Telecom Churn Case Study

Business Problem Overview

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

Business Objective

The dataset contains customer-level information for a span of four consecutive months - June, July, August & September. The months are encoded as 6, 7, 8 and 9, respectively. The business objective is to predict the churn in the last (i.e. the ninth) month using the data (features) from the first three months. To do this task well, understanding the typical customer behaviour during churn will be helpful.

Procedure-

The analysis is divided into nine main parts:

- Data Sourcing or Data Understanding
- Data cleaning and Derived Metrics
- Exploratory Data Analysis (Univariate, Bivariate Analysis)
- Model Preparation
  - Train and Test Data Split
  - Data Normalization
  - Handling Class Imbalance
- Model Building
- Residual Analysis of the Train Data
- Making Predictions
- Model Evaluation
- Final Inference

Data Collection

1. Customers who left within the last month - the column is called Churn
2. Services that each customer has signed up for - phone, multiple lines, internet, online

security, online backup, device protection, tech support, and streaming TV and movies

3. Customer account information - how long they've been a customer, contract, payment method, paperless billing, monthly charges, and total charges
4. Demographic info about customers - gender, age range, and if they have partners and Dependents

Prerequisites

- numpy version: 1.16.3
- pandas version: 0.24.2
- matplotlib version: 3.0.3
- seaborn version: 0.9.0
- sklearn version: 0.20.3
- imblearn version: 0.4.3
- xgboost version: 0.82
- lightgbm version: 2.2.3

Data Import

There are a few columns which have more than 70% of data missing. We dropped these columns as they would convey very little information.

There are 34 columns where 70% data is missing. Now this 70% data is based out of the complete data which contains about 1 lac observations. But our objective is to find the churn rate for only **high valued** customers. It has been given that after filtering high-valued customers we would be left with about 29.9k rows. So dropping the variables now might not be a good idea.

Data Preparation

1. **New Feature Creation**

We have values for 'total_rech_data_' and 'av_rech_amt_data_' (for months 6, 7, 8 & 9). Using these 2 values we can derive new feature for the respective months called **total_data_rech_amt_** which equals **total_rech_data_ * av_rech_amt_data_**

Also this new feature helps in computing the total data recharge amount - 'total_data_rech_amt_' for the months 6, 7, 8 & 9.

. 2. Filter high-value customers

We predict churn only for the high-value customers. Define high-value customers as follows: Those who have recharged with an amount more than or equal to X, where X is the 70th percentile of the average recharge amount in the first two months (the good phase).

3. Rename Columns

```
In [19]: # Rename month named vbc columns to format 6,7,8 and 9

         telecom_data.rename(columns = {'jun_vbc_3g':'vbc_3g_6',
                                        'jul_vbc_3g':'vbc_3g_7',
                                        'aug_vbc_3g':'vbc_3g_8',
                                        'sep_vbc_3g':'vbc_3g_9'}, inplace=True)
```

4. Tag churners and remove attributes of the churn phase

Now to tag 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. The attributes you need to use to tag churners are:

- total_ic_mou_9
- total_og_mou_9
- vol_2g_mb_9
- vol_3g_mb_9

Data Cleaning & Missing Values Treatment

All the columns have more than 40% missing values one way or the other related to data/mobile internet usage for the months 6, 7 & 8 respectively. One important observation that comes out is that for a particular month all these values have the same number of missing values. The below data tells us the same -

Count of missing values for Month 6 - count_rech_2g_6 13158 - count_rech_3g_6 13158 - arpu_3g_6 13158 - arpu_2g_6 13158 - night_pck_user_6 13158

Count of missing values for Month 7 - count_rech_2g_7 13905 - count_rech_3g_7 13905 - arpu_3g_7 13905 - arpu_2g_7 13905 - night_pck_user_7 13905

Count of missing values for Month 8 - count_rech_2g_8 14630 - count_rech_3g_8 14630 - arpu_3g_8 14630 - arpu_2g_8 14630 - night_pck_user_8 14630

