Telco Customer Churn Analysis

Abhishek Khatti

Western Governors University

# Contents

[Contents 2](#_Toc47274159)

[Synopsis 5](#_Toc47274160)

[The Telco Customer Churn Raw Data Set 5](#_Toc47274161)

[Tool Selection & Data Analysis Objectives 6](#_Toc47274162)

[Data Analysis Objectives 6](#_Toc47274163)

[Tool Selection 6](#_Toc47274164)

[Data Analysis Methods 8](#_Toc47274165)

[Target Variable 8](#_Toc47274166)

[Describing Existing Data 9](#_Toc47274167)

[Finding Phenomenon and Predicting Churn 9](#_Toc47274168)

[Project Assumptions & Structure 10](#_Toc47274169)

[Data Preparation & Exploration 10](#_Toc47274170)

[Goals of Data Cleaning and Preparation 11](#_Toc47274171)

[Loading Required Libraries 11](#_Toc47274172)

[Data Collection & Feature Analysis 13](#_Toc47274173)

[Input/Predictor Variables 19](#_Toc47274174)

[Predicted/Response Variable 20](#_Toc47274175)

[Data Preparation & Feature Engineering 20](#_Toc47274176)

[Reducing Feature Levels 20](#_Toc47274177)

[Rename Feature Levels for Readability 21](#_Toc47274178)

[Converting numeric to ordinal categorical feature 22](#_Toc47274179)

[Delete features 22](#_Toc47274180)

[Choice of Visualizations 24](#_Toc47274181)

[Addressing Anomalies & Missing Values 25](#_Toc47274182)

[Data Exploration & Summary Descriptions 26](#_Toc47274183)

[Summary Statistics for Target Variables 26](#_Toc47274184)

[Summary Statistics for Binary & Categorical Predictor Variables 28](#_Toc47274185)

[Binary Predictor Variables Summary 29](#_Toc47274186)

[Categorical Predictor Variables Summary 36](#_Toc47274187)

[Summary Statistics for Continuous Predictor Variables 39](#_Toc47274188)

[MonthlyCharges 41](#_Toc47274189)

[TotalCharges 42](#_Toc47274190)

[Predictor Variables Bivariate Analysis 42](#_Toc47274191)

[Categorical Predictors’ Proportion distribution by Churn 45](#_Toc47274192)

[gender by Churn 46](#_Toc47274193)

[SeniorCitizen by Churn 47](#_Toc47274194)

[Partner by Churn 47](#_Toc47274195)

[Dependents by Churn 48](#_Toc47274196)

[PhoneService and Multiple Lines by Churn 49](#_Toc47274197)

[InternetService by Churn 50](#_Toc47274198)

[OnlineSecurity and OnlineBackup by Churn 50](#_Toc47274199)

[DeviceProtection and TechSupport by Churn 52](#_Toc47274200)

[StreamingTV and StreamingMovies by Churn 53](#_Toc47274201)

[Contract by Churn 54](#_Toc47274202)

[PaperlessBilling by Churn 54](#_Toc47274203)

[PaymentMethod by Churn 55](#_Toc47274204)

[tenuregroup by Churn 56](#_Toc47274205)

[Numerical Predictors’ distribution by Churn 57](#_Toc47274206)

[Correlations between continuous variables by Churn 58](#_Toc47274207)

[Evaluative models for Predictions 60](#_Toc47274208)

[Data Pre-Processing 61](#_Toc47274209)

[Transformation of both Predictor and Response Variables 62](#_Toc47274210)

[Check Correlations between Transformed Variables 66](#_Toc47274211)

[Train-Test Split of an initial training data set 68](#_Toc47274212)

[Define Custom Functions 68](#_Toc47274213)

[Plot Confusion Matrix 69](#_Toc47274214)

[Plot ROC-AUC curve 69](#_Toc47274215)

[Plot Precision-Recall Curve 70](#_Toc47274216)

[Model Scores 70](#_Toc47274217)

[Create Predictive Models 71](#_Toc47274218)

[Dummy Baseline Model 71](#_Toc47274219)

[Synthetic Minority Oversampling Technique 73](#_Toc47274220)

[Dimension Reduction with Recursive Feature Elimination 74](#_Toc47274221)

[Stats Model for Discriminant Analysis and Finding Phenomenon 76](#_Toc47274222)

[Interpreting the parameters 77](#_Toc47274223)

[Remove additional features 78](#_Toc47274224)

[Validate Stats model 79](#_Toc47274225)

[Check the Logistic Regression Residuals Assumptions 79](#_Toc47274226)

[Logistic Regression Model for Prediction – with Default Parameters 81](#_Toc47274227)

[Hyperparameter Optimization of Logistic Regression Model for Prediction 85](#_Toc47274228)

[Random Forest Model for Prediction – with Default Parameters 88](#_Toc47274229)

[Hyperparameter Optimization for Random Forest Model for Prediction 92](#_Toc47274230)

[Visualizing Sample Decision Tree from Default Random Forest 96](#_Toc47274231)

[Scores Comparison for Predictive Models 96](#_Toc47274232)

[Top Features of Interest 98](#_Toc47274233)

[Final Output Files 99](#_Toc47274234)

[Summary and Conclusions 100](#_Toc47274235)

[Acknowledgment 103](#_Toc47274236)

[References 104](#_Toc47274237)

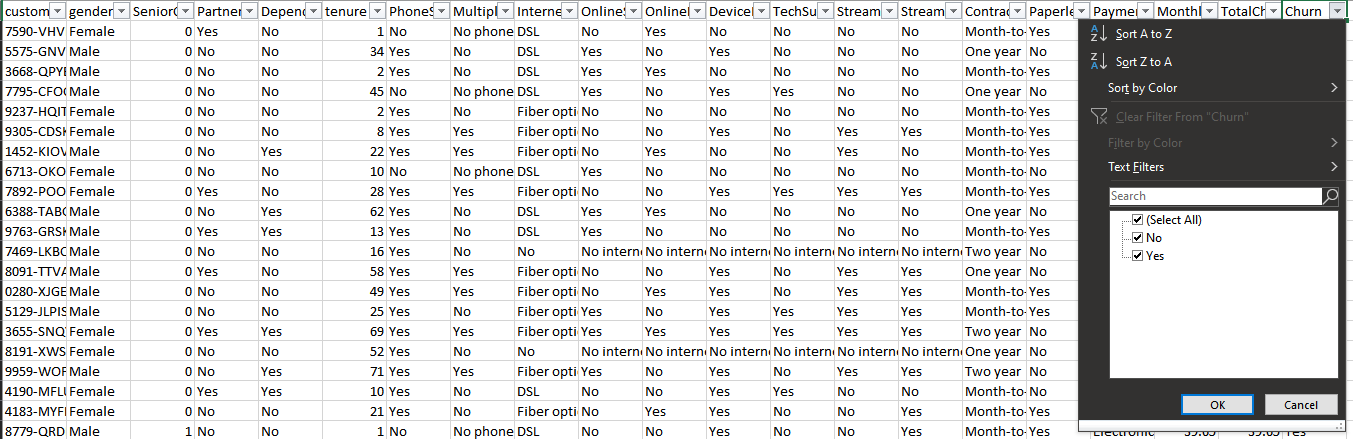
Telco Customer Churn Analysis Report

# Synopsis

A telecommunications company, Telco is concerned about customers leaving their business for competitors. It needs to know which customers are leaving and where to focus their attempts to retain the customers. An analysis of customer data is performed to identify why customers are leaving and potential indicators are identified to explain why those customers are leaving, so the company can make an informed plan to mitigate further loss. Among the different tools available to perform this analysis, a choice between R, SAS, and Python is to be performed. The data is to be cleaned and if required, transformed to complete the analysis & model building process. Detailed analysis of features of the dataset will be performed by applying descriptive and predictive methods. The company is focused on finding which set of customers are leaving, meaning, what kind of customers are leaving the company for competition. For example, is the group of customers with high monthly charges more likely to churn or customers with a month-to-month contract more likely to churn. Also, the company wants to identify indicators of why the customers are leaving to build mitigation strategies for customer loss. Finally, a summary of findings is provided to conclude the analysis ("Performance Assessment - ACE3").

# The Telco Customer Churn Raw Data Set

The Telco Customer Churn raw data set is available in a Comma Separated Value format (.CSV) file, “*WA\_Fn-UseC\_-Telco-Customer-Churn.csv*” ("Performance Assessment - ACE3"). The file consists of a total of 7044 rows, with 1 header row included and 21 columns. The raw data contains information about both, customers who are still with Telco and customers who left for the competition. Each row provides details about a customer and the “Churn” column informs whether a customer left or is still an active customer with Telco. A quick check on the Churn column shows that there are only 2 values available for this column, Yes or No. In other words, Churn is a binary variable. Further analysis of all the variables will have to be performed.



# Tool Selection & Data Analysis Objectives

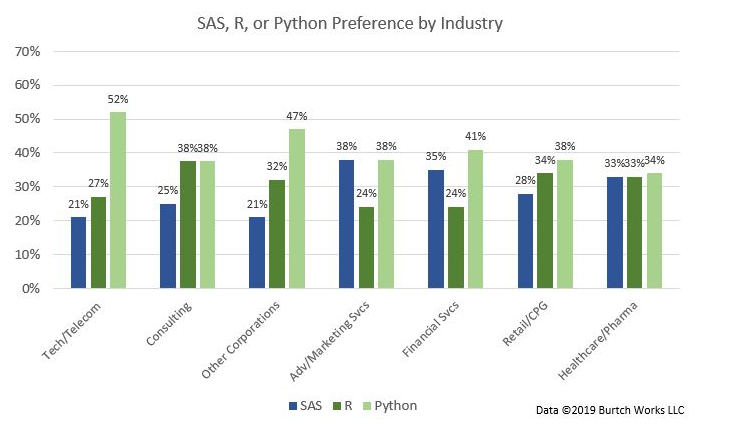
## Data Analysis Objectives

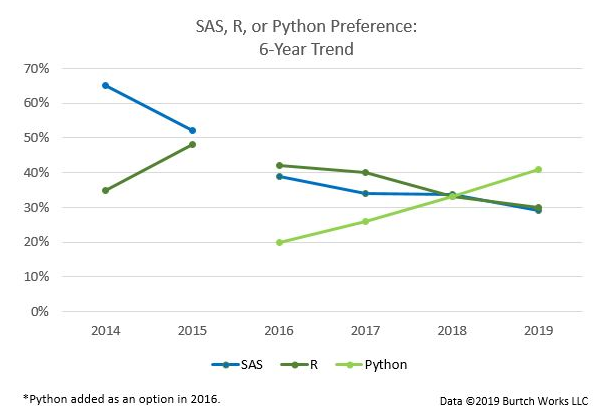
Before any of the analysis could be started, establishing meaningful objectives are utmost important for meaningful results. There are a few major objectives of this analysis, which are listed below:

1. Summarize current data in a manner that would provide additional insights about customer churn and retention.
2. Finding the variables that are associated with a higher churn rate
3. Provide proper foundational analysis of data set, so the organization could potentially predict future customer churn and help fine-tune their customer retention strategies.

## Tool Selection

Although there are numerous tools available in the market that would aid in perform meaningful data analysis, Telco has a choice between SAS, R, or Python. Based on recent research and feedback from customers from various industry and domains, Python is the first choice among various industry leaders for its versatility, a massive collection of open-sourced libraries and easily scalable solutions (Burtch Works LLC, 2019 SAS, R, or Python Survey Update: Which Tool do Data Scientists & Analytics Pros Prefer? 2019). From seven distinct industry domains, Python ties closely with SAS and R, only in Marketing Services, Consulting Services, and Healthcare domains. Python is the current choice of implementation in the remaining four sectors.





Although SAS and R provide competitive advantages, like robust customer support and unmatched predictive analytics capabilities for SAS, and great flexibility and ease for statistical analysis for R, Python provides equally challenging options. Specifically, this involves using pandas for data wrangling, manipulation, and cleaning, matplotlib and Seaborn for data visualization, and scikit-learn itself for model creation and analysis.

In addition to the factors above, Python provides an opportunity to use an HTML based programming editor called “Jupyter Notebook”. It provides a stepwise, incremental solution building capability, where analyzing data sets can be done more methodically and various sophisticated visualizations can be performed within the same HTML page. Also, it makes taking a holistic view of your current analysis much easier by letting you keep track of analysis already performed up until any given point. For example, within Jupyter Notebook, the code can be written in “cells” and each cell can be run independently of others, yet in the working session, it remembers all the imported libraries and results generated as a part of a successful run of programming code from previous cells. Also, as you keep building the analysis solution, you can easily scroll back and forth to check your earlier visualizations. On top of these features, the visualizations are very presentable and could easily be part of formal management style reports.

