TELECOM CHURN CASE STUDY

PRESENTED BY

NAVEEN UPADHYAY(DSC-43)

SARATH CHANDRAN (DSC-43)

THIPPESSH H K (DSC-43)

DATE: 06 DEC 22

PROBLEM STATEMENT

- In the telecom industry, customers switch their operators when they come across better deals.
- It costs 5-10 times more to acquire a new customer than to retain an existing one.
- Thus, customer retention is more important than customer acquisition.
- In this project, we predict the churn for pre-paid customers who are predominant in the Indian and Southeast Asian market.
- The business objective is to predict the churn in the ninth month using the data from the first three months.

GOALS OF THE CASE STUDY

- We start with exploratory data analysis, impute null values and drop columns that do not add additional information.

- We then filter for high value customers who have recharged with an amount more than or equal the 70th percentile of the average recharge amount in the first two months.

- We also derived 6 columns. We then prepared the data and applied multiple algorithms to generate a model by hyperparameter tuning.

- We first applied four machine learning algorithms and measured the effectiveness of the models by using ROC AUC score and stratified accuracy scores for Churn and Non Churn Customers.

  1. Basic Logistic Regression
  2. Logistic Regression with PCA
  3. Random Forest with Hyperparameter tuning
  4. SVM with Hyperparameter tuning

SOURCING AND UNDERSTANDING DATA

- Importing the given dataset.
- Converting the dataset into a data frame.
- Understanding the data dictionary.
- Inspecting Data for EDA.
- Performing Data Cleaning.

READING AND EXPLORING THE DATA

- Shape of dataframe: 99999 rows and 226 columns.
- Checking for missing values using info().
- Confirming outliers with describe().
- Checking for duplicates using duplicated().sum().
- Checking for Null values.
- Segregating relevant data and removing irrelevant data.

DATA PREPARATION

- Filter high-value customer.
- Handling columns with higher percentage of Null Values.
- Identify Numerical and Categorical Features.
- Performing Univariate and Multivariate Analysis.
- Imputing data for null value entries.

FEATURES IDENTIFICATION WITH NO SIGNIFICANT VALUE

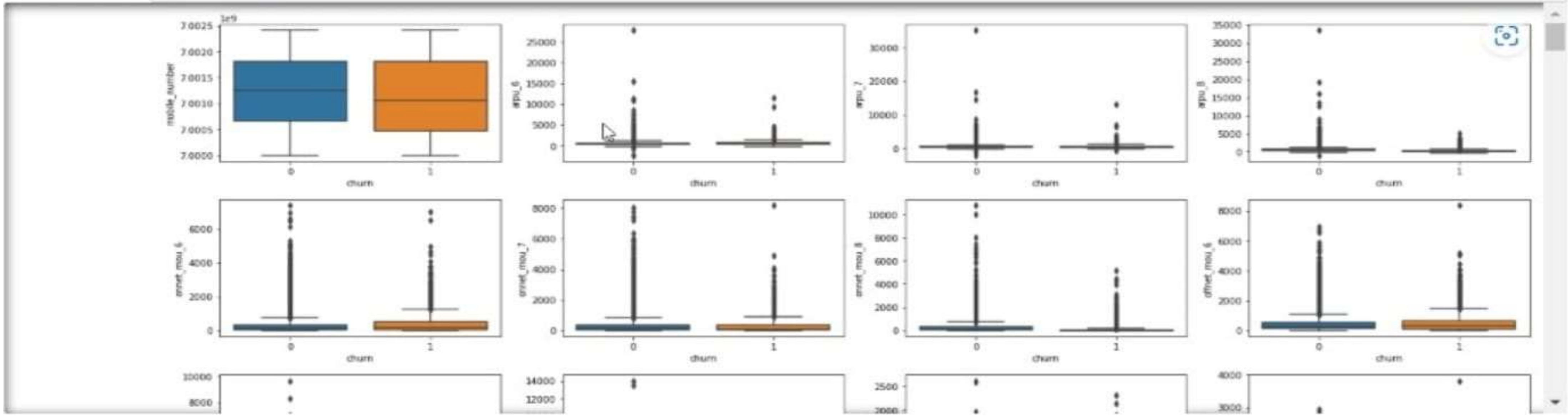
- Analyze for Numerical Variables.
- Analyze for Categorical Variables.
- Other Columns Null Value Treatment.
- Tag churners and remove attributes of the churn phase.

EXPLORATORY DATA ANALYSIS (EDA)

- Analyze for Churn and Non Churn customers distribution.
- Outliers analysis.
- Derive features to obtain percentage of calls for months 6,7,8 & 9.
- Treating Nan values.
- Visualization by plotting Histogram.
- Visualization by plotting Correlation Matrix.
- Visualization by plotting Scatter Plots.
- Visualization by plotting Distribution Plots.