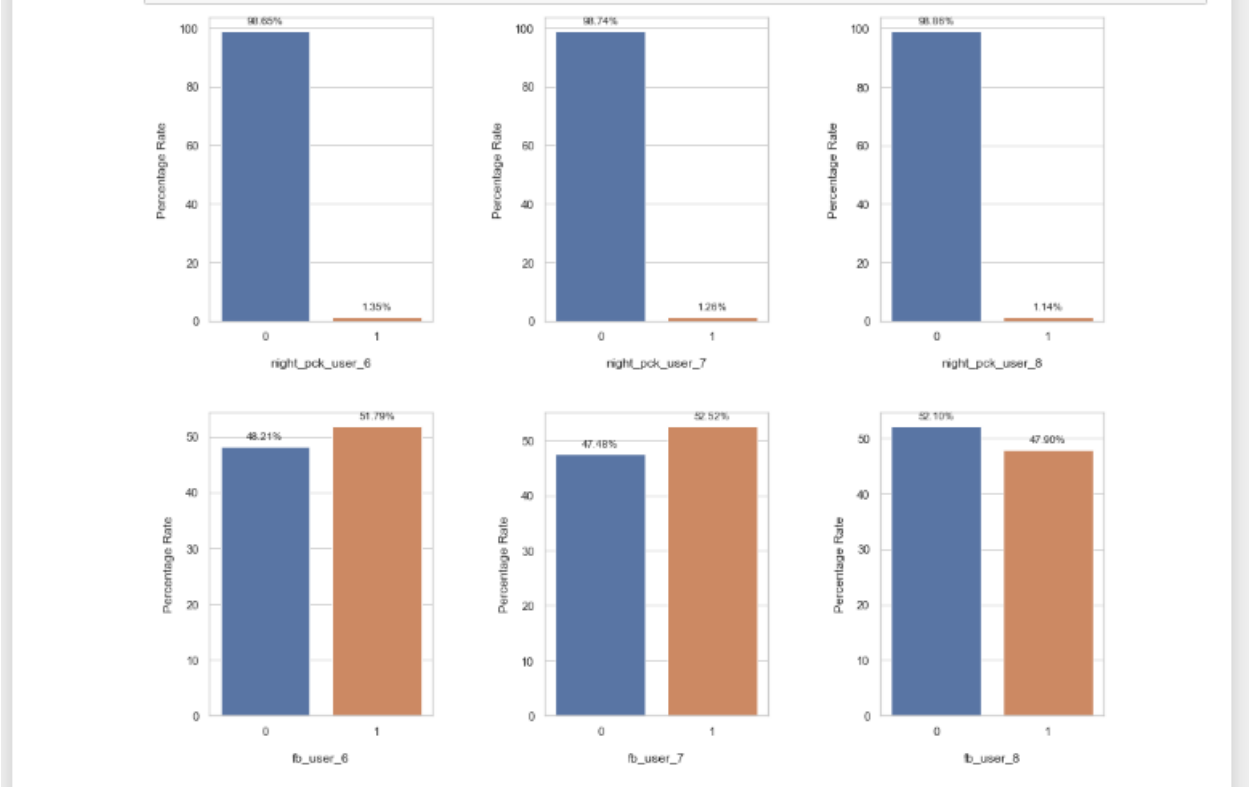
Average Revenue Per User column gives some important insight. The MINIMUM value for all the arpu related columns are NEGATIVE. Now as per the definition of ARPU which says -

"The average revenue per user is the average billing per customer earned by the telecom company every month".

Now the revenue generated from a user cannot be negative. If a customer is not using any services then the ARPU for the person would be zero (rather that being negative). Now if arpu is negative for any row, then that would mean that is a wrong/corrupt data. It will be of no use to us for analysis

Exploratory Data Analysis

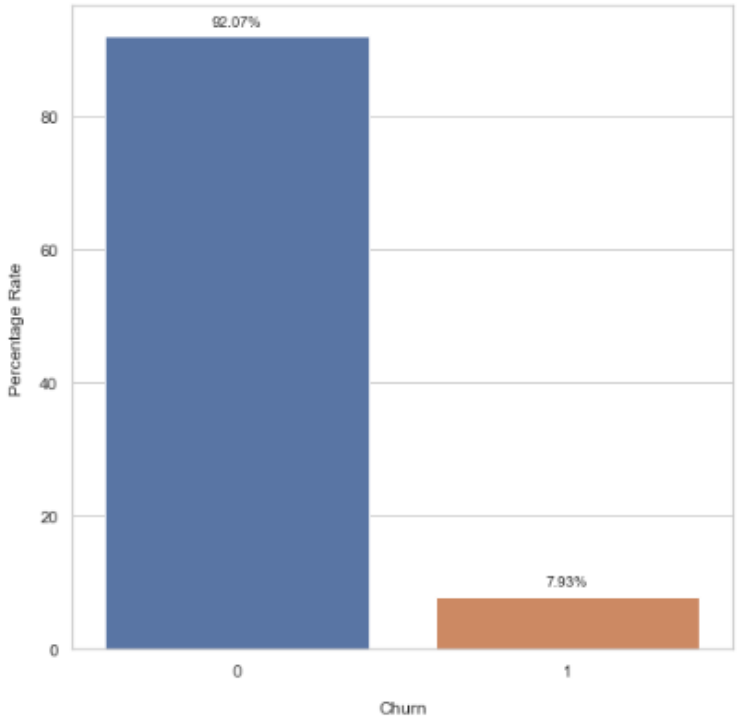
Univariate Analysis - Ordered Categorical Variables



From the `night_pck_user` variables, it can be inferred that almost 99% of users are not using nightly packs in all the 3 months.

However users are using facebook(fb) `fb_user` in equal percentage and it is observed that as the month increases, there is a decline in the fb usage.