Given these choices, Python was chosen as a tool of choice for performing the Telco customer churn analysis.

## Data Analysis Methods

### Target Variable

From a quick check of raw data, it can be observed that the *Churn* variable is binary. Since the essential criteria and phenomenon for prediction are the kind of customers “churning” from Telco’s customer base, which is indicated by the *Churn* column, this would be the target variable in our analysis. This narrows the choices given the nature of the target variable.

### Describing Existing Data

For summarizing or describing the existing data, “Measures of Central Tendencies” and “Measures of Spread” could provide meaningful insights into the data. Also, a “Correlation Matrix” could provide additional information about multicollinearity among various features and help select only discriminating variables in the analysis. The correlation matrix could be created on cleansed data as well as transformed data to get a better understanding of dependencies among features. In addition to the correlation matrix, the Recursive Feature Elimination (RFE) algorithm is also used to reduce the overall number of features by listing out the most important features. Principal Component Analysis (PCA) and Factor Analysis of Mixed Data (FAMD) are by no means bad methods, but these algorithms do not list out which individual features are important. These return only principal components which are a combination of features. “Frequency Distribution” for features by the Churn variable would also provide details on what kind of customers are currently churning the most.

### Finding Phenomenon and Predicting Churn

Since the potential target variable is a binary, the predictive methods, Binary Logistic Regression, and Random Forest algorithms could be used to predict the customer churn (Tuffery, *Table 6.2 Predictive Methods* 2011). The assumptions for these two algorithms could be checked in further sections. As a best practice, at least 2 different types of predictive algorithms are implemented, and the best performing algorithm and model is chosen for deployment. Following this practice, both Logistic Regression and Random Forest will be implemented, and the best performing model will be recommended.

# Project Assumptions & Structure

The project structure follows the standard data science template, which consists of folder structures specific to a phase of the project. For example, the “data” folder will house all data set related files. The “data” folder will have sub-folders depicting the type of data file it would consist of. For instance, “raw” folder will contain all raw data set files, “processed” folder would consist all files which are produced as an outcome of processing on raw data set files, “external” folder would consist final output files that can be released as an outcome of the analysis and so forth.

Some of the important folders that are used in this analysis are:

1. data/raw: Contains raw customer churn data for Telco
2. data/external: Contains final output files because of the analysis and predictive modeling.
3. notebooks: Contains Python Jupyter notebook files used to perform analysis and prediction. The name of the Jupyter Notebook file is “*wgu\_telco\_customer\_churn\_analysis.ipynb*”

Some important assumptions considered for this analysis & prediction are:

1. The analysis & predictions will be performed in Python version 3, using Jupyter Notebook.
2. All the Python libraries used in the analysis are upgraded to the latest versions before the analysis.
3. Since Jupyter Notebook follows a stepwise analysis approach, the successful run of output will be saved in a separate file.

# Data Preparation & Exploration

## Goals of Data Cleaning and Preparation

Data Cleaning and Preparation aims to identify errors, missing values, and duplicate entries to create a reliable data set. It improves the quality of analysis and training of predictive models for accurate predictions and decision making.

In this case, a check for missing values, incorrect data types, possible reduction of categorical feature levels, and avenues for making the feature level labels more readable will be explored. In case these errors are encountered, the methods used to address these problems will depend on the data within the data set. For example, depending on the number of missing values within a data set, a choice could be made to either drop the observations with the missing values, or logically deduce the best way to impute the data for missing entries. Similarly, if the number of feature levels is more than 2, options can be explored if these levels can be semantically combined to reduce the number of levels.

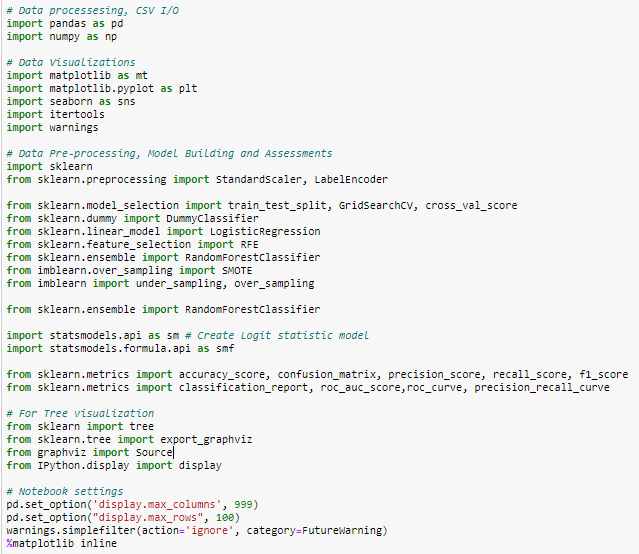
In other words, a cleansed and properly prepared data guarantees that there are no inherent issues with the data itself and brings more confidence in the final results of the data analysis.

## Loading Required Libraries

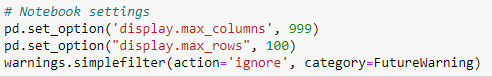
One of the first requirements of any programming language is to import the required libraries and functions into the program to ease the development process. The list of libraries & their intended use in the Python program are:

1. pandas – for file I/O, work with Dataframes, etc.
2. numpy – work with numeric arrays and functions
3. matplotlib – for visualizations
4. seaborn – for visualizations
5. itertools – labeling some confusion matrix plots
6. warnings – suppress future warning messages
7. sklearn – Sci-kit library build machine learning models and perform predictive functionalities
8. imblearn – to address the imbalanced data set
9. statsmodels – create Logit statistic model
10. graphviz – generate Decision tree visualization
11. IPython – Save the Decision Tree visualization

An additional function, “%matplotlib inline” is used to enable HTML inline visualization of plots and graphs. Specific functions from the above libraries are also loaded before starting the analysis.



Also, before starting the development process, some notebook level settings are set up for ease of use.



## Data Collection & Feature Analysis

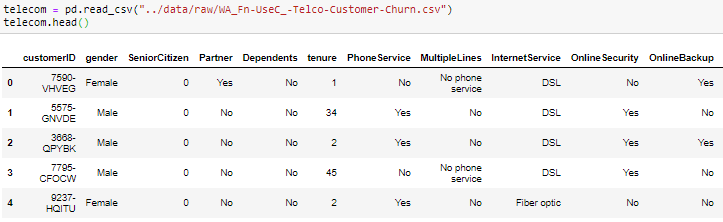
The location of the notebook within the Project folder is:

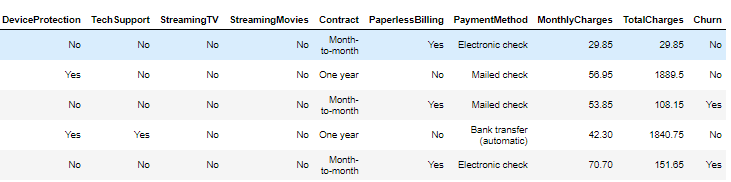
Customer\_Churn/notebooks/ *wgu\_telco\_customer\_churn\_analysis.ipynb*

The location of the raw data file within the Project folder is:

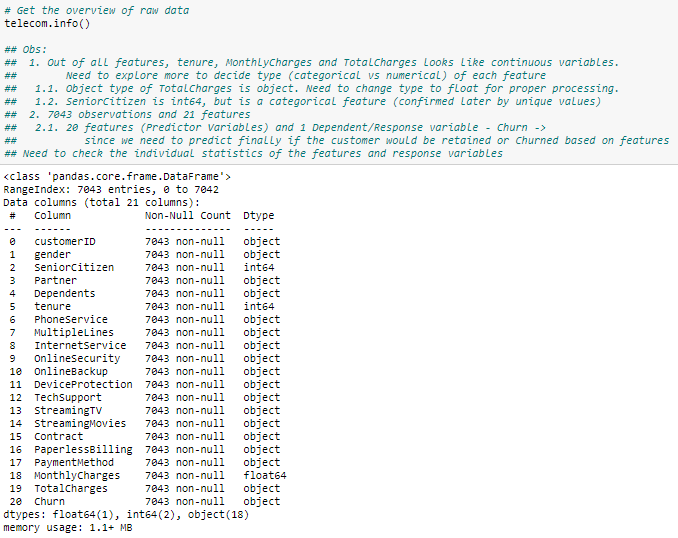
Customer\_Churn/data/raw/*WA\_Fn-UseC\_-Telco-Customer-Churn.csv*

The data from the CSV (Comma Separated Values) type file can be loaded directly into the data frame by a simple *read\_csv()* function from the *pandas* library. Once the data is loaded into the data frame, the top 5 rows can be checked by the *head()* function from the *pandas* library. The data is loaded into the “*telecom*” data frame.





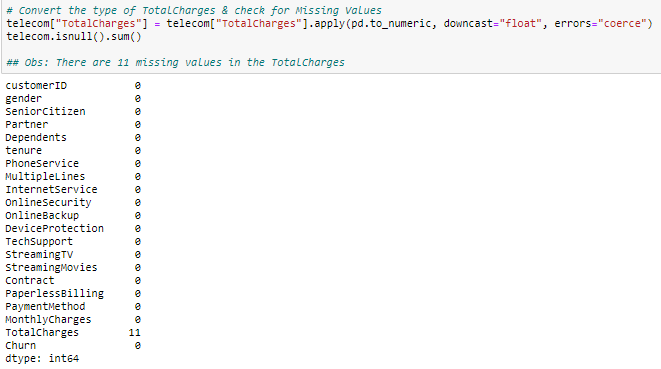
To get an overview of information about the data types of each of the variables, *info()* function can be used.



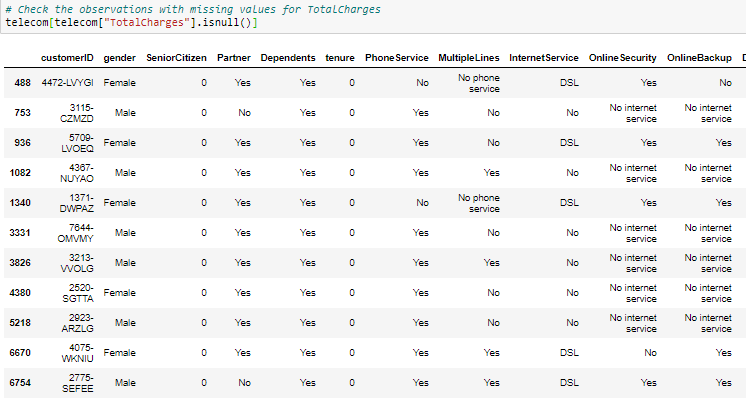
From a brief overview of sampled top 5 rows and variable data types, certain meaningful observations can be made about the data. There are 7043 rows and 21 features in the *telecom* data frame, which ties to the raw data file. Also, it can be inferred that *SeniorCitizen* and *TotalCharges* variables have incorrect data types assigned to them in the *telecom* data frame.

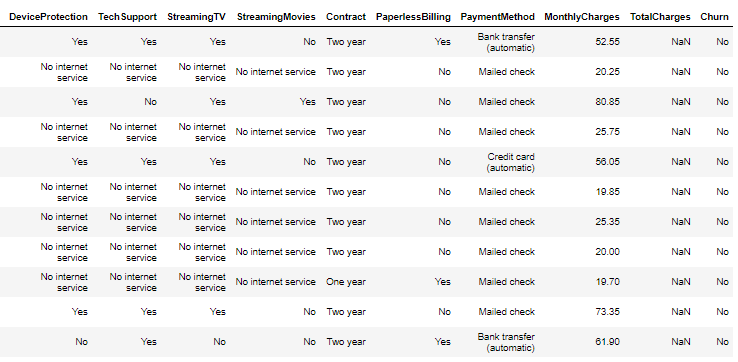
The data types for these two variables will have to be corrected for proper processing of the analysis and avoid any potential miscalculations by Python functions. For example, after checking the raw data set, it can be observed that *SeniorCitizen* has only 2 distinct entries, 0 and 1, and looks like a categorical variable. If the data type for this variable is left as an integer, the Python functions would treat it as a numerical variable, which could lead to false conclusions. Thus, it makes sense to convert it to string type. Similarly, on the other hand, *TotalCharges* will have to be changed from string type to float data type.

In addition to changing the data types, a check for missing values will also have to be performed. The following piece of code will convert the *TotalCharges* to float and check for missing values in the *telecom* data frame.



It can be noted that there are 11 missing values in *TotalCharges*. It can either be imputed by a variety of methods, e.g. median, mean, etc. or be replaced by 0. A check for records with missing *TotalCharges* values in *telecom* gives more insight into other details.

