# Telecom Users Dataset

A REPORT ON COMPARITIVE STUDIES

Deepak Tripathi (dt20483@essex.ac.uk)

**Business Objective:**

1st: Predict behavior to retain telco customers. Analyze all relevant customer data and develop focused customer retention programs.

2nd: Identify which all attributes are important in deciding if a customer will leave the service or not

INTRODUCTION

I will develop a classification model which can predict if a customer can leave the service in the future or not. To verify the performance of the model I will use following metric-

Classification Metrics

- Accuracy.
- Precision
- Recall
- F1-Score
- Confusion Matrix

Here, only accuracy is not a good matric to verify the performance of the model as data is highly imbalanced.

For our case, I think Precision is important, which says among total predicted for a category what percentage were correct predictions. This is due to that fact that if our model predicts someone will not leave the service and later if that persons leaves then it will be a loss for the company.

Recall also seems a good metric which says of total data available for a certain category, what percentage has returned correctly. This is because if there are many people who will leave the service, but our model is not able to identify them, it will again be a big loss for the company.

Since we want precision and recall both be high, I will check F1-core as well, which is the harmonic mean of both.

I will also use confusion matrix which can give detailed analysis of how model is performing by drawing a comparison between actual and predicted values in form of a matrix

Summary on what all I did:

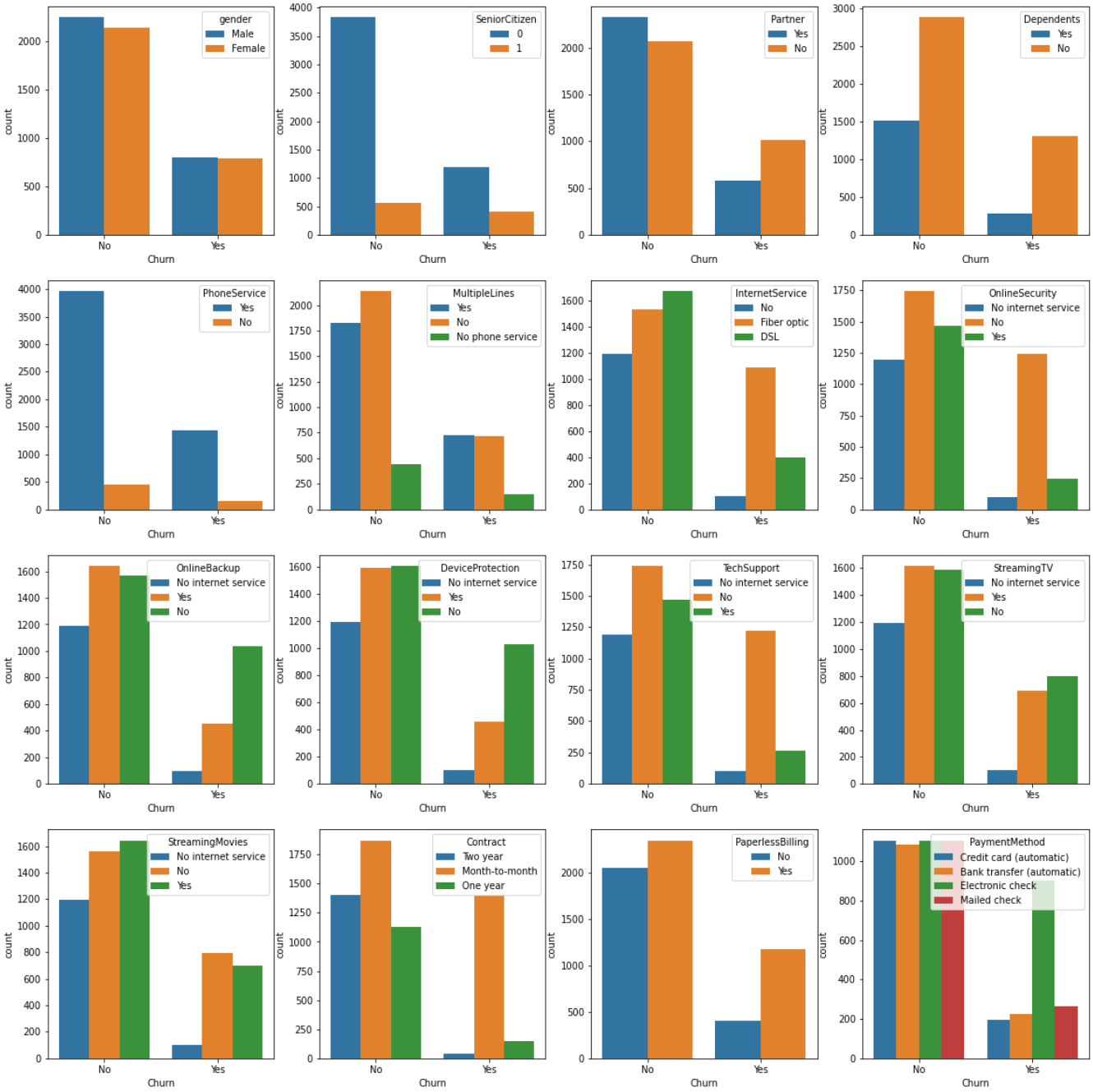
- Upon loading the data, I performed few initial checks and found that data was highly imbalanced.
- It contained 5986 rows and 22 columns.
- It did not contain any null values.
- There were 3 numeric and 19 categorical columns.
- Used chi square test to check which all categorical columns are impacting categorical target ‘’Churn”.
- Used Annova test to check which all numerical columns are impacting categorical target ‘’Churn”.
- Used correlation and VIF to verify if there is any collinearity in data
- Tried standard scaling on numeric features and one hot encoding with categorical columns.
- Did a comparative study among multiple models. first with all columns and then with only selected columns (selected after chi square and annova test).
- Did hyperparameter tunning to get the best model possible.
- Tried deep learning model MLP on this data to check how does it perform.
- Eventually used Logistic regression and Gradient boosting to predict the test data and validate its performance.
- Used to embedded technique to get important features. I used Gradient boosting to get the important features.