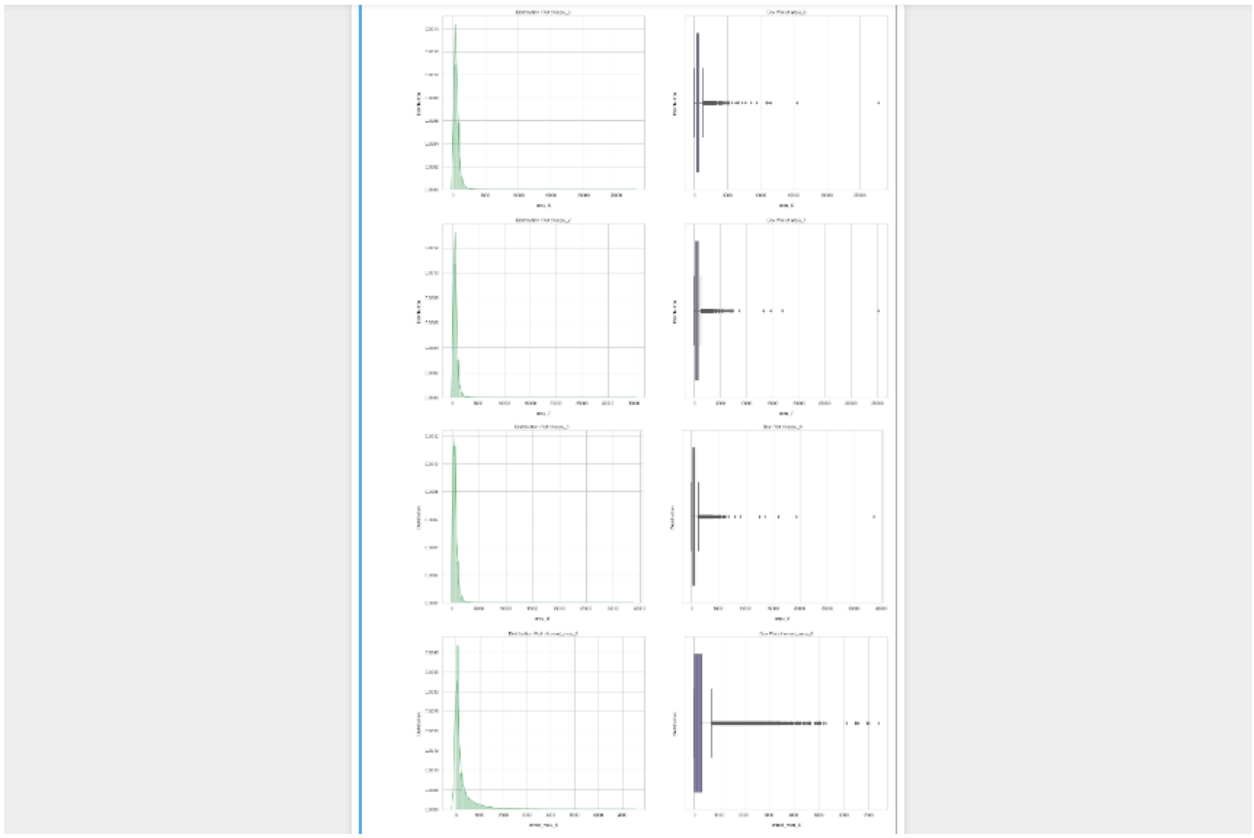
Bar Plot



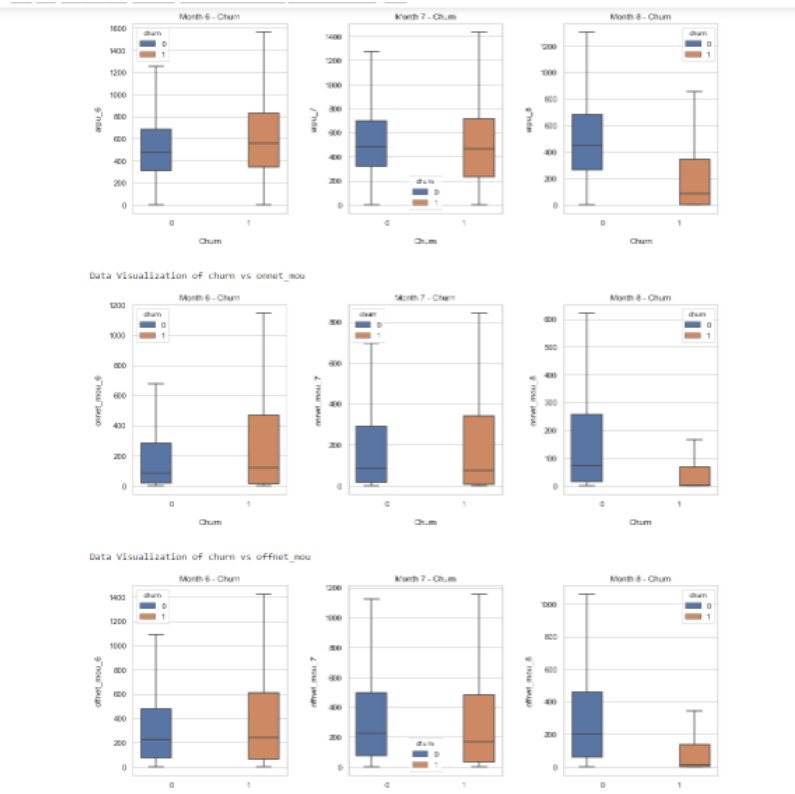
From the above bar plot it is evident that the dataset is highly imbalanced. The proportion for churn to non-churn is around 8%.