EXPLORATORY DATA ANALYSIS (EDA)


```
In [1223]: fig = plt.figure(figsize=(20,100))
           for i in range(len(numCols)):
               fig.add_subplot(34, 4, i+1)
               sns.boxplot(y=data_high_val_cust.loc[:, numCols].iloc[:,i],x=data_high_val_cust['churn'])
           plt.tight_layout()
           plt.show()
```



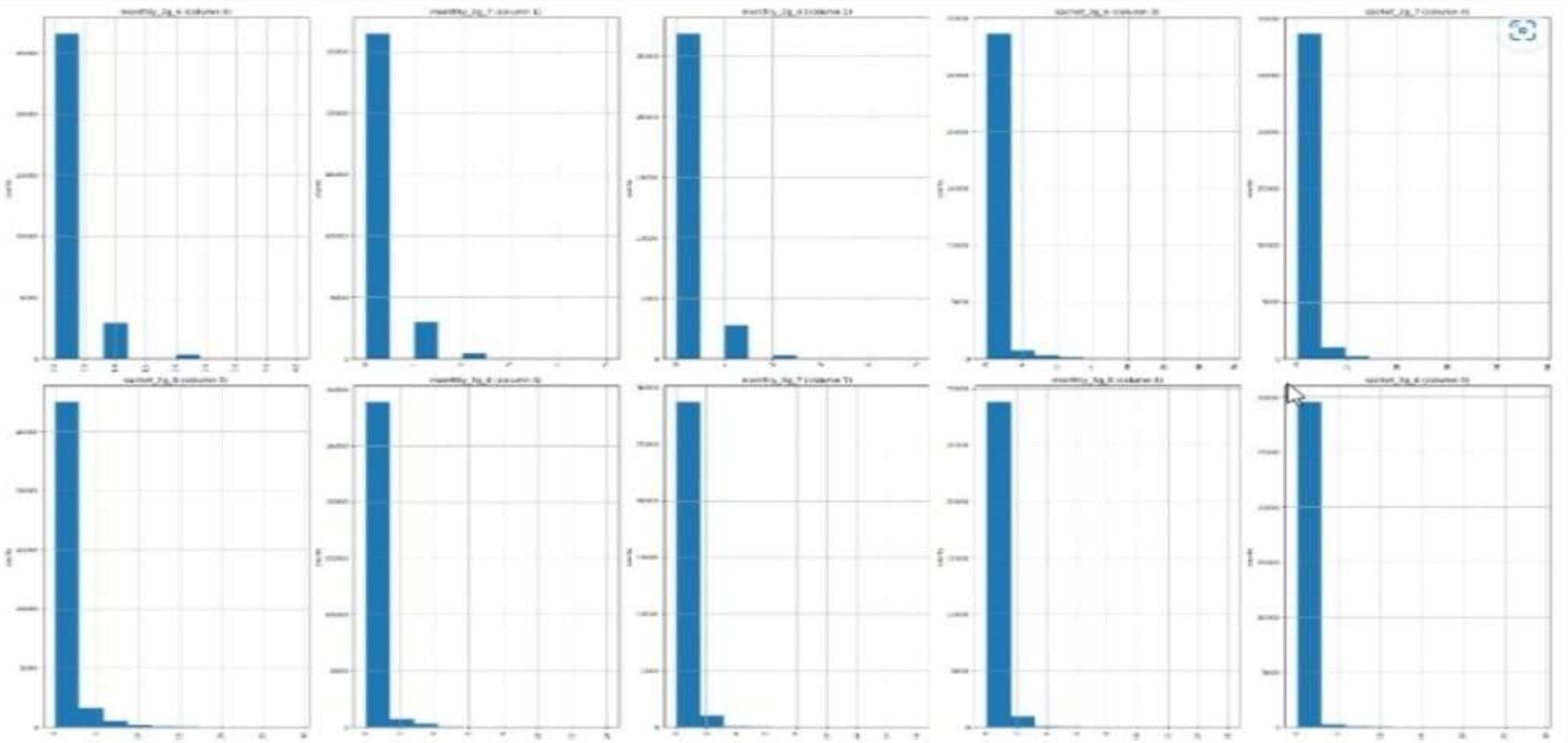
Inference

- Majorily seen under Non-Churn customers data
- As part of outlier treatment, lets drop those outlier data

Outliers analysis using Box Plot.

