**Churn Prediction in Telecom using ML**

By

**Arya Padmakumar**

**Objective of the Project** : - Develop a churn prediction model in telecom to predict customers who are most likely to churn.

**Importance of the study** :- Retaining a client is far cheaper than acquiring a new client. Hence companies take special care to understand who is likely to churn and retain them with customized packages suiting their needs.

**Data Source for the project** : Customer’s database in the form of a csv file with 7043 rows ( representing each customer data) and 21 columns ( representing the various features pertaining to these customers).

**The Process**

**Basic Processes – Importing the basic and classification packages, model parameters, loading the csv file , renaming and reading the csv file.**

1. Imported Data Manipulation and Wrangling packages Pandas and Numpy.
2. Imported the Visualization packages Matplotlib and Seaborn.
3. Imported the Classification packages like Logistic Regression, Decision Tree, Random Forest , Support Vector and N Nearest Neighbors.

Imported the model parameters or matrix like accuracy\_score, precision\_score, recall\_score, f1\_score, roc\_auc\_score and log\_loss

1. The csv file was imported, renamed and read.

**Analysing and Visualising the data set**

1. Checked the top and bottom 5 rows of the dataset
2. Checked the number of rows and columns of the dataset.
3. Checked the type of data under each column of the dataset
4. Converted datatype under the column ‘TotalCharges” from object to Float64, for data manipulation and rechecked if the change was done.
5. Checked for null values and removed the rows with null values. ( there were 11 null values under the column ‘ TotalCharges). Rechecked if the null values were removed.
6. Checked the description ( count, minimum and maximum values, 25-50-75percentile, standard deviation of the data under each column having int64 or Float64 data ( the columns were ‘SeniorCitizen’, ‘tenure’, ‘MonthlyCharges’, ‘TotalCharges’) and transposed the same data.
7. Created a heatmap to visualize the correlation between each numeric variable.
8. Created distribution plot to visualize skewness of each numeric variable.
9. Created pair plot to visualize the correlation of each feature with one another.
10. Created count plot to visualize the number of unique values under each feature

**Handling Outliers**

1. Checked for outliers for the columns ‘SeniorCitizen’, ‘tenure’, ‘MonthlyCharges’, ‘TotalCharges’ by drawing boxplot. No outliers found.

**Data Processing**

1. The input variables (x) and output variable(y) were separated into 2 different datasets dfx1 and dfy.
2. Dummy Encoding was done to convert all categorical input variables into 0 and 1 . These categorical input variables were ‘gender','Partner','Dependents','PhoneService','MultipleLines','InternetService','OnlineSecurity','OnlineBackup','DeviceProtection','TechSupport','StreamingTV','StreamingMovies','Contract','PaperlessBilling', and 'PaymentMethod'.
3. Label Encoding was done to convert output variable into 0 and 1.

**Data Normalisation**

1. Normalisation of input variables were done using MinMaxScaler and a new dataframe of input variables ‘dfx1\_normalised’ was created.

**Feature Selection**

1. Those input features which impacts the output variable (outcome) the most were selected using Ch2 test and Mutual Info Classifier. These significant features were **'Contract\_Month-to-month', 'Contract\_Two year','PaymentMethod\_Electronic check','OnlineSecurity\_No','TechSupport\_No', 'InternetService\_Fiber optic'** . The rest of the features were dropped.

**Data Split**

1. Data Split was done to split the data first into Training and Combined Data ( Testing Data + Validation Data) in a 80/20 ratio and then the Combined Data was divided in a 50/50 ratio into Testing data and Validation Data

**Schematic Minority Oversampling Technique ( SMOTE)**

1. SMOTE was undertaken to cancel the imbalance in number of output variables by replication of the minority class number to equal that of majority class.
2. First, imblearn package was pip installed
3. SMOTE was imported from imblearn.over\_sampling and fitted to X\_train and Y\_train datasets.

**Under Sampling**

1. Under sampling technique was used to bringdown the number of majority output classes to equal that of the minority class with the least number.
2. ‘under\_sampling’ was imported from imblearn package and RandomUnderSampler was imported from imblearn.under\_sampling
3. RandomUnderSampler was fitted to X\_train and Y\_train data

**Model Building**

1. Logistic Regression / Decision Tree / Random Forest SVC models , K Nearest Neighbors was build and trained on test data and prediction was done for test data.

**Model Evaluation**

1. Evaluation of all the models were done using Confusion Matrix and using accuracy\_score, recall\_score, precision\_score, f1\_score, roc\_auc\_score and log\_loss.
2. All models returned an Accuracy Score of above 70%.
3. Confusion matrix was drawn using Seaborn for each model.
4. For Decision Tree, Criterion taken was ‘Entropy’ and Maximum Depth taken was 3.
5. For Random Forest , n\_estimator was 20 with Criterion as ‘Gini’
6. Roc\_auc curve was plotted.
7. Accuracy was recalculated using the SMOTE and Undersampling techniques.

**SVC – Grid Search – Cross Validation**

1. For SVC, model parameters and grid parameters were obtained.
2. Cross Validation was done by importing GridSearchCV.
3. The grid was fitted into the training dataset.
4. Inspected the best parameters using GridSearchCV in the best\_params\_attribute and best estimator in the best\_estimator\_attribute
5. The predictions were re-run and confusion matrix was re-obtained using Y\_test and grid\_predictions
6. Better values for model Evaluation parameters were re-obtained using Y\_test and grid\_predictions

**Conclusion**

All the ML models built gave good prediction ( >80%) on churn.