For a correct and smooth analysis we need to deal with this class imbalance problem. We will deal with this in a later section after feature engineering.

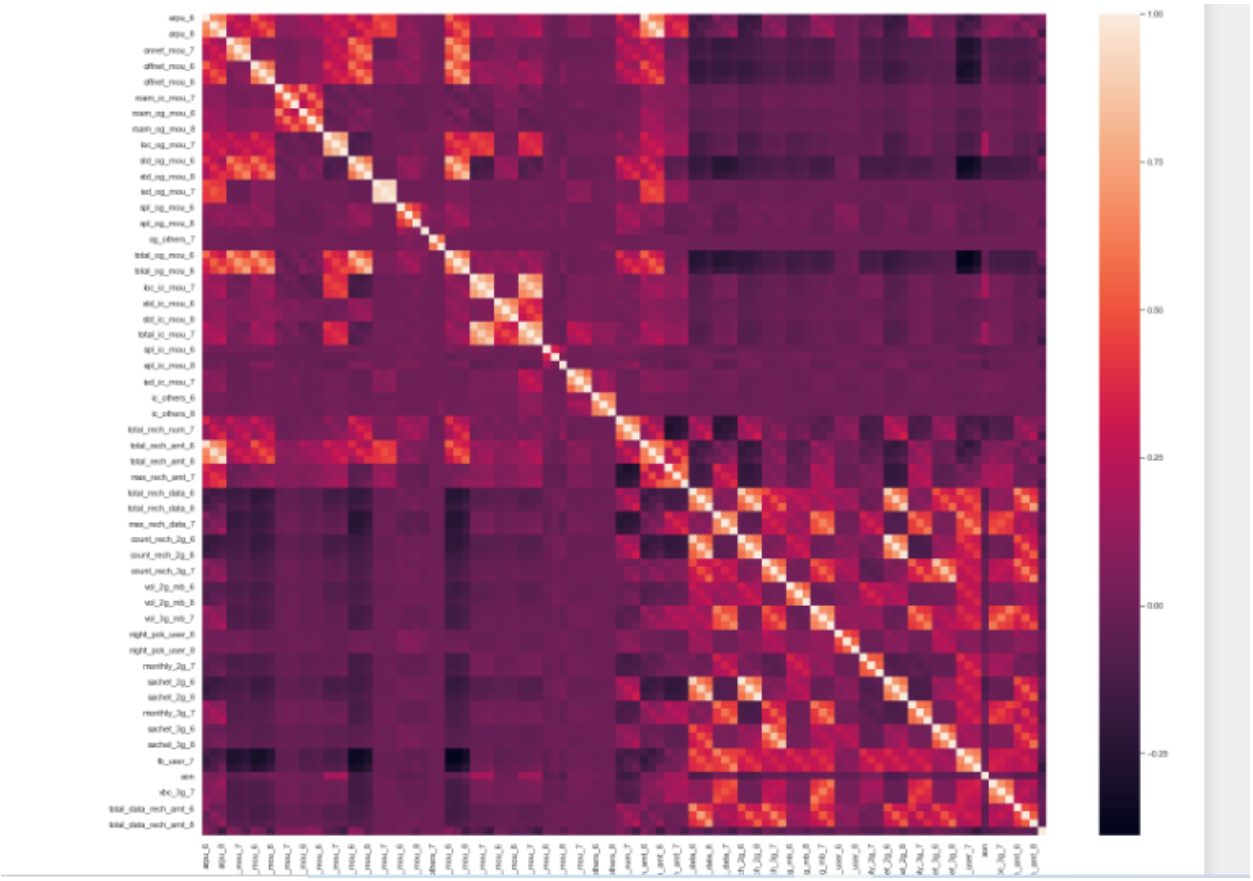
Univariate Analysis - Quantitative Variables



. **Bivariate Analysis - Variables with hue = churn**



Bivariate Analysis - Quantitative Variables



Insight

- There are many features that are highly correlated.
- Total of 25 features with a correlation > 0.80

Note -

- **We will keep these highly correlated features for now. As we will be performing Principal Component Analysis in a later section these should get dropped automatically**

Feature Engineering

Understanding Customer Behavior During Churn

Customers usually do not decide to switch to another competitor instantly, but rather over a period of time (this is especially applicable to high-value customers). In churn prediction, we assume that there are three phases of customer lifecycle :

The ‘good’ phase: In this phase, the customer is happy with the service and behaves as usual.