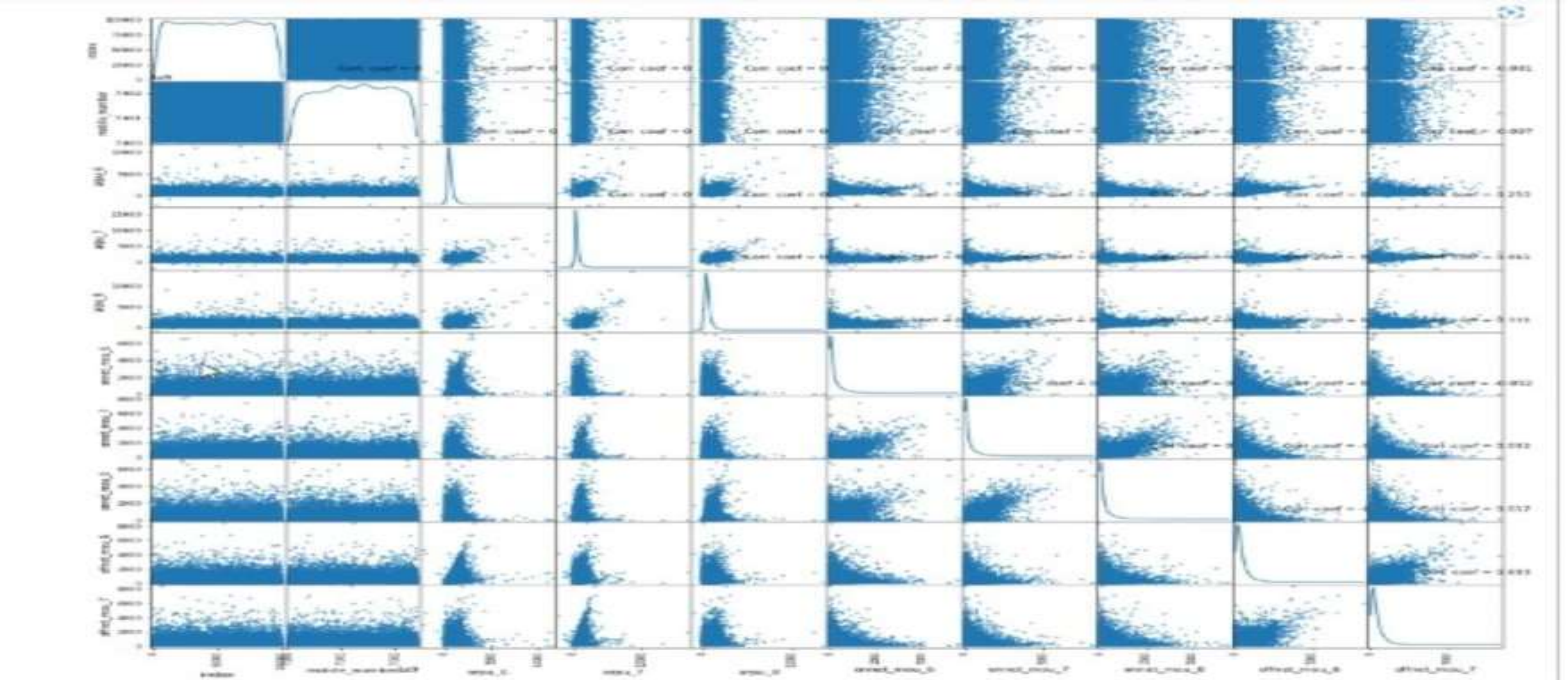
EXPLORATORY DATA ANALYSIS (EDA)


```
In [1243]: #Plottng the Distribution plot
           plotPerColumnDistribution(data_high_val_cust, 10, 5)
```



Visualization by plotting Distribution Plots.

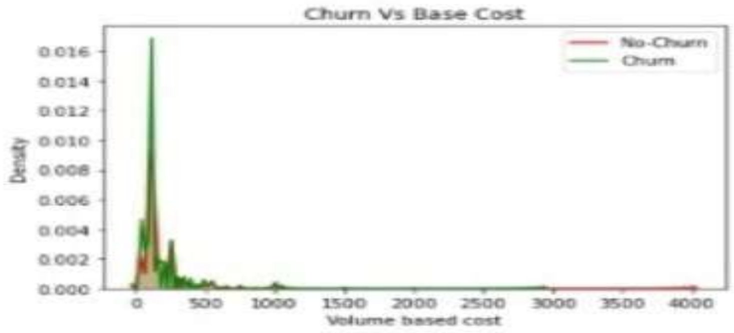
EXPLORATORY DATA ANALYSIS (EDA)



Visualization by plotting Scatter Plots.