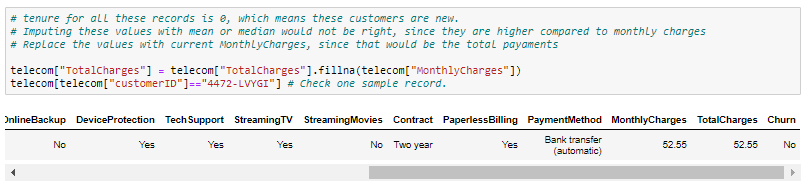




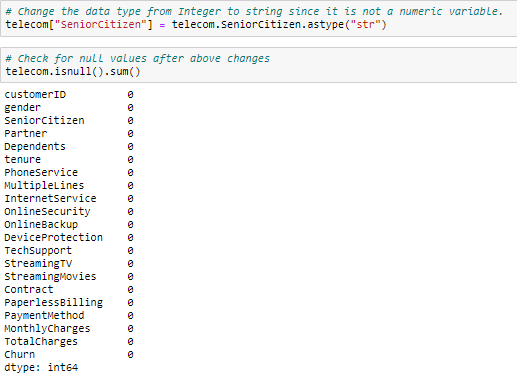


It can be noted that *tenure* for all these records is 0, which means these customers are new. Thus, the *TotalCharges* would not have accrued enough to be imputed as a mean value of 2283.30 or a median of 1397.47. A more realistic approach would be to copy the *MonthlyCharges* for these customers.

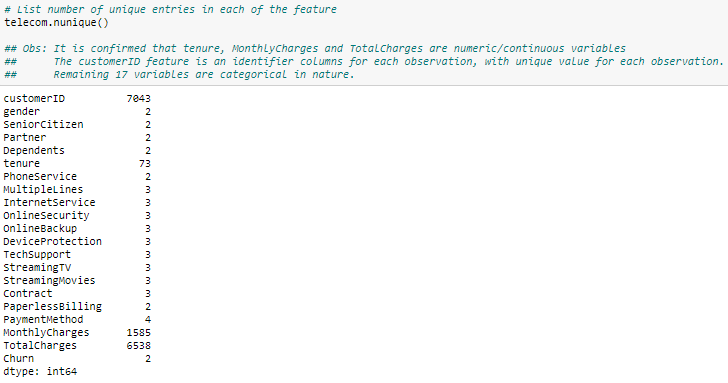
Once the missing values are addressed appropriately, a quick check for one sample record is made to make sure the changes are as expected.



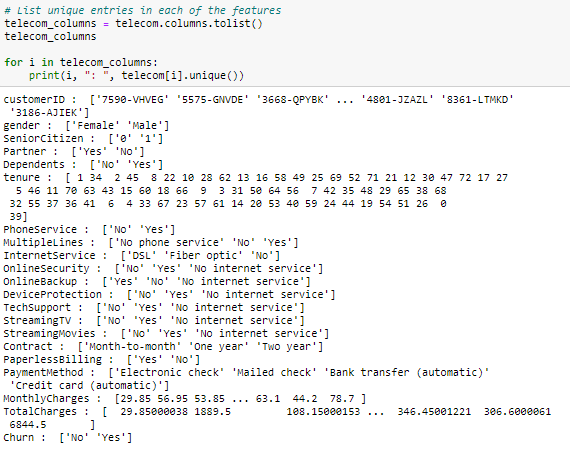
Similarly, the data type for the *SeniorCitizen* variable is also changed to string (object) type with the help of the following code. Following these two transformations, a quick check for missing values in the *telecom* data frame is performed again to make sure there are no more additional missing values are introduced.



Now, a check for the total number of unique values per variable can be performed to under what kind of variables the *telecom* data frame has.



After checking the number of unique values per variable, it would be better to find unique values per variable.



Based on the sample data, variable data types, and the total number of unique values per variable within the *telecom* data frame, certain observations about the input variables can be made. These observations about these variables are:

### Input/Predictor Variables

1. customerID – unique record identifier with 7043 unique entries (string)
2. gender (nominal binary: “Female”, “Male”)
3. SeniorCitizen (nominal binary: “0”, means “No, “1”, means “Yes”)
4. Partner – Whether a customer a commercial partner (nominal binary: “Yes”, “No”)
5. Dependents (nominal binary: “Yes”, “No”)
6. tenure – Number of months a customer is with Telco (numeric/continuous)
7. PhoneService (nominal binary: “Yes”, “No”)
8. MultipleLines (nominal categorical: “Yes”, “No”, “No phone service”)
9. InternetService (nominal categorical: “DSL”, “Fiber optic”, “No”)
10. OnlineSecurity (nominal categorical: “No”, “Yes”, “No internet service”)
11. OnlineBackup (nominal categorical: “Yes”, “No”, “No internet service”)
12. DeviceProtection (nominal categorical: “No”, “Yes”, “No internet service”)
13. TechSupport (nominal categorical: “No”, “Yes”, “No internet service”)
14. StreamingTV (nominal categorical: “No”, “Yes”, “No internet service”)
15. StreamingMovies (nominal categorical: “No”, “Yes”, “No internet service”)
16. Contract (nominal categorical: “Month-to-month”, “One year”, “Two year”)
17. PaperlessBilling (nominal binary: “Yes”, “No”)
18. PaymentMethod (nominal categorical: “Electronic check”, “Mailed check”, “Bank transfer (automatic)”, “Credit card (automatic)”)
19. MonthlyCharges – Currency is assumed to be USD (numeric/continuous)
20. TotalCharges – Currency is assumed to be USD (numeric/continuous)

### Predicted/Response Variable

1. Churn – Has a customer Churned or was retained with the company (nominal binary: “No”, “Yes”). This is chosen as a target since it helps to separate the customer base into two distinct classes between “Churned” for customers who left Telco and “Retained” for customers who are still with the company. Since this distinction between the two groups is an important factor for the outcome of the analysis, this must be the predicted variable.

# Data Preparation & Feature Engineering

Based on the observations made for the variables, the data for some of the variables can be more streamlined for a better understanding of the results. Some of the features like *TotalCharges* and *SeniorCitizen* were already updated to reflect correct data types, which is also part of feature engineering.

In addition to these changes, after checking the unique values of the variables, it can be deduced that some variables like *OnlineSecurity*, have values which can be replaced with another value from the unique list to reduce the overall number of levels. Also, some of the features like *PaymentMethod* have feature level labels that are verbose. These can be shortened for better readability and it could be helpful if this variable is encoded for predictive models.

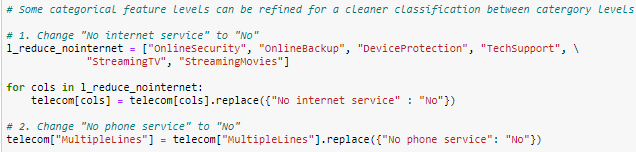
## Reducing Feature Levels

Some of the feature levels can be combined to reduce the number of levels within a feature. For example, feature *OnlineSecurity* has 3 feature levels, “Yes”, “No” and “No internet service”. It can be logically deduced that if a customer has not subscribed for an Internet plan, *OnlineSecurity* will not be available for that customer. As a result, “No internet service” can be replaced with “No”.

Similarly, the features having feature level as “No internet service” can be replaced with “No”. The list of features with this level are:

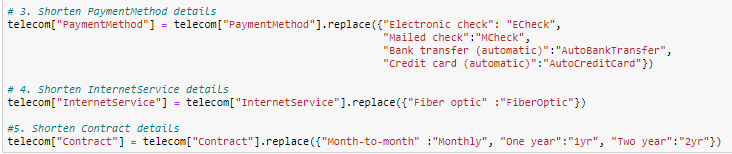
1. *OnlineSecurity*
2. *OnlineBackup*
3. *DeviceProtection*
4. *TechSupport*
5. *StreamingTV*
6. *StreamingMovies.*

Another feature *MultipleLines* have a similar feature level, which is “No phone service”. This can be replaced with value “No” as well.



## Rename Feature Levels for Readability

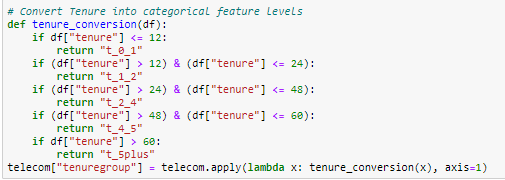
Some of the feature levels are very verbose and these could be shortened for better readability. For example, for the *PaymentMethod* feature, the lengthy name “Bank transfer (automatic)” can be replaced with “AutoBankTransfer”. Similarly, for *Contract*, feature level name “Month-to-month” can be replaced with “Monthly”. The detailed list of replacements is below.



## Converting numeric to ordinal categorical feature

It can be observed that *tenure* is a continuous feature, but has a finite set of unique values, ranging between 1-70. This can be converted into several “bins” or ranges depending on the observation value for *tenure*. Since this feature denotes the time in months a customer is an active subscriber with the company, the bins are converted to yearly buckets. For example, the first bin is for a duration of 0 to 1 year, the second bin could be for 1-2 years and so forth. The new feature is called *tenuregroup*.

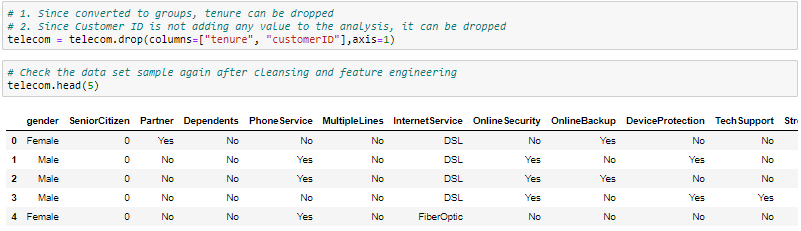
Performing this conversion helps in better interpretation of the information about the tenure of customers with the company, as well as check the churn rate of customers based on their tenure with the company.

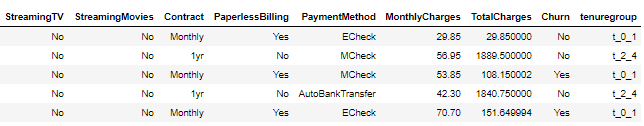


## Delete features

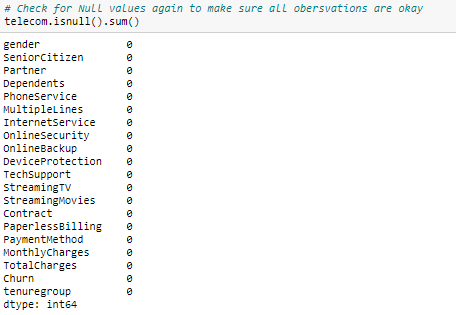
After data cleansing, some of the transformed features and the ones which would not be adding any value to the analysis or are redundant can be dropped from the data set. For example, the *tenure* feature was transformed into *tenuregroup*, *tenure* is redundant information and could be dropped from our analysis.

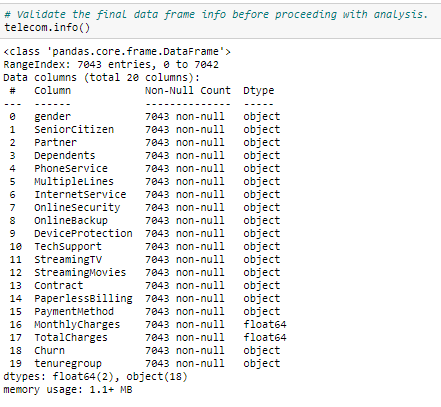
In another example, *customerID* does not add value in a sense where this feature cannot be meaningfully summarized. Since the current focus is not finding information about the individual customer but performing a high-level analysis and prediction, this feature can also be dropped. Below is the sample of the first 5 rows of the cleansed *telecom* data frame.





Since the data was transformed in some manner, a check for potential null values can be performed again. No null values were created as a part of these transformations. Also, check the updated *telecom* data frame information by each feature.