The ‘action’ phase: The customer experience starts to sore in this phase, for e.g. he/she gets a compelling offer from a competitor, faces unjust charges, becomes unhappy with service quality etc. In this phase, the customer usually shows different behaviour than the ‘good’ months. Also, it is crucial to identify high-churn-risk customers in this phase, since some corrective actions can be taken at this point (such as matching the competitor’s offer/improving the service quality etc.)

The ‘churn’ phase: In this phase, the customer is said to have churned. You define churn based on this phase. Also, it is important to note that at the time of prediction (i.e. the action months), this data is not available to you for prediction. Thus, after tagging churn as 1/0 based on this phase, you discard all data corresponding to this phase.

In this case, since we are working over a four-month window, the first two months are the ‘good’ phase, the third month is the ‘action’ phase, while the fourth month is the ‘churn’ phase.

Model Building

1. Train and Test Split

In [75]:
```
X = telecom_data.drop('churn', axis = 1)
y = telecom_data[['churn']]

X_train, X_test, y_train, y_test = train_test_split(X, y, train_size = 0.7, test_size = 0.3, random_state = 100)

print(X_train.shape)
print(y_train.shape)
print(X_test.shape)
print(y_test.shape)
```

```
(20876, 71)
(20876, 1)
(8948, 71)
(8948, 1)
```

2. Data Normalization (Outlier Treatment)

In [76]:
```
# Normalize the data
scaler = RobustScaler()

scaled_data = scaler.fit_transform(X_train)

X_train = pd.DataFrame(data = scaled_data, index = X_train.index, columns = X_train.columns)
X_test = pd.DataFrame(data = scaler.transform(X_test), index = X_test.index, columns = X_test.columns)
```

Handling Imbalanced Dataset

Pre-Processing Techniques

As a part of the pre-processing stage of ML pipelines prior, the following algorithms will be used for handling an imbalanced dataset.

1. Undersampling
- Random undersampling
2. Oversampling
- Random oversampling: generates new samples by random resampling with replacement of under represented class
- Synthetic Minority Oversampling (SMOTE)
3. Combined over and under sampling
- SMOTEENN
- SMOTETomek

Training techniques

Number of learning models themselves do provide some built in support to deal with imbalance data.

1. Sample weighting

Fact:

SMOTE allows you to generate samples. However, this method of over-sampling does not have any knowledge regarding the underlying distribution. Therefore, some noisy samples can be generated, e.g. when the different classes cannot be well separated. Hence, it can be beneficial to apply an under-sampling algorithm to clean the noisy samples. Imbalanced-learn provides two ready-to-use combined samplers:

- SMOTETomek
- SMOTEENN

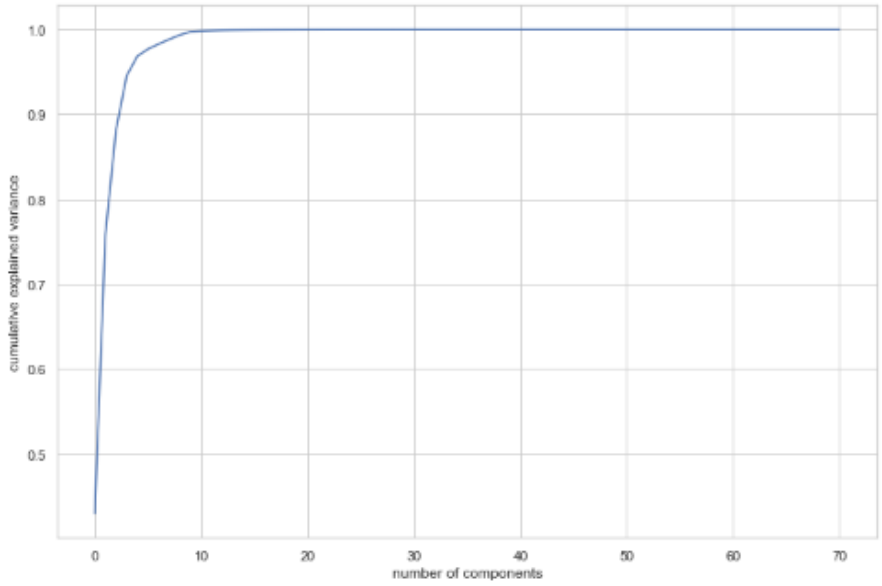
Both the methods are good but in general, SMOTEENN cleans more noisy data than SMOTETomek.