# Exploratory Data Analysis

**EDA on categorical predictor**

First, I have tried analysing what all categorical columns are contributing to the target variable 'churn'. Please observe below figure and you will notice following observations-

1. In the very first figure which is gender against churn, gender does not seem to have much impact on target churn. Distribution of male, female for each target type is same. When churn is No, male female both increases in count whereas when is churn is Yes, male, and female both decreases in count. So, gender (male, female) does not seem to be a strong predictor.
2. Similar pattern we can observe in 1st figure of 2nd row, which is PhoneService against target churn. Distribution of male, female for each target is same.
3. Rest categorical columns follow different pattern against target churn for its different values ('Yes', 'No'). So those all seem to be good predictor for the task at hand.




Let us check statistically which all categorical predictors are contributing to target. We will use Chi Square Test to verify on what categorical columns target churn depends. In Chi Square Test below are the hypothesis-

Null Hypothesis(H0) = Columns are independent

Alternative hypothesis (H1) = Columns are dependent.

We will calculate P-Value and if that P-Value is more than .05 then we will accept Null Hypothesis(H0) otherwise we will select Alternative Hypothesis (H1).

```
P-Value for  gender  =  0.4600692428
P-Value for  SeniorCitizen  =  0.0000000000
P-Value for  Partner  =  0.0000000000
P-Value for  Dependents  =  0.0000000000
P-Value for  PhoneService  =  0.4660500727
P-Value for  MultipleLines  =  0.0191440653
P-Value for  InternetService  =  0.0000000000
P-Value for  OnlineSecurity  =  0.0000000000
P-Value for  OnlineBackup  =  0.0000000000
P-Value for  DeviceProtection  =  0.0000000000
P-Value for  TechSupport  =  0.0000000000
P-Value for  StreamingTV  =  0.0000000000
P-Value for  StreamingMovies  =  0.0000000000
P-Value for  Contract  =  0.0000000000
P-Value for  PaperlessBilling  =  0.0000000000
P-Value for  PaymentMethod  =  0.0000000000
```

P-Value for “gender” and “PhoneService” are significantly large which proves that these does not have any impact on target. Same analysis we got with descriptive analysis as well.