## Choice of Visualizations

There are a ton of options for visualizing the information that would help better and quickly understand details about the data. Depending on the type of data, type of aggregations, and what format the results need to be displayed in the visualizations, specific types of visualizations can be performed. Based on these, the types of visualizations used & the reason for those choices in this analysis are:

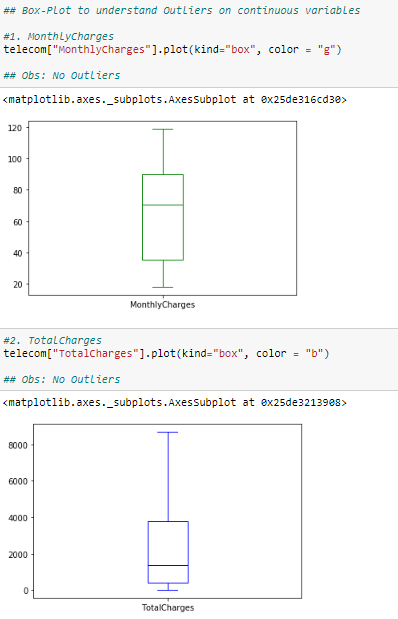
1. Count Plots: Used to display the counts of records in each categorical bin using bars, to get insights into the distribution of classes within a category.
2. Box Plots: Used to detect outliers within a variable.
3. Histograms & Density Plots: Used to understand the distribution of data within continuous variables.
4. Pie Charts: Used to check the proportions in percentage by each category level by target variable.
5. Scatter Plots: Used to check the interaction of two variables.
6. Heatmaps: Used to check the correlations between variables by color-coding them and check areas that matter most.
7. imshow: Used to display the data as an image for checking the confusion matrix for a model.
8. Regression plot: Used to check the residual assumptions for logistic regression.
9. Tree Plots: Using *Graphviz*, to plot one of the decision trees with a Random Forest.

## Addressing Anomalies & Missing Values

Before finally checking the statistical identities and summarizing the data for the features within the data set, any anomalies within the data, like outliers and missing values within the features must be addressed appropriately.

The missing values within *TotalCharges* are already addressed earlier. In addition to missing values, the incorrect data types for *TotalCharges* and *SeniorCitizen* features were also addressed appropriately.

Since there are only two continuous variables after data cleansing, a check for outliers for each of these variables is performed. Using box plots for MonthlyCharges and TotalCharges, outliers within each of these variables can be detected. As observed below, there are no outliers present in MonthlyCharges or TotalCharges.



# Data Exploration & Summary Descriptions

## Summary Statistics for Target Variables

Before checking any other statistics, the first step is to check some summary statistics for *Churn*. The below step is to check the summary of the *Churn* feature.



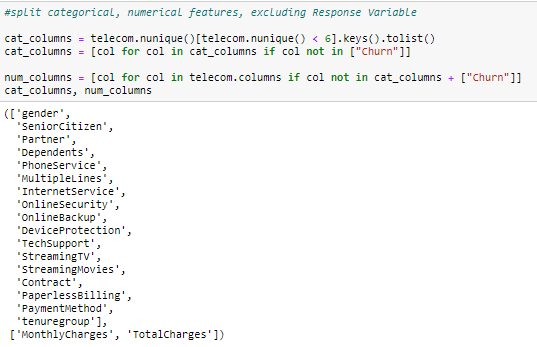
It can be observed that the percentage of customers leaving Telco is around 26.54%, whereas the percentage of customers choosing to stay with Telco is around 73.46%. In any business, a customer churn rate of 26.54% is considered very high and considered very risky, to a point where the company could go out of business in a few years. Since this feature provides insights about the most important information that the organization is looking for, whether the customer “churns” or “stays” with Telco, it makes sense to treat this variable as the target or response variable for the analysis.

Making the Churn feature as a target variable introduces another problem for the predictive methods. As clear from the above chart, the classes (Yes vs No) within the Churn variable are imbalanced in favor of “No” class, which are for customers choosing to stay with Telco. This imbalance could provide incorrect predictions and outputs for the algorithms if not handled before training the models.

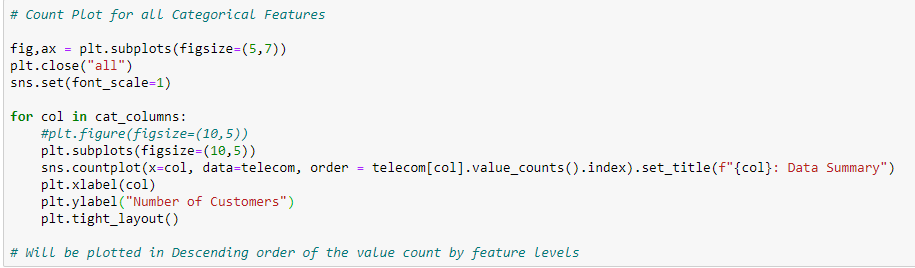
This imbalance of target variable classes is addressed by a technique known as the “*Synthetic Minority Oversampling Technique (SMOTE)*”, which upsamples the minority classes to generate balanced target variable classes. The implementation of this technique is discussed in detail in the upcoming sections.

## Summary Statistics for Binary & Categorical Predictor Variables

Before generating the visualizations to understand the categorical variables more in detail, two lists of all categorical and numerical variables are created for ease of plotting these variables. It needs to be noted that the *Churn* variable is excluded from these lists since that variable would be treated separately.



The plots for each of the binary & categorical predictor variable is created by the following code.



### Binary Predictor Variables Summary

These plots represent the frequency of each of the feature levels within the binary predictor variable.

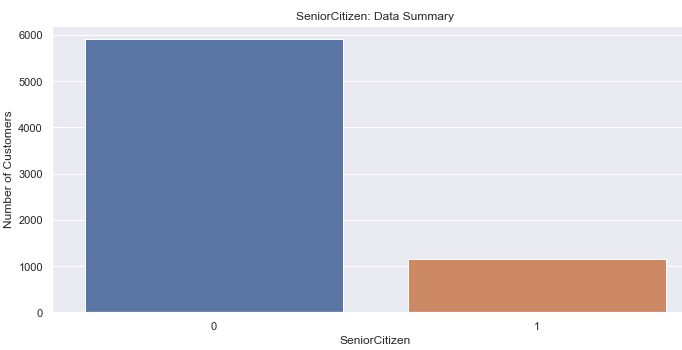
#### gender

The below frequency plot shows that classes within the *gender* variable are approximately equal. This could mean



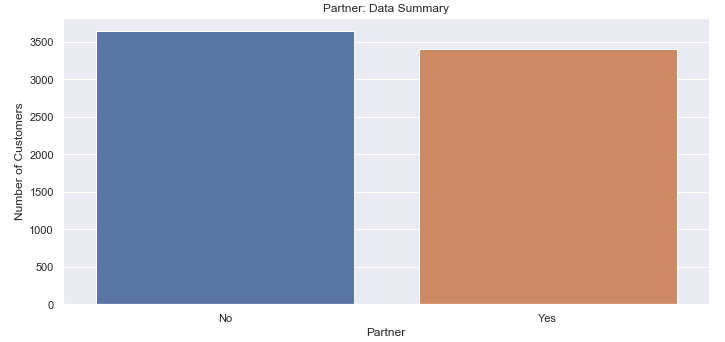
#### SeniorCitizen

The below frequency plot shows that the proportion of younger customers is higher than senior customers within the data set.



#### Partner

The below frequency plot shows that the proportion of commercial business partner customers and retail customers is approximately the same.



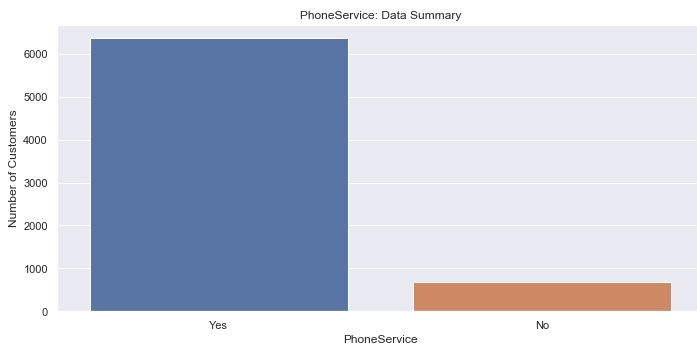
#### Dependents

The number of customers with dependents in the household is larger than customers having no dependents. This could mean younger single or married people with no dependents make up the majority of Telco’s customer base, based on this data set.



#### PhoneService & MultipleLines

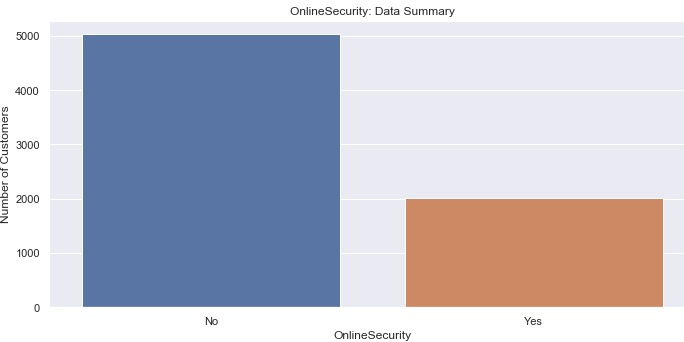
It shows that majority of the customers have an active phone service with Telco, but comparatively a smaller number of customers have multiple phone lines active with their accounts. This could either mean that most customers are part of the bundled packages to get better-packaged deals or need an active permanent landline.

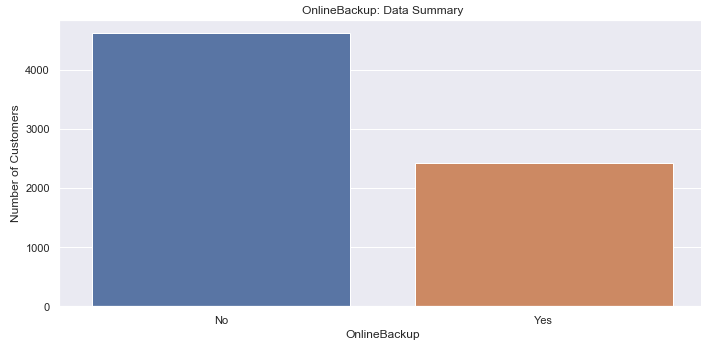


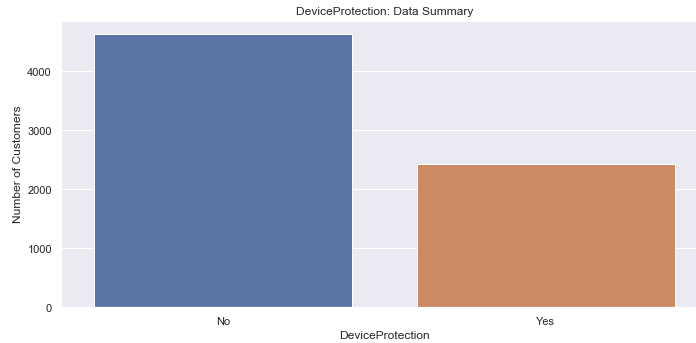


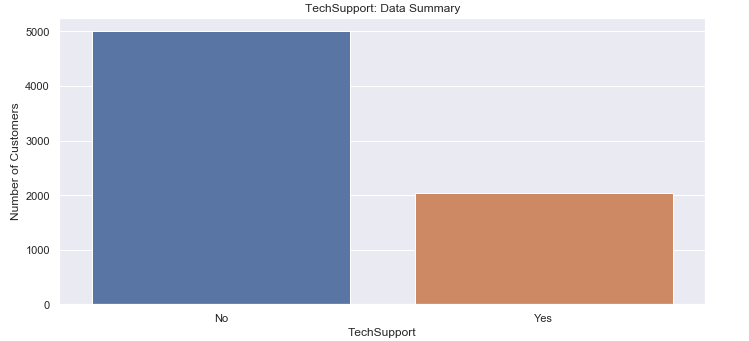
#### OnlineSecurity, OnlineBackup, DeviceProtection & TechSupport

It can be observed that very few customers have opted for online security and online data back up plans with Telco. Also, very few customers have opted for online device protection and preferred technical support plans.



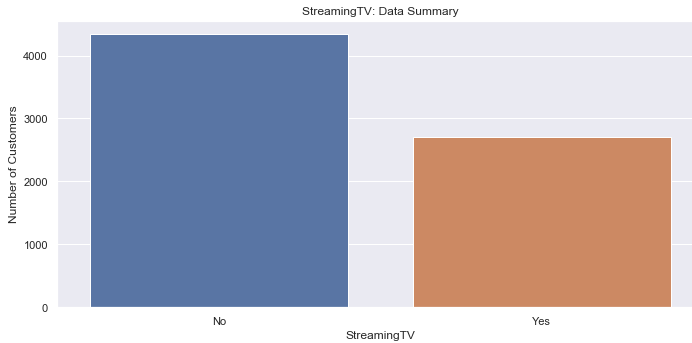


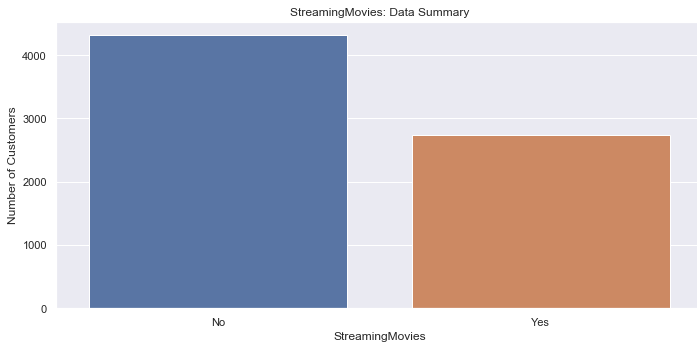




#### StreamingTV and StreamingMovies:

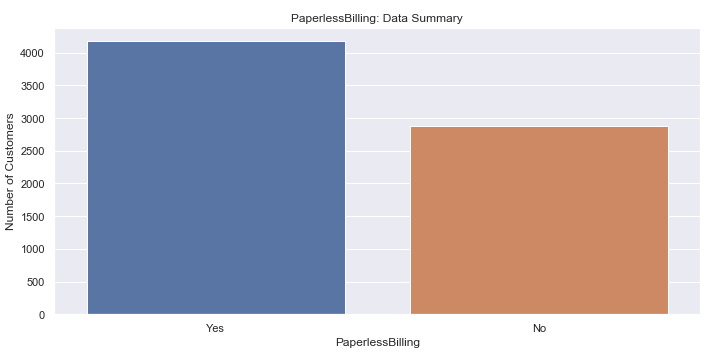
This makes it clear that fewer number of customers like to stream live TV & on-demand movies through Telco on their smart devices, like their phones or tablets.