Note:

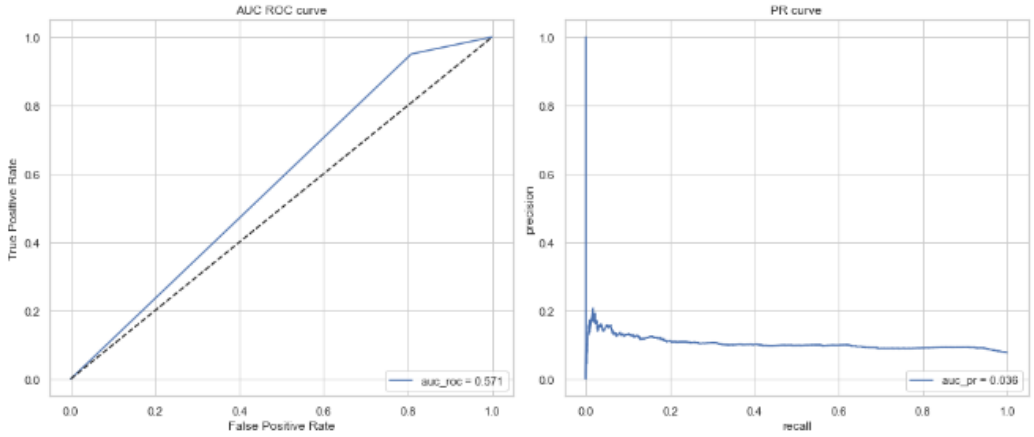
- **It is not possible to check different sampling techniques on very cost sensitive Machine Learning models like SVM, Decision Trees, Random Forest. For this Case Study, we will particularly use SMOTEENN sampling technique to handle an imbalanced dataset as it uses both oversampling and under-sampling methods and helps in cleaning noisy samples.**