EXPLORATORY DATA ANALYSIS (EDA)

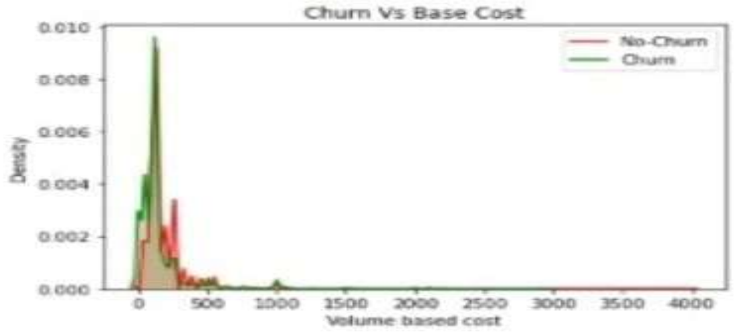
Out[1245]: Text(0.5, 1.0, 'Churn Vs Base Cost')



In [1246]:

```python
# churn Vs Base Cost month 7
ax = sns.kdeplot(data_high_val_cust.max_rech_amt_7[(data_high_val_cust["churn"] == 0)],
                color="Red", shade = True)
ax = sns.kdeplot(data_high_val_cust.max_rech_amt_7[(data_high_val_cust["churn"] == 1)],
                ax =ax, color="Green", shade= True)
ax.legend(["No-Churn","Churn"],loc='upper right')
ax.set_ylabel('Density')
ax.set_xlabel('Volume based cost')
ax.set_title('Churn Vs Base Cost')
```

Out[1246]: Text(0.5, 1.0, 'Churn Vs Base Cost')



Analyze for Churn and Non Churn customers distribution.

PREDICTION MODELING USING LOGISTIC REGRESSION

- Data Standardisation.
- Splitting data into 80% for train and 20% for test
- Obtaining Confusion Matrix
- Plotting ROC (Receiver Operating Characteristics) Curve

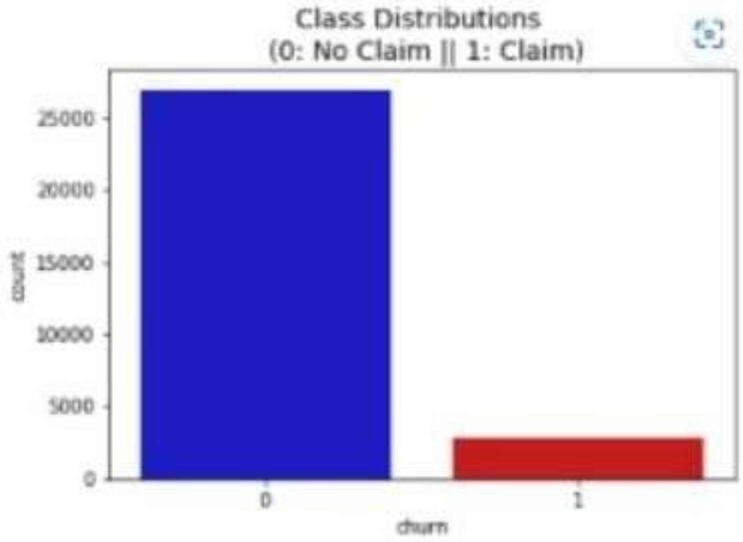
HANDLING IMBALANCED DATA

- Data Standardization.
- Splitting data into 80% for train and 20% for test.
- Obtaining Confusion Matrix.
- Plotting ROC (Receiver Operating Characteristics) Curve.

HANDLING IMBALANCED DATA

Handling Imbalanced Dataset

```
In [1256]: colors = ["#0101DF", "#DF0101"]

           sns.countplot('churn', data=data_high_val_cust, palette=colors)
           plt.title('Class Distributions \n (0: No Claim || 1: Claim)', fontsize=14)

Out[1256]: Text(0.5, 1.0, 'Class Distributions \n (0: No Claim || 1: Claim)')
```