**EDA on numerical predictor**

Now let’s see how numerical columns are impacting the target ‘churn’.

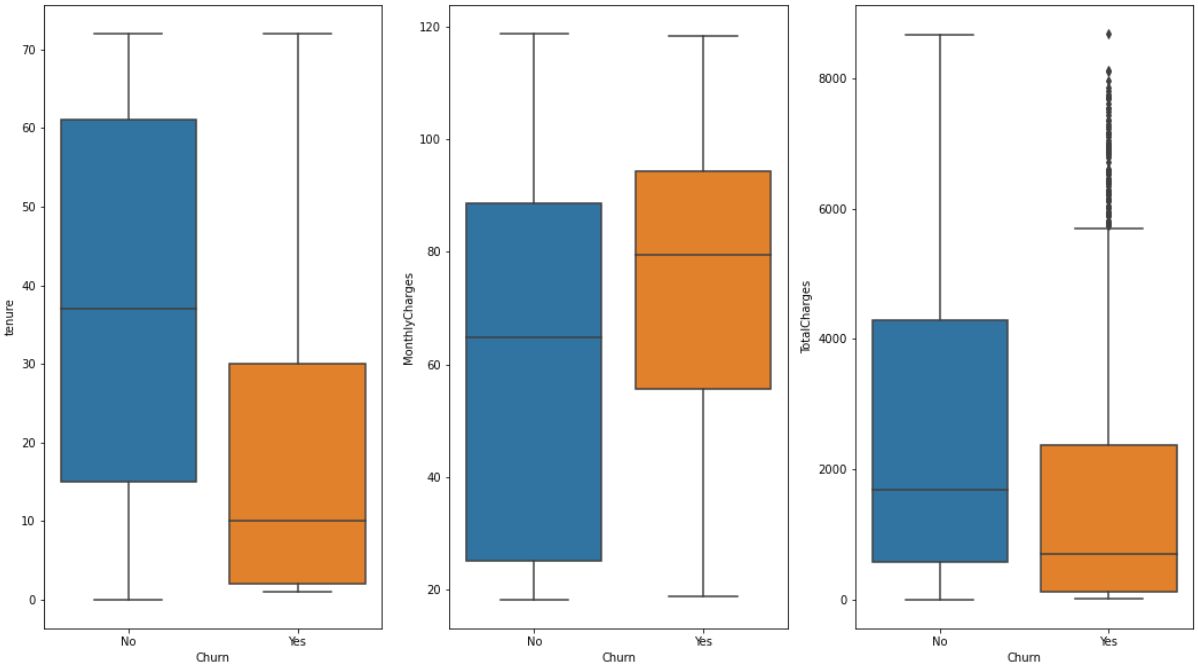
Following are observations of the below figure-

Customers churn when tenure is low (1st figure).

Customers churn when monthly charges are higher.

Customers churn when total charges are low.

For each numerical column, average value changes for different values of target, so all of them seem to be good predictors for this task.




Let us check statistically which all numerical predictors are contributing to the target. We will use 1-way Annova and F-Test to verify, what numerical columns target churn depends on. In Annova Test below are the hypothesis-

Null Hypothesis(H0) = There is no difference in mean for different values of target

Alternative hypothesis (H1) = There is some difference in mean for different values of target.