Model Building

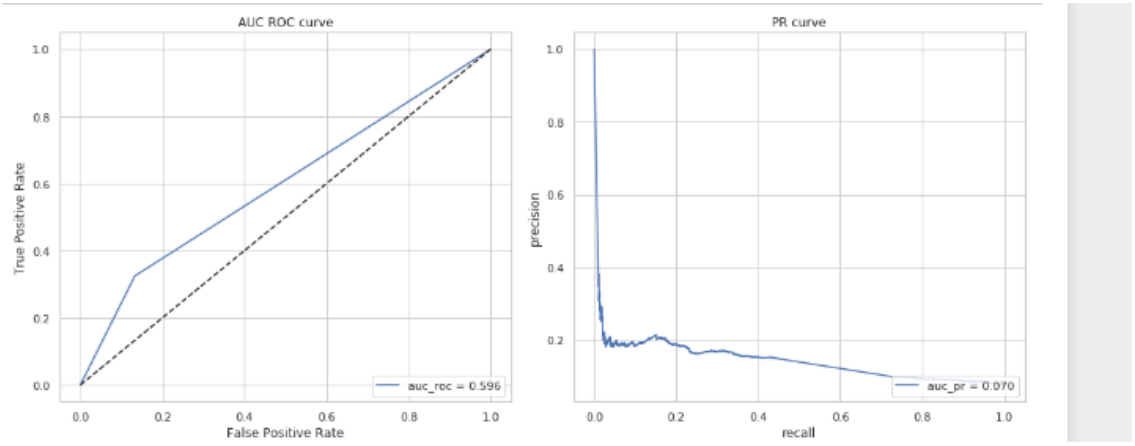
1. PCA - Dimensionality Reduction



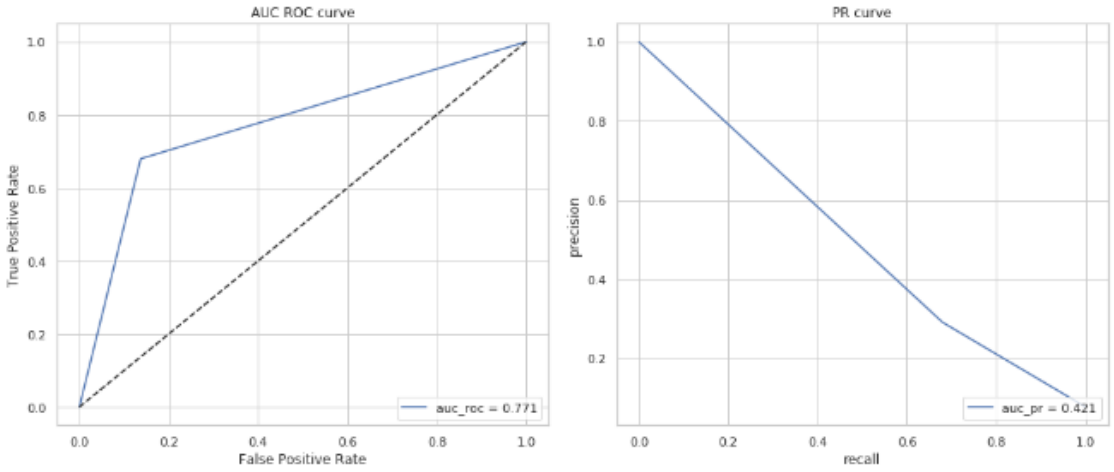
2. Logistic Regression



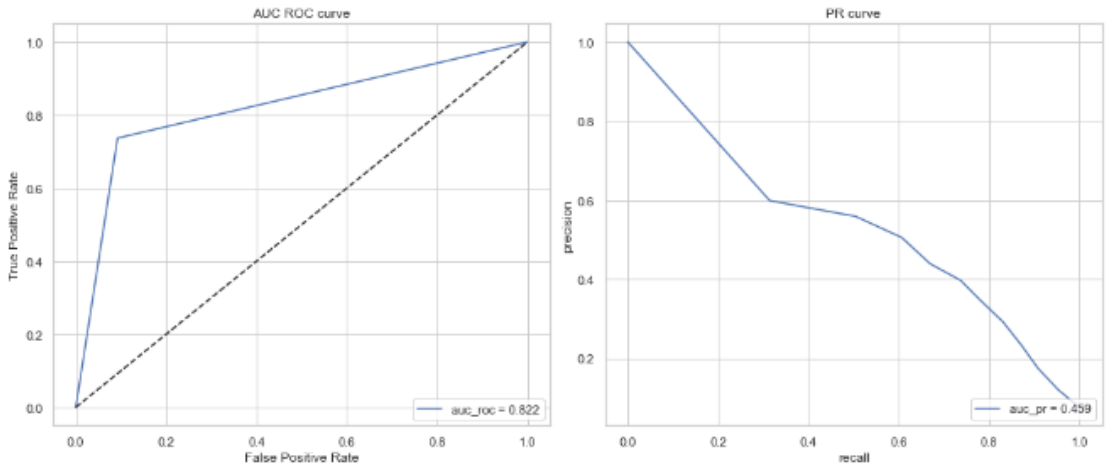
3. Support Vector Machine (SVM)