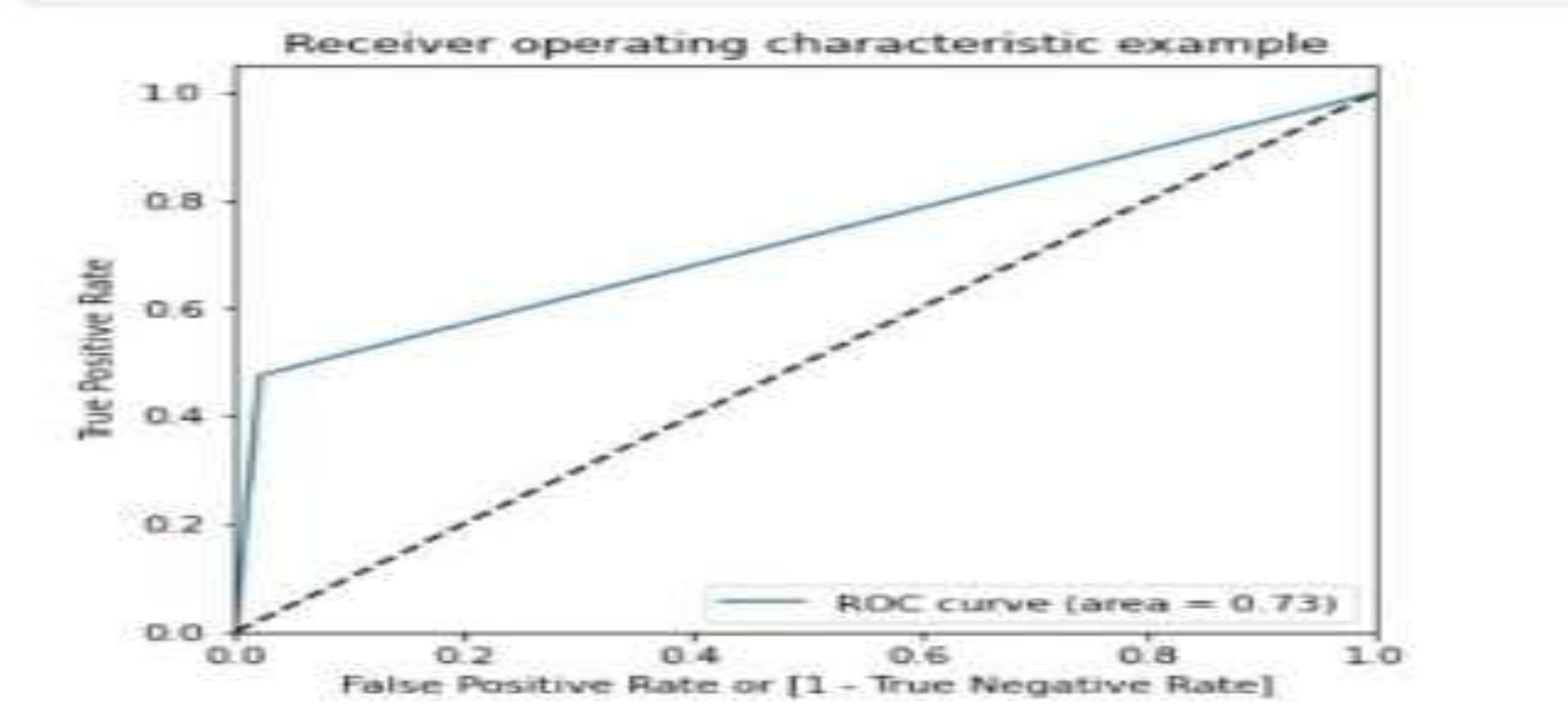


Plotting Class Distribution of Imbalanced Data

PREDICTION POST IMBALANCED TREATMENT

- RANDOM FOREST : SMOTE (SYNTHETIC MINORITY OVERSAMPLING TECHNIQUE)
- PLOTTING ROC CURVES

PREDICTION POST IMBALANCED TREATMENT



Plotting ROC (Receiver Operating Characteristics) Curve

RANDOM FOREST : SMOTE

Random Forest - SMOTE

In [913]:

```
from sklearn.ensemble import RandomForestClassifier
rf_smt = RandomForestClassifier()
rf_smt.fit(X_resampled_smt, y_resampled_smt)

y_pred_rf_smt = rf_smt.predict(X_test)
```

In [914]:

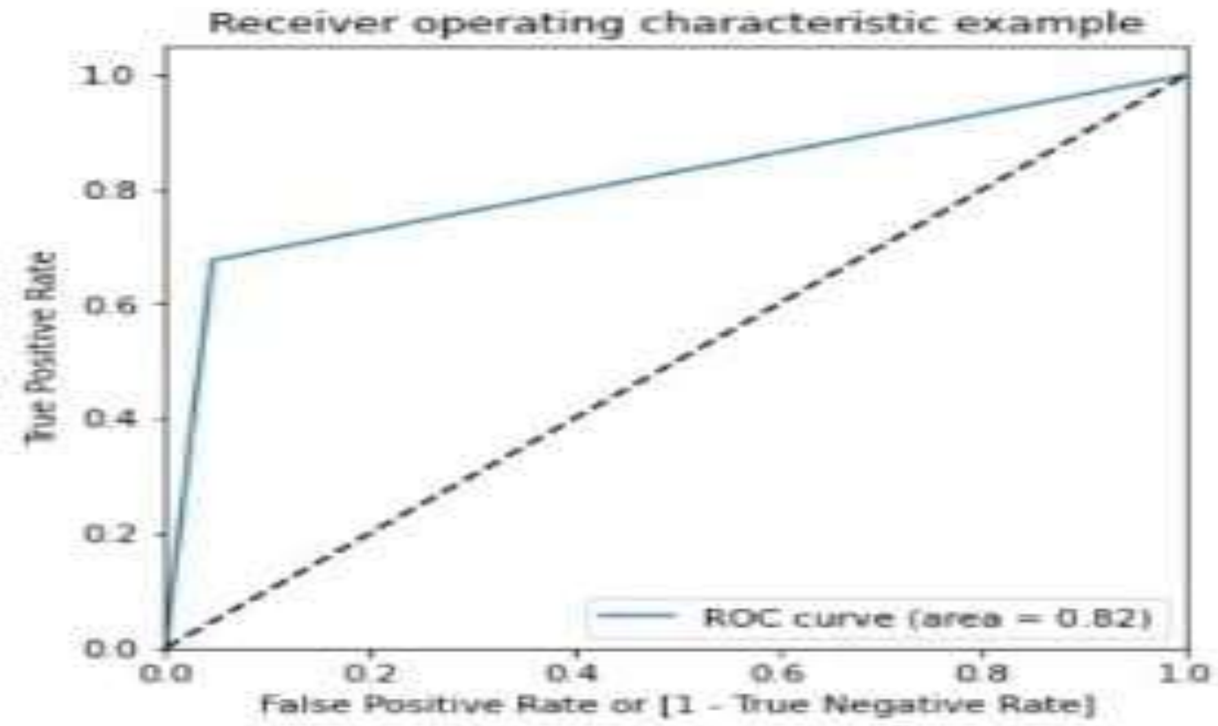
```
print ('Accuracy: ', accuracy_score(y_test, y_pred_rf_smt))
print ('F1 score: ', f1_score(y_test, y_pred_rf_smt))
print ('Recall: ', recall_score(y_test, y_pred_rf_smt))
print ('Precision: ', precision_score(y_test, y_pred_rf_smt))
print ('\n clasification report:\n', classification_report(y_test,y_pred_rf_smt))
print ('\n confussion matrix:\n',confusion_matrix(y_test, y_pred_rf_smt))
```

```
Accuracy:  0.9253931080628973
F1 score:  0.6477093206951027
Recall:  0.6765676567656765
Precision:  0.6212121212121212

 clasification report:
               precision    recall  f1-score   support

           0       0.96      0.95      0.96      5372
           1       0.62      0.68      0.65       606

    accuracy                           0.93      5978
   macro avg       0.79      0.82      0.80      5978
weighted avg       0.93      0.93      0.93      5978


 confussion matrix:
 [[5122  250]
 [ 196  410]]
```

RANDOM FOREST : SMOTE

RANDOM FOREST : SMOTE


```
In [915]: draw_roc(y_test, y_pred_rf_smt)
```



RANDOM FOREST : SMOTE
ROC CURVE

DECISION TREE : SMOTE

Decision Trees - SMOTE

In [916]:
```
from sklearn.tree import DecisionTreeClassifier
dtc_smt = DecisionTreeClassifier(random_state=0)
dtc_smt.fit(X_resampled_smt, y_resampled_smt)

y_pred_dtc_smt = dtc_smt.predict(X_test)
```