#### PaperlessBilling

It can be deduced that more number of customers like to opt for paperless bills than old school paper bills in their mails.

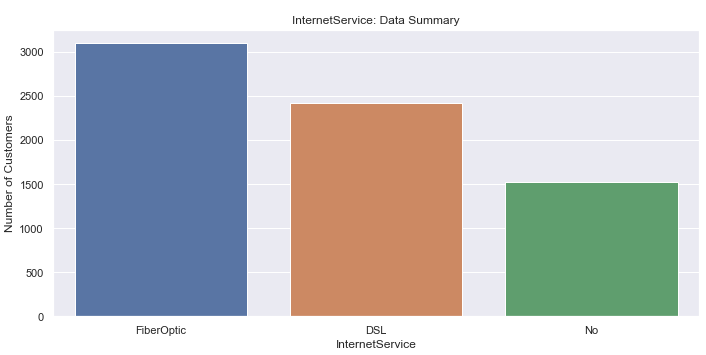


### Categorical Predictor Variables Summary

These plots represent the frequency of each of the more than 2 feature levels within the categorical predictor variable.

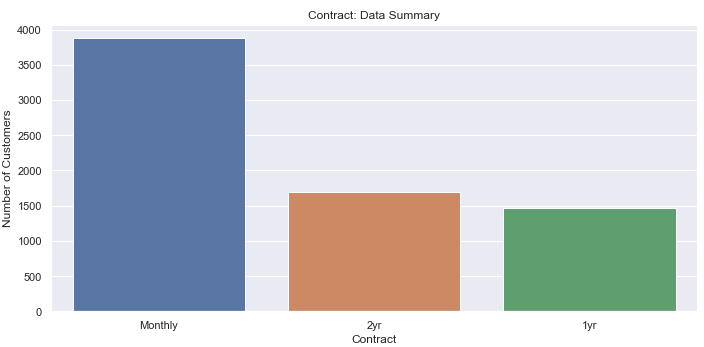
#### InternetService

Based on the below plot, within the group of customers subscribed to Internet service, the majority of the customers opt for fiber optic style of internet connection between fiber optic and DSL connection options. A group of customers does not have active Internet service with Telco at all.



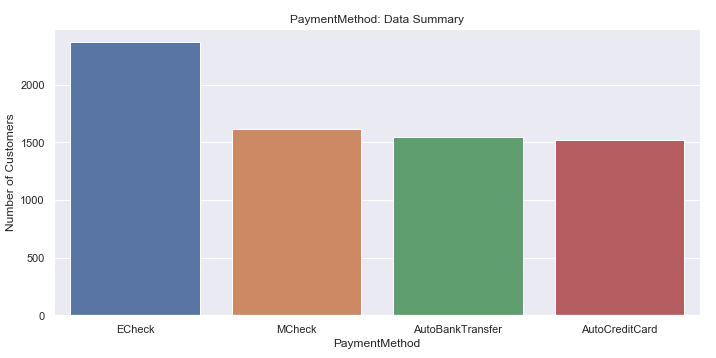
#### Contract

The highest proportion of customers are on a month to month contract type, while the customer groups with 1-year and 2-year contracts are approximate of the same proportions. Since the customers with monthly contracts are not bound to stay with the company for an extended time, this customer group could be focused to possibly convert to have long term contracts for better customer churn control.



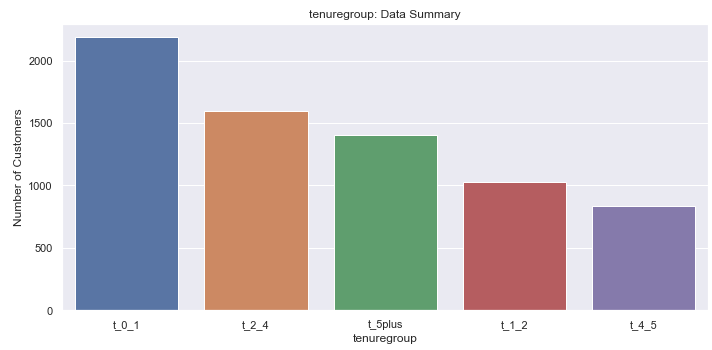
#### PaymentMethod

It can be noted that more customers opt for an E-check or electronic check than any other method of payment.



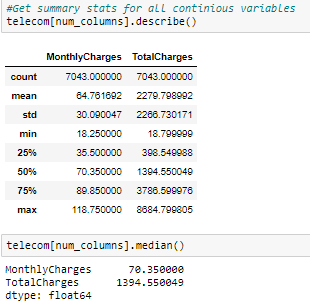
#### tenurgroup

It can also be noted that most of the customers are either new customers or are not staying for more than a year with the company. While there are enough customers who are happy with Telco to stay for more than 5 years with the company, based on the graph below, the focus can be directed towards customers who are fairly new or for ones where the 1 or 2-year contract is up for renewal.

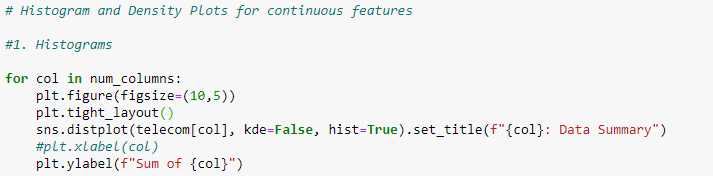


## Summary Statistics for Continuous Predictor Variables

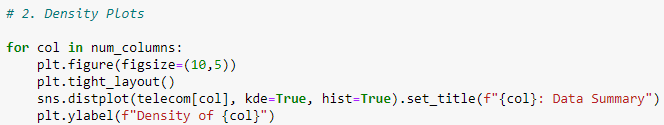
From the summary statistics, it can be observed that the average monthly charge for the customer is $64.76 and the total charges accrued per customer is $2279.73, with a standard deviation of $30.09 and $2266.79, respectively. It should also be observed that the range for *TotalCharges* is very wide at $8666, which could mean that there are some customers with long-standing subscriptions with Telco.



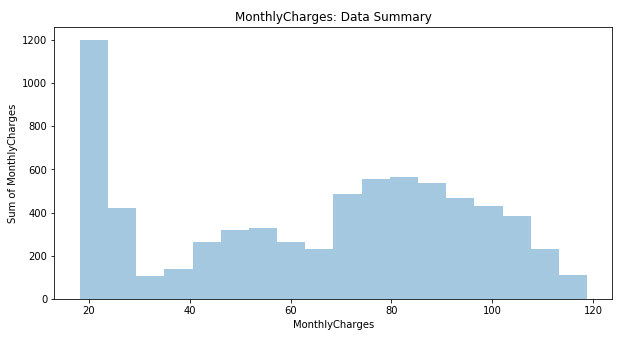
Two of the best ways to summarize continuous variables are histogram frequency distribution and density plots for the variable. The following code generates the histogram plots for both *MonthlyCharges* and *TotalCharges*.

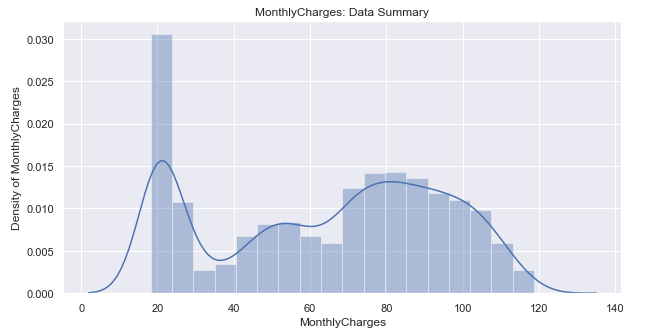


The below code generates the density plot for *MonthlyCharges* and *TotalCharges*.



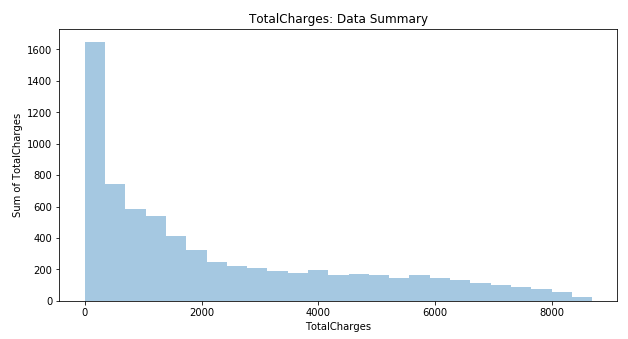
### MonthlyCharges

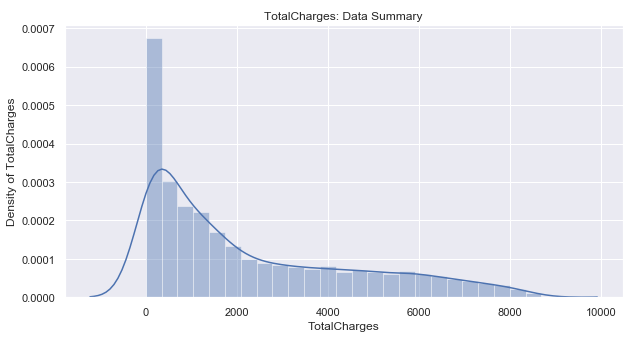




Based on the above histogram & density plot, it can be observed that the highest percentage of customers have monthly charges between $20-$30 and the second highest concentration is between $70-$110. This means that there are two customer groups, one with the minimum number and the cheapest plans available at Telco. The other group is subscribed to expensive plans, potentially with subscriptions to multiple services through Telco.

### TotalCharges

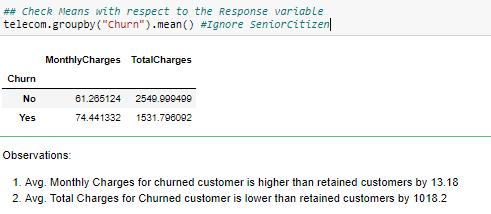




Based on the above histogram & density plot, it can be observed that the highest concentration of customers has total charges between $0-$2000. This means that most of the customers are either new or short-term customers or long-term patrons with cheaper plans.

# Predictor Variables Bivariate Analysis

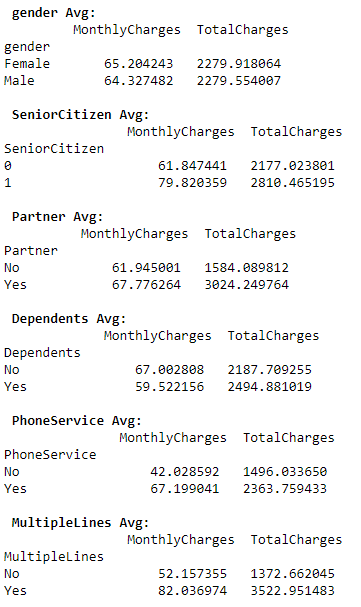
Before analyzing the predictor variables by Churn, checking the average charges by each of the feature levels for each of the categorical predictor variables will give some good insight into the data.

