Laxmi Prasanna Chennadi

DATA 606

Deliverable-3 Report

**Analysis of Telecom Churn Data**

**Data Preparation:**

Initially after loading the data, I have started data cleaning by checking for missing values and observed no missing values. I proceeded with converting data types of the data for better analysis. Further I moved on to exploratory data analysis by heat map and correlation methodology to understand the correlation between the columns and noticed that there is correlation between churn and total day charge, customer service calls and international plan. Moving forward I tried detecting the outliers in the columns and tried clipping them for better analysis.

Later on, to prepare data for Model building I have created two data frames X with predictor variables and Y with target variable. Here international plan and voice mail plan are categorical types(yes/no), pd.get\_dummies is used to convert them into numerical(1/0). Now data is normalized i.e, all features are set to one range of scale for feature scaling of training data using Min-Max scaling is done, by importing MinMaxScaler from sklearn.preprocessing. Later X and Y data frames are split into training and testing sets using train\_test\_split from sklearn.model\_selection. Finally, now that data is prepared we can proceed with Model Construction.

**Model Building:**

Our goal is to predict customers who are most likely to churn or change telecom networks. As it turns out to be a classification problem, I am using Logistic Regression from regression algorithms, KNearestNeighbors to learn how NearestNeighbours would work on this dataset and Random forest algorithms from decision trees family to perform classification analysis on data to predict churn.

Applying Binomial Logistic regression would give better results and tried fitting the model on the X\_train and Y\_train dataframes and performed prediction on train data where 77% of accuracy is recorded and applying the same classifier on test data, logistic regression gives 78% accuracy.

Secondly, I applied k-NearestNeighbors classifier by iterating k value from 1-10 and plotting the accuracy of each iteration shows 1 as the optimum value of k, therefore by taking n\_neighbors as 1, I tried fitting the model on train data and obtained 100% accuracy but when applied on test data the accuracy rate has dropped by 14% which happens in case of overfitting of data. Overfitting occurs when a model learns the detail and noise in the training data to the extent that it negatively impacts the performance of the model on new data. This means that the noise or random fluctuations in the training data is picked up and learned as concepts by the model. The problem is that these concepts do not apply to new data and negatively impact the models ability to generalize.

Random Forest Classifier is an ensemble method which combines more than one algorithm and. It belongs to the family of decision trees, it is based on the principle of creating a set of decision trees from random subsets of a training set and then aggregates their votes to decide the final class of test object. When applied on train set 95% accuracy is observed and on test set 94% which clearly depicts Random Forest Classifier has given best results so far.

Unique identity columns like state, area code and phone number are just providing information about customers, though these columns do not have any significant impact on the churn column. But, similarities between data points of these columns might impact churn. For example, if customers from single state or area code are changing networks this analysis helps to detect such patterns from data. So to predict such patterns we are creating a new column called cluster by detecting similarities among all the rows and giving them a group id which is a cluster.

K-Means clustering is used to cluster data points with similar properties and will help train models to learn from those similarities. How many clusters to create? To know the number of clusters to create, find WCSS(within cluster sum of squares) using k-means classifier by iterating k value in range of 1-10 and plotting an elbow curve to know where the value of WCSS tends to flatten. By applying this on our dataset gives 5 as the optimum value of k. Why do we choose k as the optimum value where the elbow curve flattens? This method calculates SSE (sum of squared distance) between each data point and mean of the cluster. As we go on increasing the value of k WCSS becomes zero which is not ideal to find the similarities. So we choose the value of k where the curve starts to flatten.

After creating clusters, I have observed that there are columns which have multicollinearity, applying dimensionality reduction on these features reduces the computational complexity by reducing the number of samples created by model and interpreting parameters of the model. I chose the PCA(Principal Component Analysis) algorithm and created two new components. After creating these 2 new components, a new data frame is created by reloading the dataset to append two pca components to churn, international plan, voicemail plan and clusters.

As a final step, after altering the dataset I have applied the RandomForest classifier with an expectation of obtaining better results. On fitting on train data accuracy of 100% has been recorded whereas, on test data prediction accuracy has dropped to 88-89%. This was supposed to be the best model but an overfitting problem is observed.

**Deliverable 4:**

As part of further analysis of my project I would like to work on reducing overfitting problems by using Cross validation and Hyperparameter tuning of algorithms using Gridsearch techniques.

**Model Results:**

**Predictions on Test sets:**

f1-score of **LogisticRegression Model**

precision recall f1-score support

0 0.96 0.78 0.86 857

1 0.38 0.80 0.51 143

accuracy  **0.78**  1000

macro avg 0.67 0.79 0.69 1000

weighted avg 0.88 0.78 0.81 1000

f1-score of **KNN Model**

precision recall f1-score support

0 0.91 0.93 0.92 857

1 0.51 0.47 0.49 143

accuracy  **0.86**  1000

macro avg 0.71 0.70 0.70 1000

weighted avg 0.86 0.86 0.86 1000

f1-score of **RandomForest Model**

precision recall f1-score support

0 0.97 0.96 0.97 857

1 0.78 0.83 0.80 143

accuracy **0.94** 1000

macro avg 0.87 0.89 0.88 1000

weighted avg 0.94 0.94 0.94 1000

**Clustering+ PCA+Random Forest Classifier:**

**Train set:** precision recall f1-score support

False 1.00 1.00 1.00 1993

True 1.00 1.00 1.00 340

micro avg 1.00 1.00  **1.00** 2333

macro avg 1.00 1.00 1.00 2333

weighted avg 1.00 1.00 1.00 2333

Accuracy: 100%

**Test set:** precision recall f1-score support

False 0.90 0.97 0.93 857

True 0.68 0.36 0.47 143

micro avg 0.88 0.88  **0.88** 1000

macro avg 0.79 0.66 0.70 1000

weighted avg 0.87 0.88 0.87 1000

Accuracy: 88%