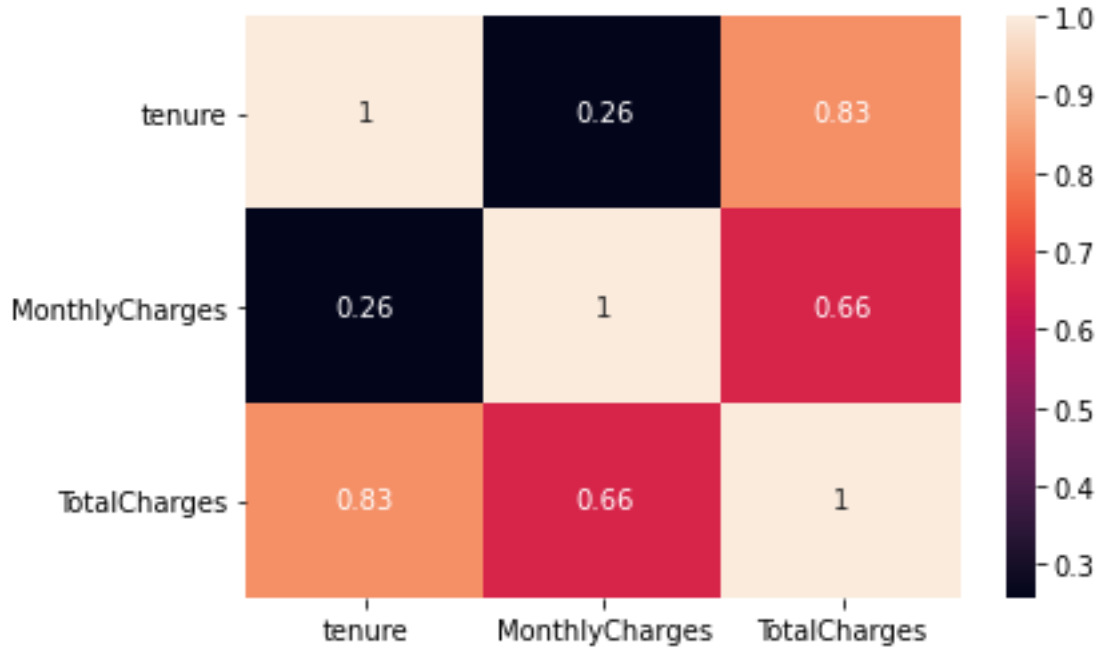
```
P-Value for ' tenure ' =  1.774696701316516e-170
P-Value for ' MonthlyCharges ' =  4.0904880183731545e-48
P-Value for ' TotalCharges ' =  4.0713147719280728e-53
```

All P-Values are significantly small so we will reject Null Hypothesis and Accept Alternative Hypothesis.

Note that, same result we had got with the help of boxplot as well.

## Collinearity Check

Now let's check if we have collinearity in our data. For this, we will first draw correlation among numerical predictors. Below is the result-




In above figure TotalCharges seems to be highly correlated with tenure. To be certain if we have collinearity, we will check variance inflation factor(VIF) score as well. Note normally if a column has more than 5 VIF score, then it is considered to be collinear with some other column.

```
vif = pd.Series([variance_inflation_factor(num_df.values, i) for i in range(num_df.shape[1])],index=num_df.columns)
print(vif)
```

```
tenure            5.871210
MonthlyCharges    3.250388
TotalCharges      9.642726
const            14.948593
dtype: float64
```

In above result we can see that tenure and TotalCharges have VIF greater than 5 so they might be collinear.

Note that same result we found with correlation matrix as well.

## Model Training

Now let us train a model first without removing any predictor and see what result we are getting. We will use multiple models and perform a comparative study to select the best model.

Before fitting the model, first we will divide the whole data in train and test part in 70:30 ratio, respectively. We will apply 5-fold cross validation on train part and eventually model with best cross validation score will be used to predict the test data and validating its performance on test data.

Below is the result obtained after fitting models on complete train data without removing any columns.

| | Accuracy | F1-Score | Precision | Recall | Parameter |
|---|---|---|---|---|---|
| DecisionTreeClassifier | 0.724883 | 0.723859 | 0.722888 | 0.724883 | Default |
| LogisticRegression | 0.803074 | 0.796255 | 0.793968 | 0.803074 | Default |
| RandomForestClassifier | 0.789040 | 0.778740 | 0.776448 | 0.789040 | Default |
| GradientBoostingClassifier | 0.799287 | 0.790536 | 0.788550 | 0.799287 | Default |

As you can see logistic regression has performed the best among all.