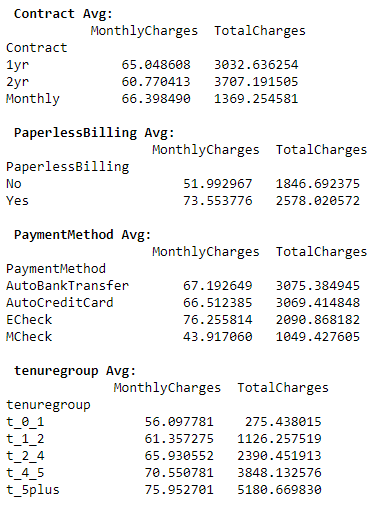
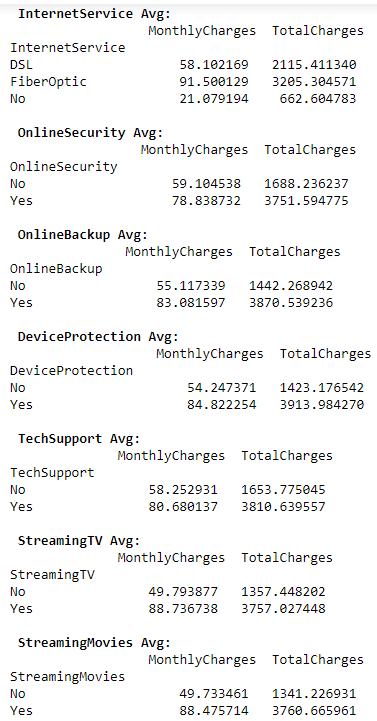


The calculation above shows that the average monthly charge for churned customers is higher than retained customers by 13.18, and the average total charges for churned customers are lower than retained customers by 1018.2.



The above code generates the summary data for average monthly & total charges for each of the categorical variables.



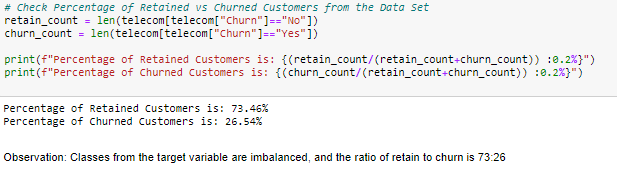


Based on the above summary, certain interesting observations can be made, such as:

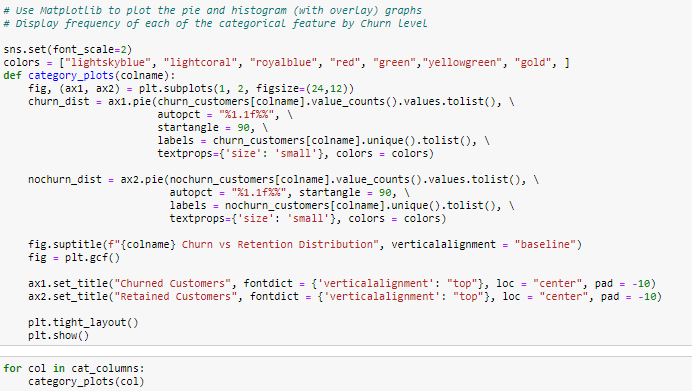
1. Senior citizens tend to pay more than their younger counterparts.
2. Commercial partners tend to pay more monthly and total overtime than retail customers.
3. Customers who do not have dependents pay more monthly, but less total overtime compared to ones with dependents.
4. Customers who have phone service and multiple lines pay more than customers with no phone service and a single line, respectively.
5. Customers opting for fiber optic Internet service pay the highest per month charge than DSL and ones with no internet subscription.
6. Customers opting for online security, online backup, premium tech support, and device protection pay significantly higher than customers opting out of these options.
7. Customers opting for streaming live TV and movies on apps also tend to pay significantly higher than their counterparts who do not subscribe to these options.
8. An interesting observation regarding the average monthly charge by contracts is that, although highest among three types of contracts, monthly subscribers pay about the same monthly charge as customers with a 1-year contract. The difference between monthly subscribers to 2-year contract customers is not too high either. But the total charges cost is highest with customers with a 2-year contract with Telco.
9. Customers with paperless billing options are paying higher average charges.
10. Customers who pay their bills via electronic check pay the highest average charges among all 4 payment types.
11. As the tenure for the customers' increase, the average per month and total charges tend to get higher.

## Categorical Predictors’ Proportion distribution by Churn

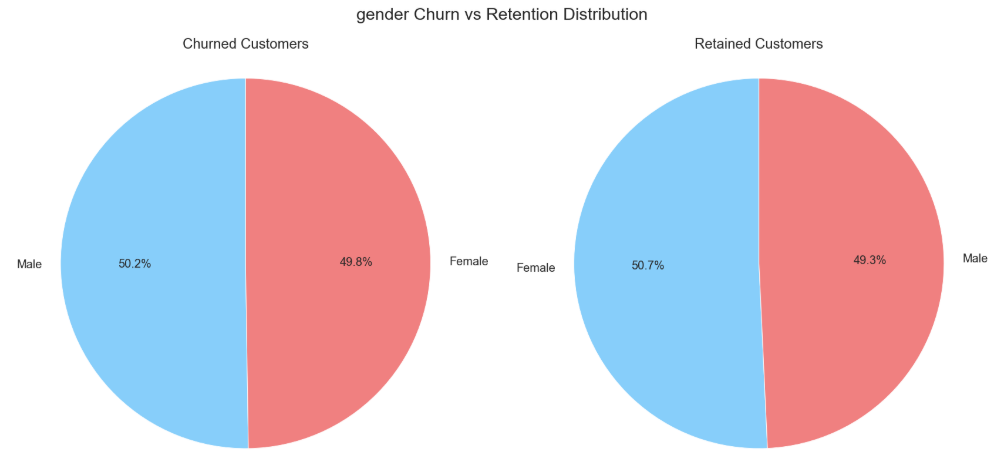
Splitting the *telecom* data frame into separate data frames by the Churn status of the customers will generating the visualizations easier. The following code creates two new data frames, *churn\_customers* and *nochurn\_customers*. As the name suggests, the first data frame will have churned customers and the latter one where customers chose to stay with Telco.



The following code will generate the proportions of churned vs retained customers by each of the predictor feature levels.

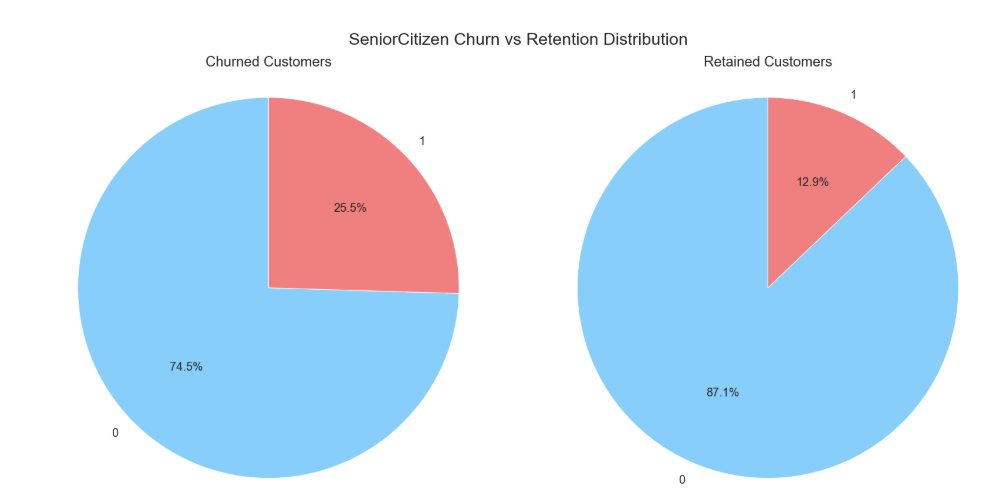


### gender by Churn



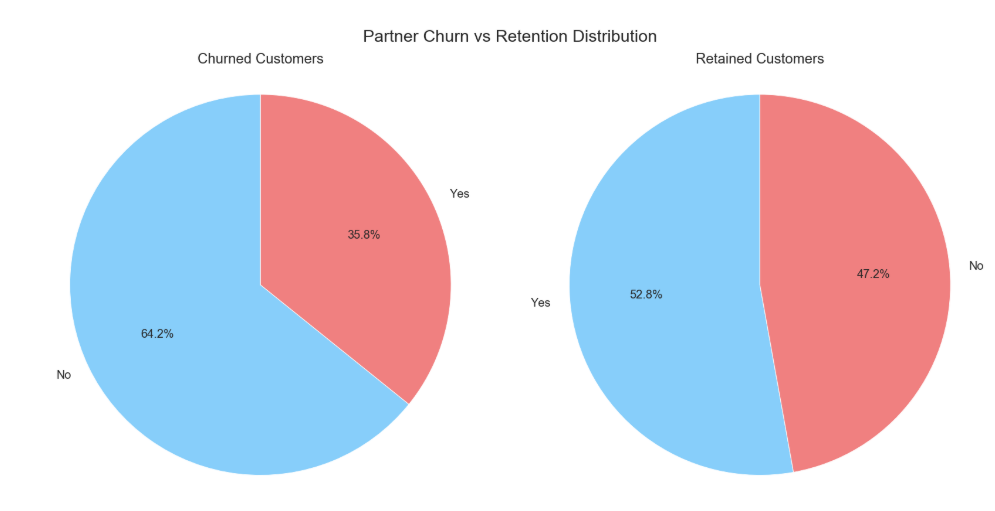
It can be noted that gender has an almost equal percentage of churned and retained customers by males and females.

### SeniorCitizen by Churn



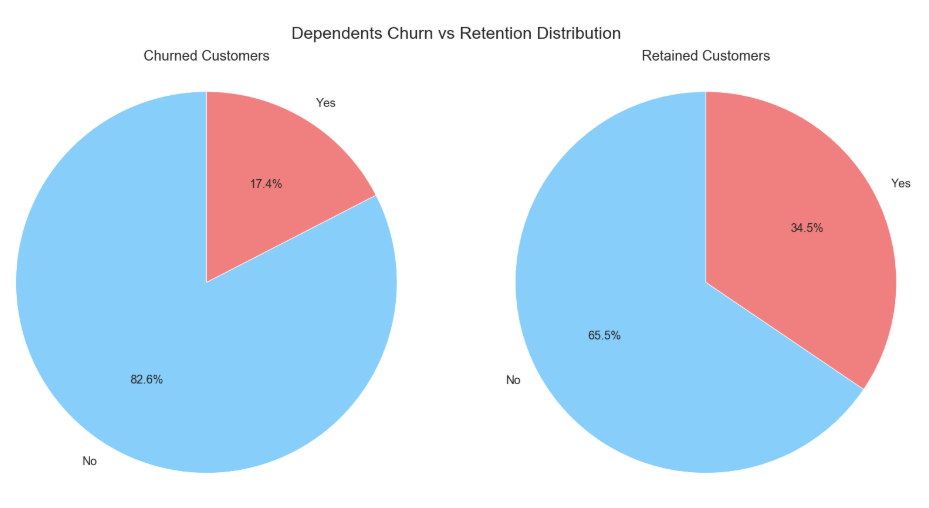
Out of all churned customers, 74.5% are young patrons, whereas only 25.5% of senior citizens are choosing to leave Telco. On the other hand, from all retained customers, 87.1% of young customers and 12.9% of senior customers chose to stay.

### Partner by Churn



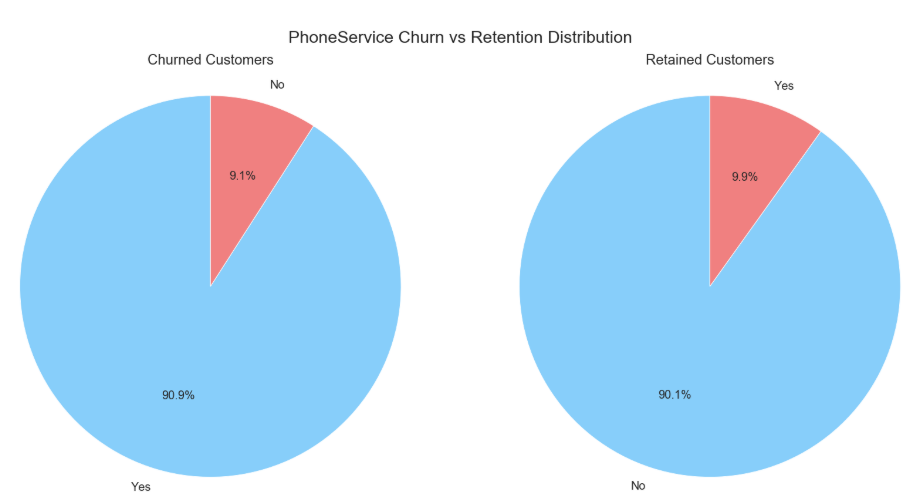
Out of all the churned customers, 35.8% are commercial partners and the majority are retail customers. Whereas within retained customers set, the proportions are approximately the same for commercial and retail customers.

