 Specificity = 0.91

The model's performance shows that the sensitivity declined when it was being assessed on the test set. The test set's precision and specificity, however, are fairly high.

Recomendations

1.Target the clients that use fewer minutes for incoming local calls and outgoing ISD calls during the action phase (mostly in August).

2.Additionally, clients that experience greater value-based costs throughout the action phase are more likely to leave than other customers. Therefore, making an offer to these customers may be a good idea.

3.Customers with higher August monthly 3G recharges are more likely to be churned.

4.Customers who used fewer STD inbound minutes on fixed T lines from operators T in August are more likely to churn. 5.Customers that use less 2G data each month in August are more likely to churn.

6.Customers who used less incoming minutes on fixed T lines from operators in August are more likely to churn.

7.Variables in roam_og_mou_8 have positive coefficients (0.7135). Customers who are using more roaming outbound minutes are hence more prone to churn.

A recommendation system is a subclass of Information filtering Systems that seeks to predict the rating or the preference a user might give to an item. In simple words, it is an algorithm that suggests relevant items to users