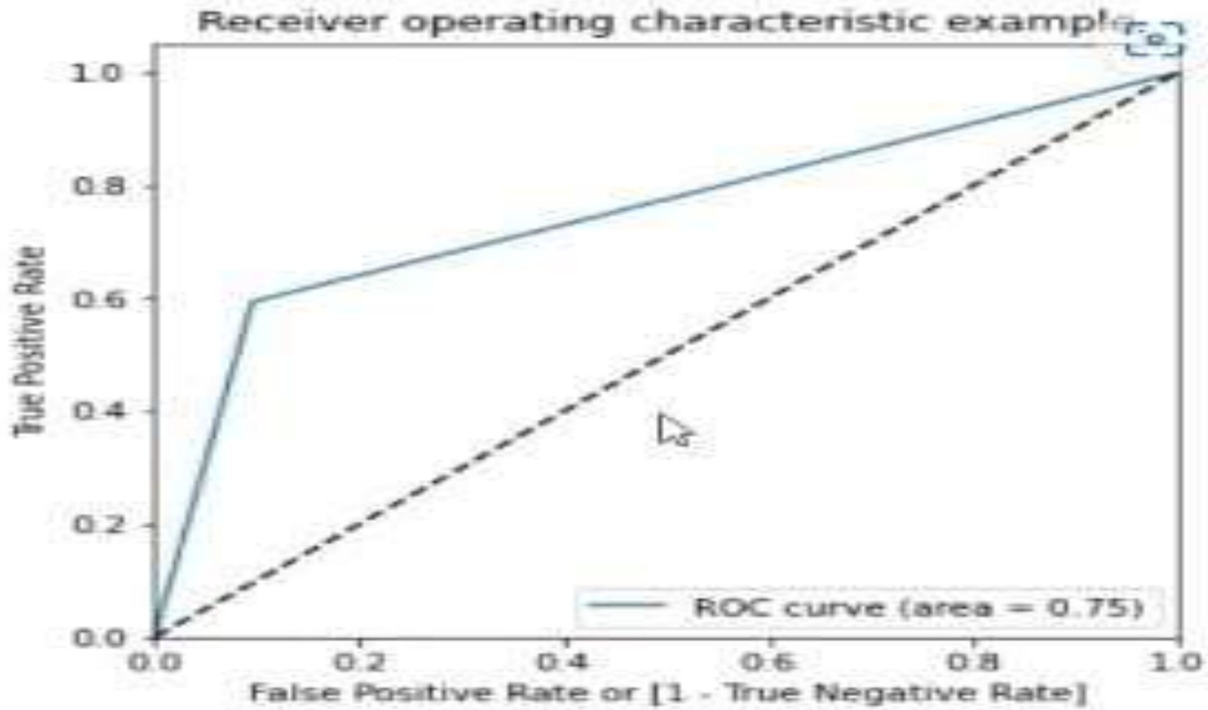
In [917]:
```
print ('Accuracy: ', accuracy_score(y_test, y_pred_dtc_smt))
print ('F1 score: ', f1_score(y_test, y_pred_dtc_smt))
print ('Recall: ', recall_score(y_test, y_pred_dtc_smt))
print ('Precision: ', precision_score(y_test, y_pred_dtc_smt))
print ('\n clasification report:\n', classification_report(y_test,y_pred_dtc_smt))
print ('\n confussion matrix:\n',confusion_matrix(y_test, y_pred_dtc_smt))
```

```
Accuracy:  0.8723653395784543
F1 score:  0.4855023600809171
Recall:  0.594059405940594
Precision:  0.41049030786773O9

 clasification report:
               precision    recall  f1-score   support

           0       0.95      0.90      0.93      5372
           1       0.41      0.59      0.49       606

    accuracy                           0.87      5978
   macro avg       0.68      0.75      0.71      5978
weighted avg       0.90      0.87      0.88      5978


 confussion matrix:
 [[4855  517]
 [ 246  360]]
```