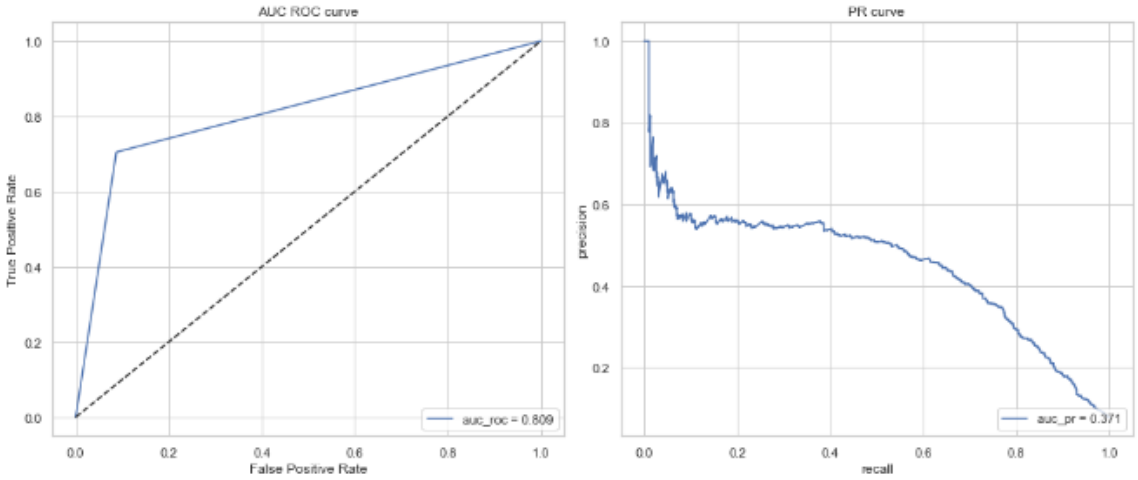
4. Decision Tree



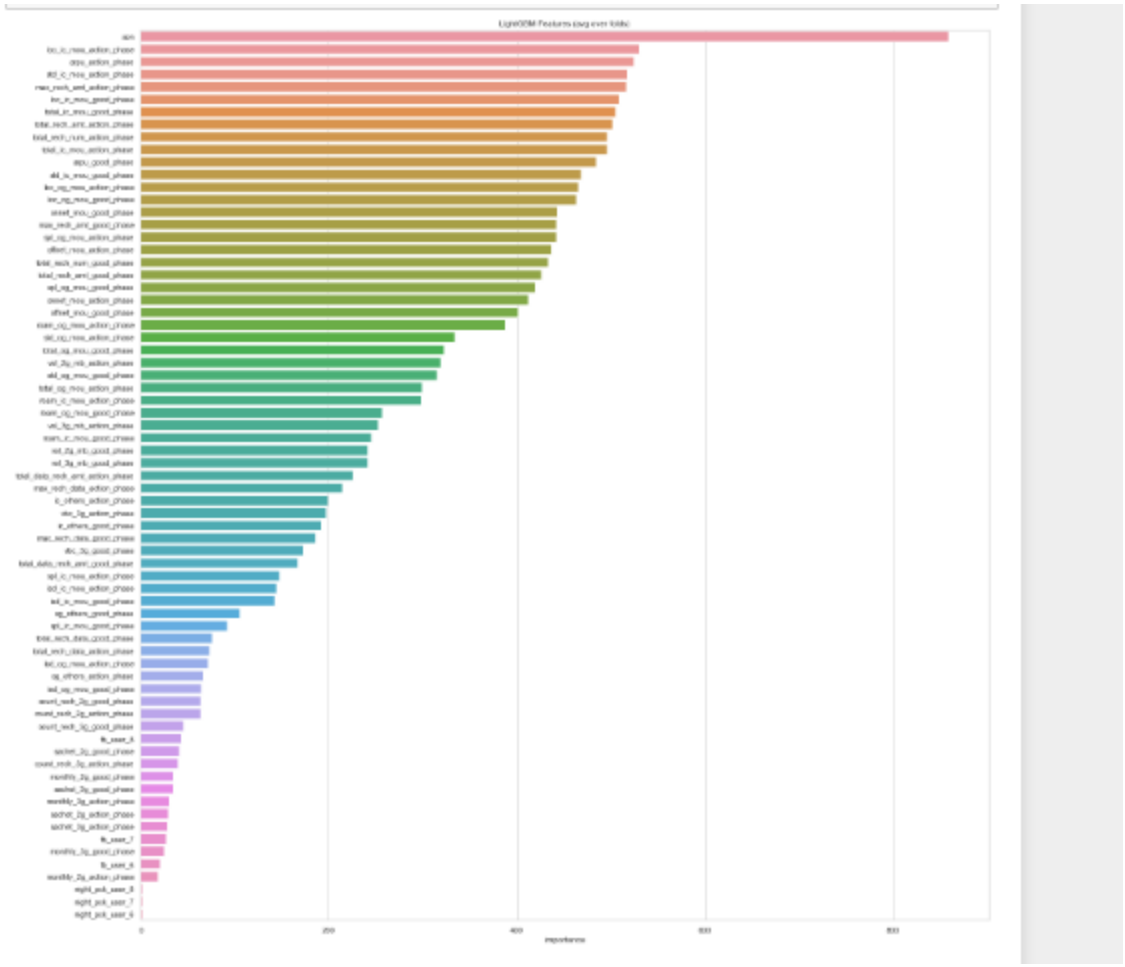
5. Random Forest



6. Adaboost



Important Features:



Final Inference

The telecom churn analysis is required to predict the customer churn behavior i.e., if a particular customer is likely to churn or not churn. As the problem preview said -

"In this highly competitive market, the telecommunications industry experiences an average of 15-25% annual churn rate. Given the fact that it costs 5-10 times more to acquire a new customer than to retain an existing one, customer retention has now become even more important than customer acquisition."

Thus to predict or analyze the behavior of a customer over a time period of good phase and the action phase we created about 4 predictive models as mentioned below -

1. Logistic Regression Model.
2. Support Vector Machine.
3. Decision Tree Classifier.
4. Random Forest Classifier.

The above models were initially created with default parameters which did not give accurate results and the score metrics were not good. Then we hypertuned each model and recreated them with the **best estimators**. The hyper tuned model showed an increase in the classification scores though marginally. These scores were still not good enough to say the model were good enough.

Each of the above models were assessed on area under the curve, precision, recall.

These models did not fare that well and were not able to classify accurately.

We then applied the following **Boosting mechanisms**.

1. AdaBoostClassifier.
2. Extreme Gradient Boosting with XGBoost.
3. Light Gradient Boosting Machine with LightGBM.

The above boosting models were initially created with default parameters which performed better than any of the logistic regression model , SVM model, the decision tree or the random forest classifier. The classification scores and metrics shoot up very high.

We then hypertuned each boosting model and recreated them with the **best estimators**. The hyper tuned model again performed better than their default counterpart. There was an increase in the classification scores.

Winner

Out of all the above models, **Light Gradient Boosting Machine with LightGBM** with RandomOverSampler Technique came out as the winner giving the best metrics of scores. Let's see the summary of scores for the LightGBM model and how it performed.

Summary of Scores

5.2.1 LightGBM - Default v/s Hypertuned Model

- with RandomOverSampling Technique:

	f1	precision	recall	accuracy	auc_roc	auc_pr	confusion_matrix
	0.568236	0.625659	0.520468	0.93954	0.747347	0.49533	[[8051, 213], [328, 356]]

There is a significant improvement in f1, precision, recall, accuracy and auc_roc in the final model of LightGBM (After HyperTuning).

Top 7 Features affecting churn

- aon
- loc_ic_mou_action_phase
- arpu_action_phase

- std_ic_mou_action_phase
- max_rech_amt_action_phase
- loc_ic_mou_good_phase
- total_ic_mou_good_phase

Action phase features are defined above in the derived features section.

Bottom Line

Our LightGBM model is a decent model. We are able to predict with accuracy of **93.95 %** .

If the goal is to engage and talk to the customers to prevent them from churning, its ok to engage with those who are mistakenly tagged as 'not churned,' as it does not cause any negative problems. It could potentially make them even happier for the extra attention they are getting. This is the kind of model that can surely add value.