Now let's try removing "gender" and "PhoneService" as they do not seem to have any impact on target. Below is the result after removing "gender" and "PhoneService".

| | Accuracy | F1-Score | Precision | Recall | Parameter |
|---|---|---|---|---|---|
| DecisionTreeClassifier | 0.724883 | 0.723391 | 0.722009 | 0.724883 | Default |
| LogisticRegression | 0.803297 | 0.797095 | 0.794722 | 0.803297 | Default |
| RandomForestClassifier | 0.783025 | 0.772120 | 0.769482 | 0.783025 | Default |
| GradientBoostingClassifier | 0.797728 | 0.789028 | 0.786903 | 0.797728 | Default |

As we suspected, performance has not changed much. Rather performance of logistic regression has now slightly increased.

## Hyper tuning models

Now let's try hyperparameter tunning to get the best possible score. We will use GridSearchCV to for this purpose.

best_score_ and best_params_ after grid search on decision tree

```
0.7865883059840646
{'decisiontreeclassifier__criterion': 'gini', 'decisiontreeclassifier__max_depth': 4}
```

---

best_score_ and best_params_ after grid search on logistic regression

```
0.808417814392444
{'logisticregression__C': 0.03162277660168379}
```

best_score_ and best_params_ after grid search on random forest

```
0.799730107534891
{'randomforestclassifier__max_features': 'log2', 'randomforestclassifier__min_samples_split': 8, 'randomforestclassifier__n_est
imators': 100}
```

best_score_ and best_params_ after grid search on gradient boosting

```
0.8055214982880325
{'gradientboostingclassifier__learning_rate': 0.01, 'gradientboostingclassifier__max_depth': 4, 'gradientboostingclassifier__ma
x_features': 'log2', 'gradientboostingclassifier__n_estimators': 500}
```

Again, we have got logistic regression to be performing the best. Gradient Boosting too some what closure to logistic regression in performance.

## Try deep learning model MLP

Now let's try using deep learning model (MLP) and check how does it perform on this data

```
Epoch 58/500000
1/1 [==============================] - 0s 42ms/step - loss: 0.5658 - accuracy: 0.7433 - val_loss: 0.5628 - val_accuracy: 0.75
95
Epoch 59/500000
1/1 [==============================] - 0s 54ms/step - loss: 0.5620 - accuracy: 0.7574 - val_loss: 0.5558 - val_accuracy: 0.76
39
```

Above is the best score that I could get with MLP after tunning its various parameter such activation function, No. of neurons and Layers.

## Selection of model by comparing scores on test data

Eventually, the models that have performed the best so far are logistic regression and gradient boosting. Lets check what score they are giving on test data. Below is the result.

Logistic Regression

```
model1 = make_pipeline(ct, LogisticRegression(C = 0.03162277660168379))

model1.fit(X_train, y_train)

model1.score(X_test,y_test)
```

```
0.7942551770207081
```

Gradient Boosting

```
model3 = make_pipeline(ct_selected, GradientBoostingClassifier(learning_rate = 0.01,max_depth = 4, max_features = 'log2' , n_esti

model3.fit(X_train, y_train)

model3.score(X_test,y_test)
```

```
0.8036072144288577
```

**Conclusion:**

According to my analysis gradient boosting is the best performing model. One advantage of this is that getting important attributes is very handy. Below are the important attributes as per this model.

Most important is at 1st position and least important is at last position.

```
index=np.array(clf.feature_importances_).argsort()[::-1]
X_train.columns[[index]]
```

```
Index(['PaymentMethod', 'gender', 'Partner', 'DeviceProtection',
       'SeniorCitizen', 'StreamingMovies', 'OnlineBackup', 'TechSupport',
       'TotalCharges', 'StreamingTV', 'MonthlyCharges', 'PaperlessBilling',
       'OnlineSecurity', 'Contract', 'tenure', 'PhoneService', 'MultipleLines',
       'InternetService', 'Dependents'],
      dtype='object')
```