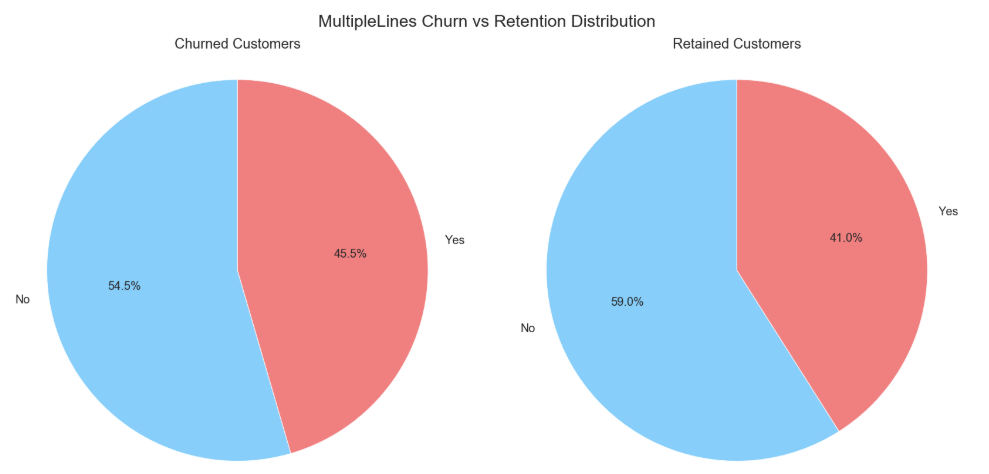
### Dependents by Churn



82.6% of all churned customers do not have dependents, compared to 65.5% of all retained customers.

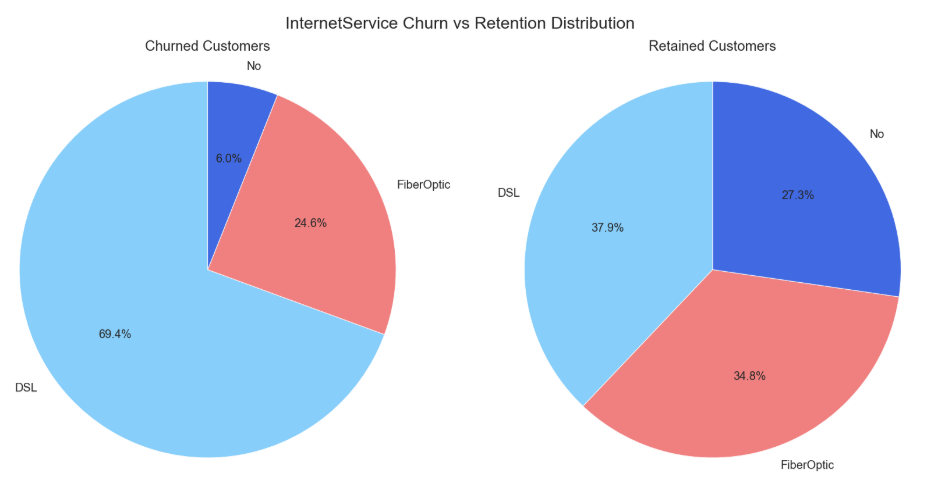
### PhoneService and Multiple Lines by Churn





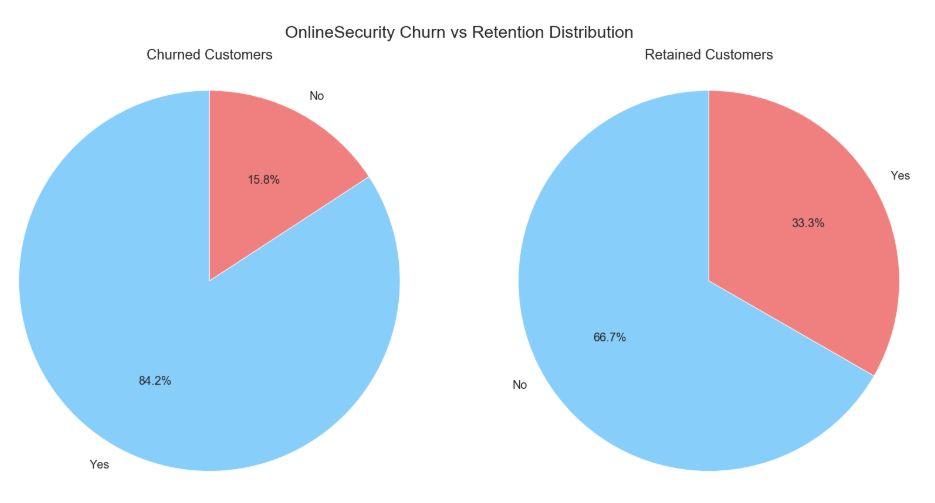
The proportions for phone service and multiple lines are approximately the same within both churned and retained customers.

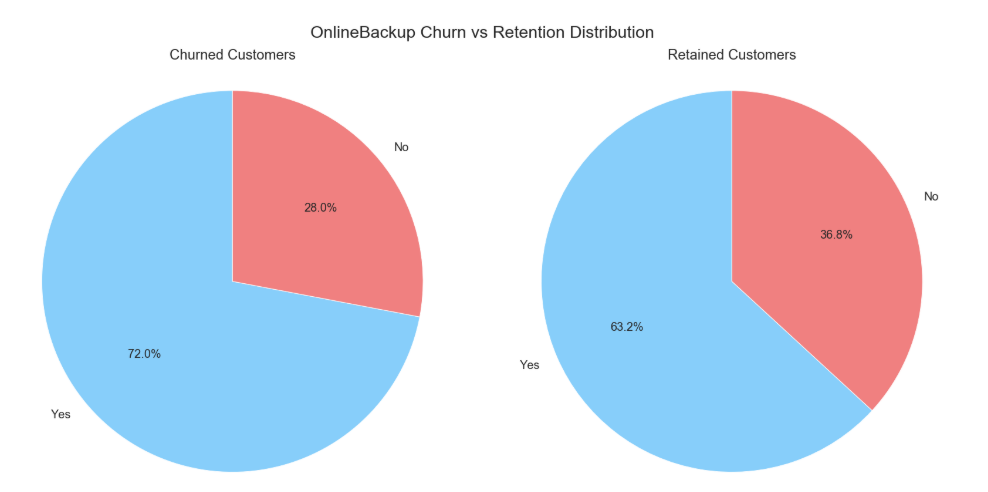
### InternetService by Churn



Among all churned customers, the majority of them had a DSL connection for Internet service followed by fiber optic. Only a small percentage of customers did not have an internet connection. The proportions among retained customers for the type of Internet service are approximately the same.

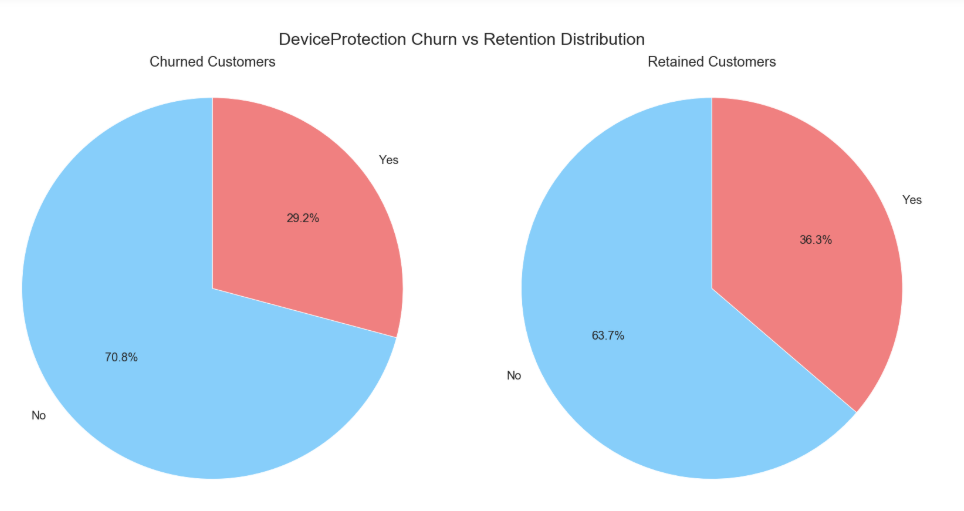
### OnlineSecurity and OnlineBackup by Churn





Among all churned customers, the highest churn rate is for customers paying for online security and online backup.

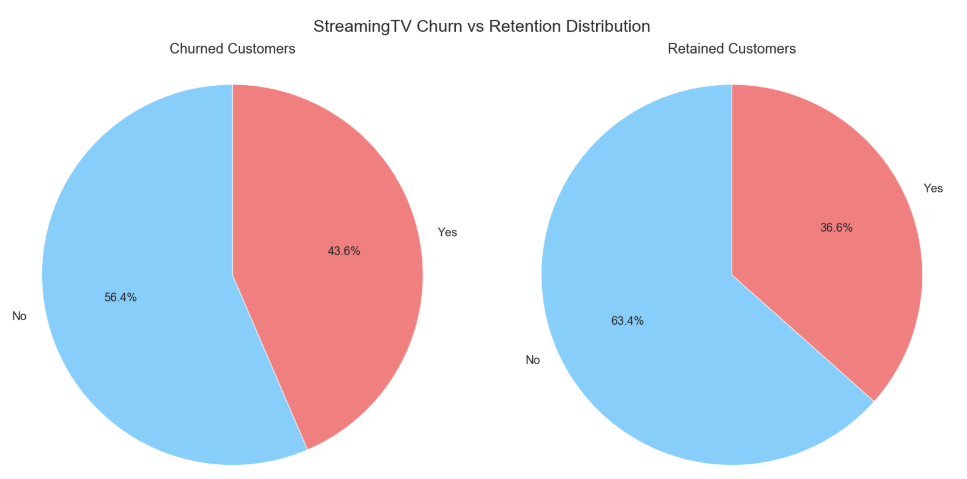
### DeviceProtection and TechSupport by Churn

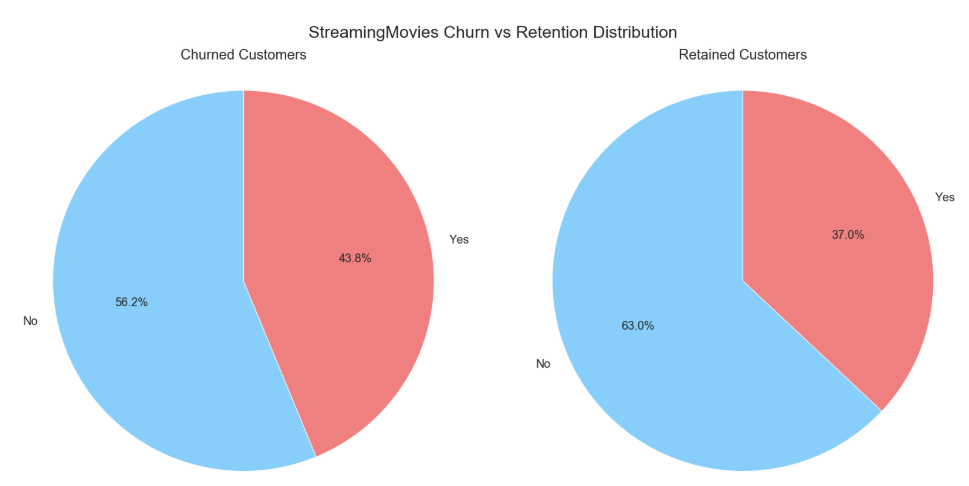




Among all churned customers, the majority of the customers did not opt for device protection and premium technical support.