DECISION TREE : SMOTE

DECISION TREE : SMOTE


```
In [918]:    draw_roc(y_test, y_pred_dtc_smt)
```



DECISION TREE : SMOTE
ROC CURVE

FEATURE IMPORTANT ANALYSIS

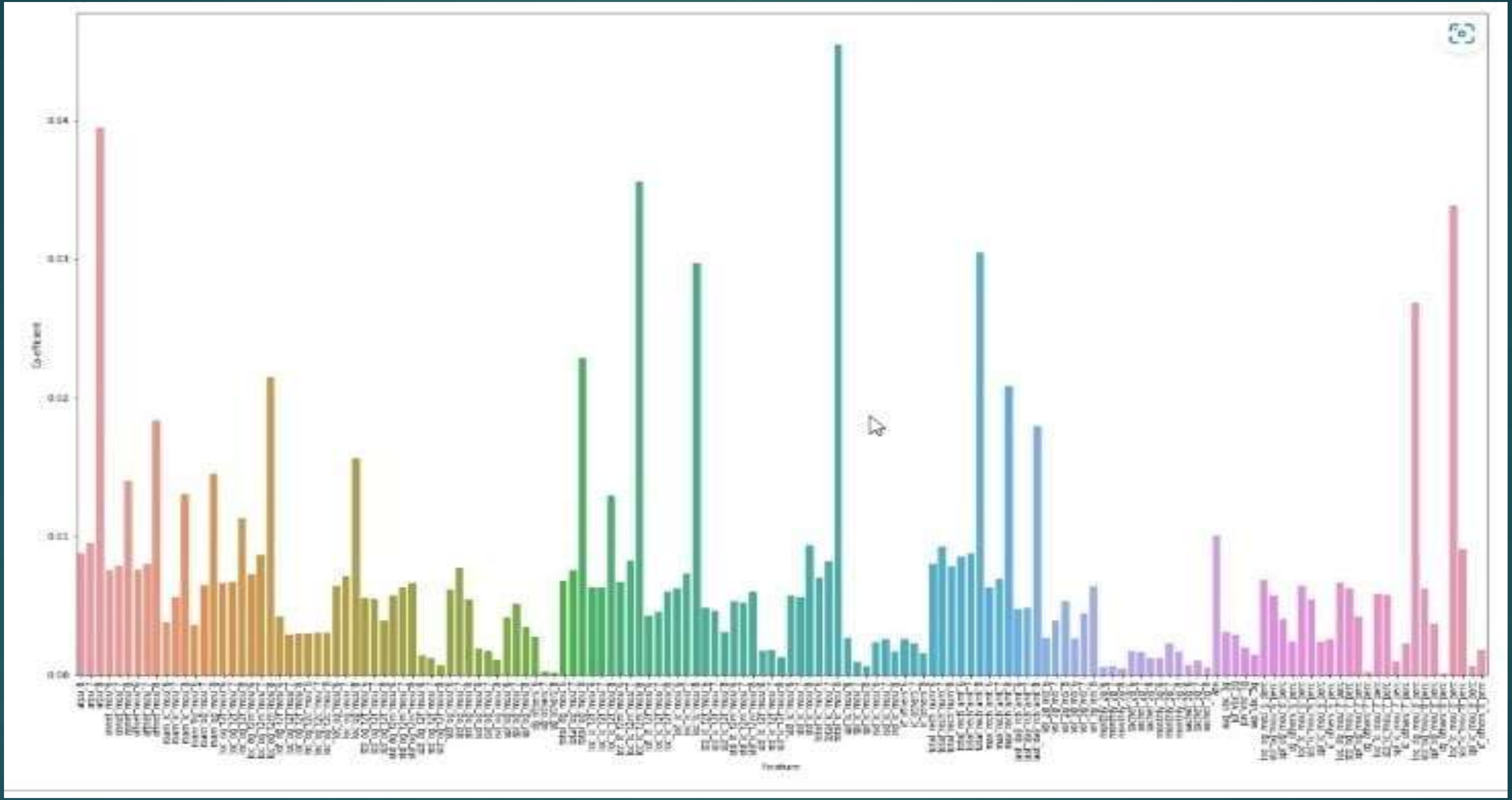
In [921]:
```
feature_importance.sort_values(by='Importance',ascending=False).head(15)
```

Out[921]:

	Feature	Importance
80	total_ic_mou_8	0.045455
2	arpu_8	0.039494
59	loc_ic_t2m_mou_8	0.035565
145	loc_ic_mou_8_perc	0.033872
95	total_rech_amt_8	0.030451
65	loc_ic_mou_8	0.029687
141	loc_og_mou_8_perc	0.026847
53	total_og_mou_8	0.022870
20	loc_og_t2m_mou_8	0.021476
98	max_rech_amt_8	0.020809
8	offnet_mou_8	0.018339
101	last_day_rch_amt_8	0.017916
29	loc_og_mou_8	0.015629
14	roam_og_mou_8	0.014498
5	onnet_mou_8	0.013978

FEATURE IMPORTANCE ANALYSIS

FEATURE IMPORTANT ANALYSIS



FEATURE PLOT FOR NON ZERO COEFFICIENTS

CONCLUSION

- **In terms of modeling effectiveness, observe Random Forest post SMOTE class imbalance treatment, there is a better ROC curve of 0.81 with accuracy as 92%.**

- **Features that majorly influences the Churn are as below - Mainly its covering the Customer behavior on Recharge, Outgoing Calls**
- **arpu_8**
- **loc_ic_t2m_mou_8**
- **total_ic_mou_8**
- **loc_og_mou_8_perc**
- **loc_og_t2m_mou_8**
- **total_rech_amt_8**
- **loc_ic_t2t_mou_8**
- **loc_og_mou_8**
- **max_rech_amt_8**
- **loc_ic_mou_8_perc**
- **last_day_rch_amt_8**
- **total_og_mou_8**
- **loc_og_t2t_mou_8**
- **offnet_mou_8.**

RECOMMENDATIONS

- **There is a need to observe Month on Month on ARPU, Recharge and Call Activities to watch for the decline trend which need to be reversed by :**

    1. **Giving Recharge Discounts.**

    2. **Value added packs for recharges.**

    3. **Hidden concerns on the network quality to see if the network performance hindering the call activities.**