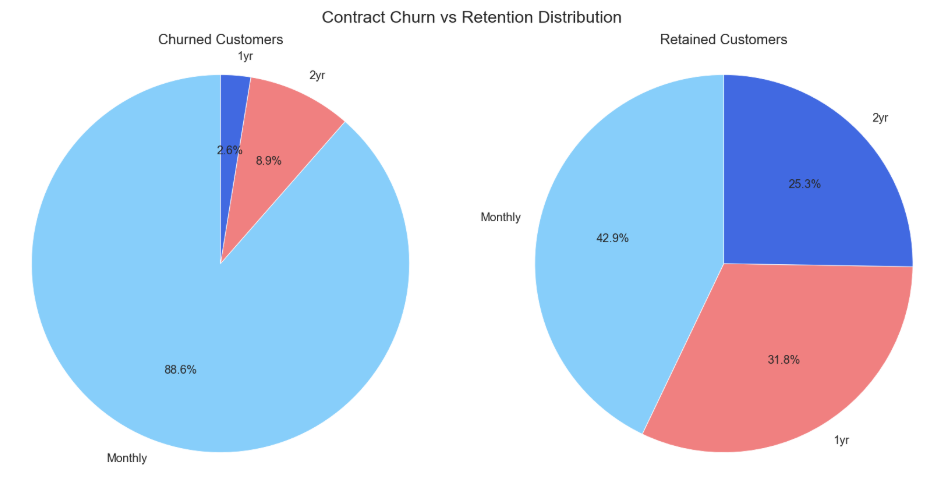
### StreamingTV and StreamingMovies by Churn





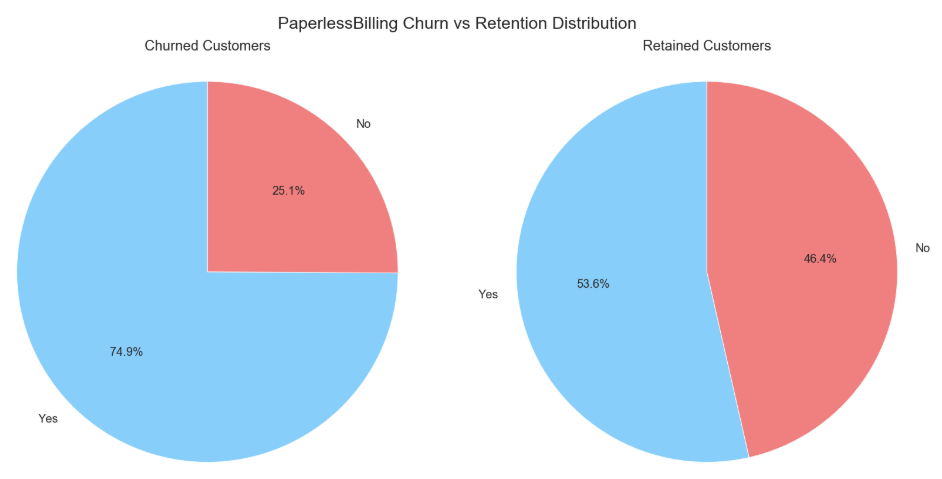
The percentage of customers not opting for streaming live TV and movies is approximately the same within the churned customers and retained customers, respectively.

### Contract by Churn



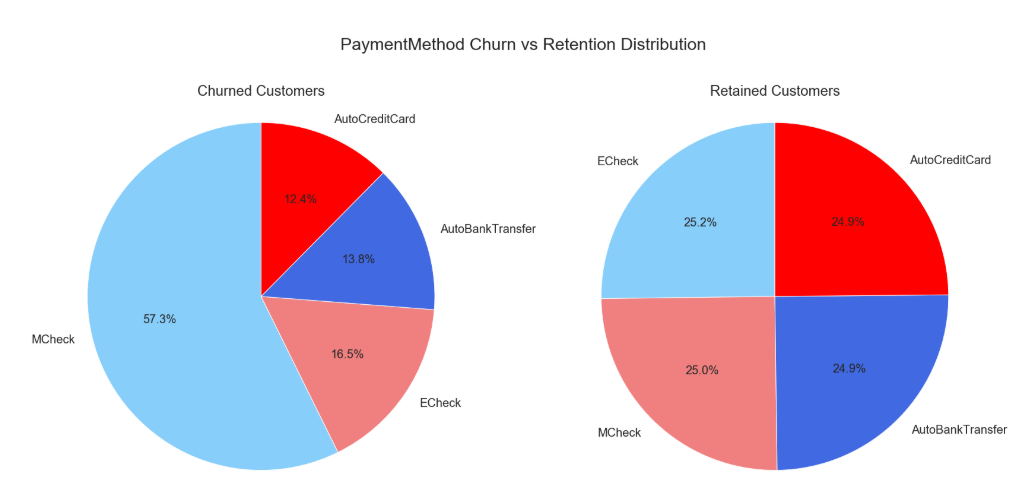
It is quite evident that the monthly subscribers are the highest among all the churned customers with churn rate at 88.6%.

### PaperlessBilling by Churn



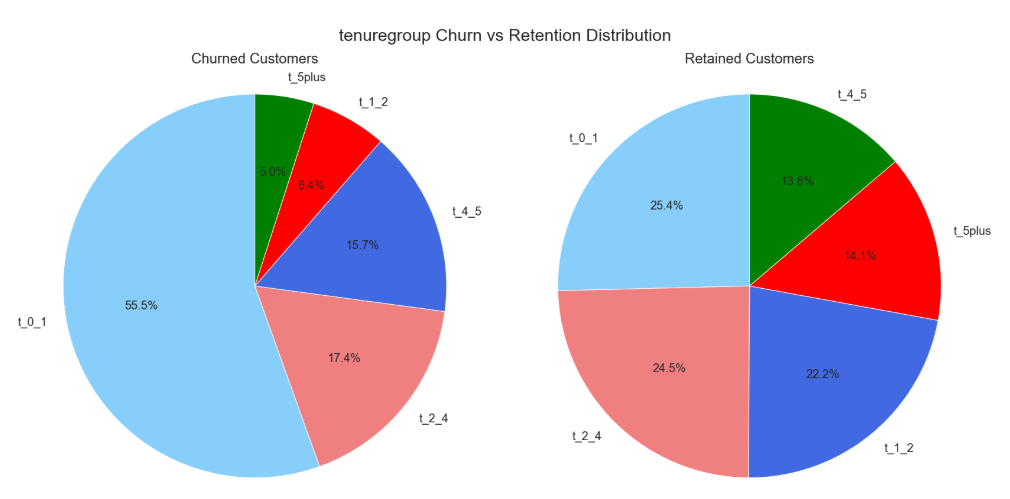
Customers opting for paperless billing has the highest churn rate among churned customers.

### PaymentMethod by Churn

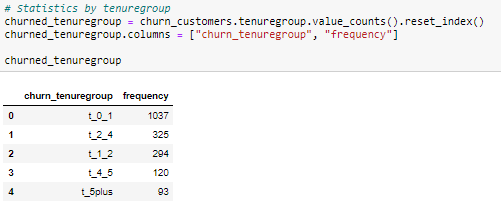


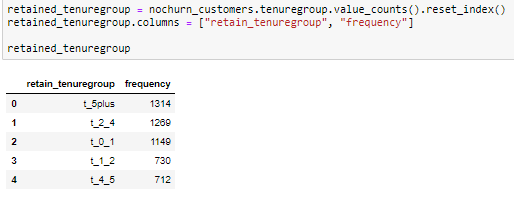
Customers paying by manual check has the highest churn rate among churned customers. Whereas, all payment methods have approximately the same distribution rate between retained customers.

### tenuregroup by Churn

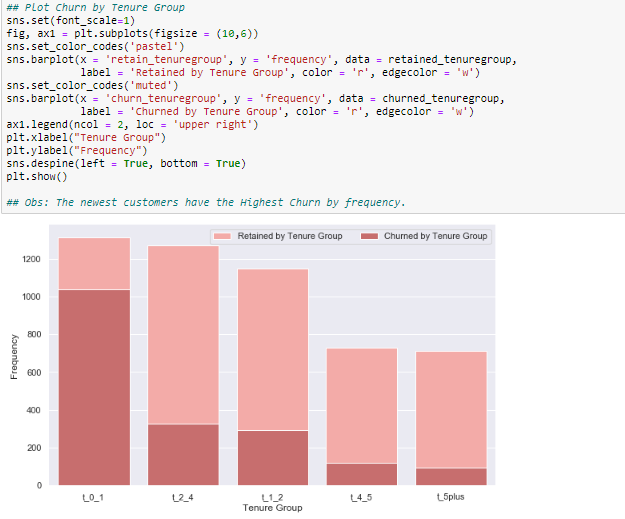


The tenure group t\_0\_1, which is a class of customers with tenure duration 1 year or less, has the highest churn rate at 55.5%, followed by t\_2\_4 class at 17.4%. Checking the total number of churned and retained customers per tenure group will give more insight into it.





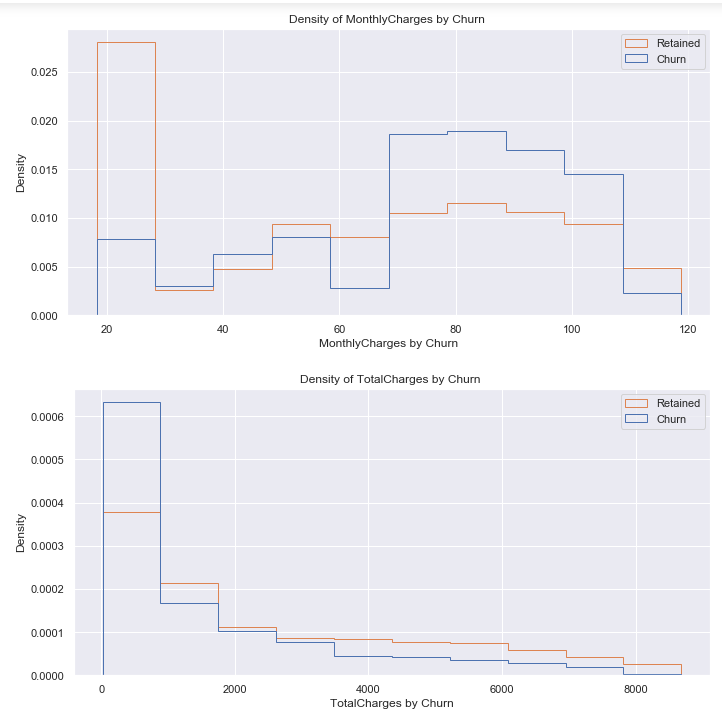
It can be noted that the churned class within the *Churn* response variable was very imbalanced towards the lower end. Thus, the number of churned customers in *t\_0\_1* class is quite concerning and should be addressed in the mitigation plan. The below chart gives a visual comparison between churned and retained customers by *tenuregroup*.



## Numerical Predictors’ distribution by Churn

The below code will generate an overlaid histogram for *MonthlyCharges* and *TotalCharges* by *Churn*.



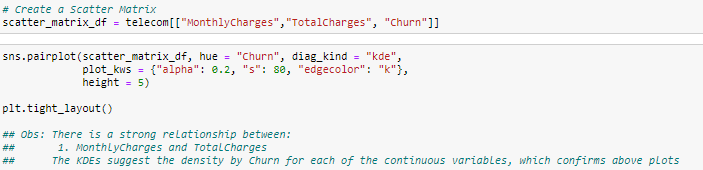


From the above two histograms, two interesting observations can be made.

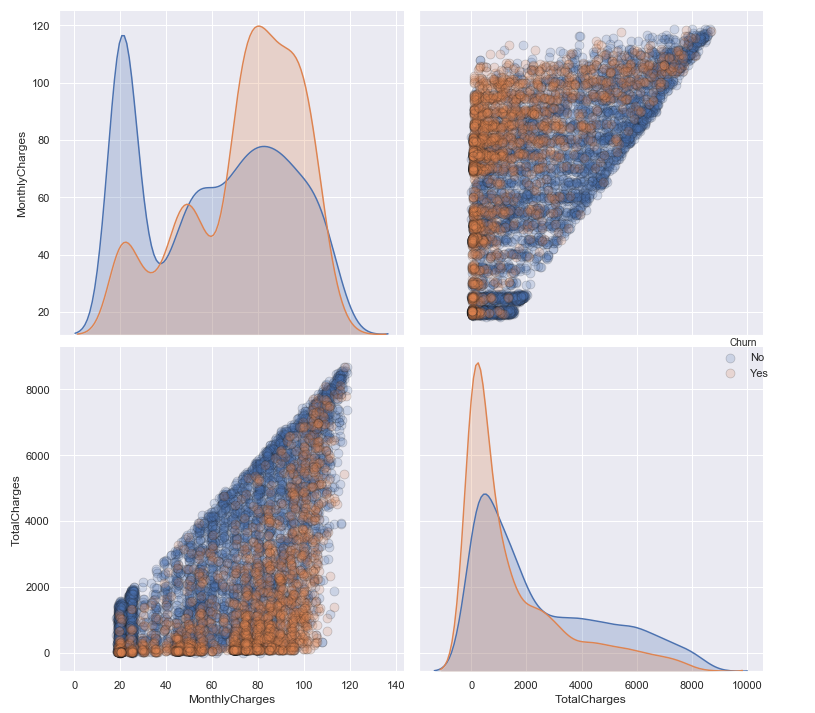
1. Most of the churning customers are paying monthly charges between $70 to $110
2. Most of the churning customers are accruing total charges between $0 to $2000.

## Correlations between continuous variables by Churn

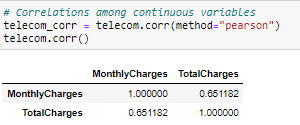
A quick multicollinearity check is performed between the two continuous variables to check if there is a high interdependency among these two variables. By generating the density scatterplots and creating a correlation matrix, it can be easily visualized and deduced whether high correlations exist between the two.

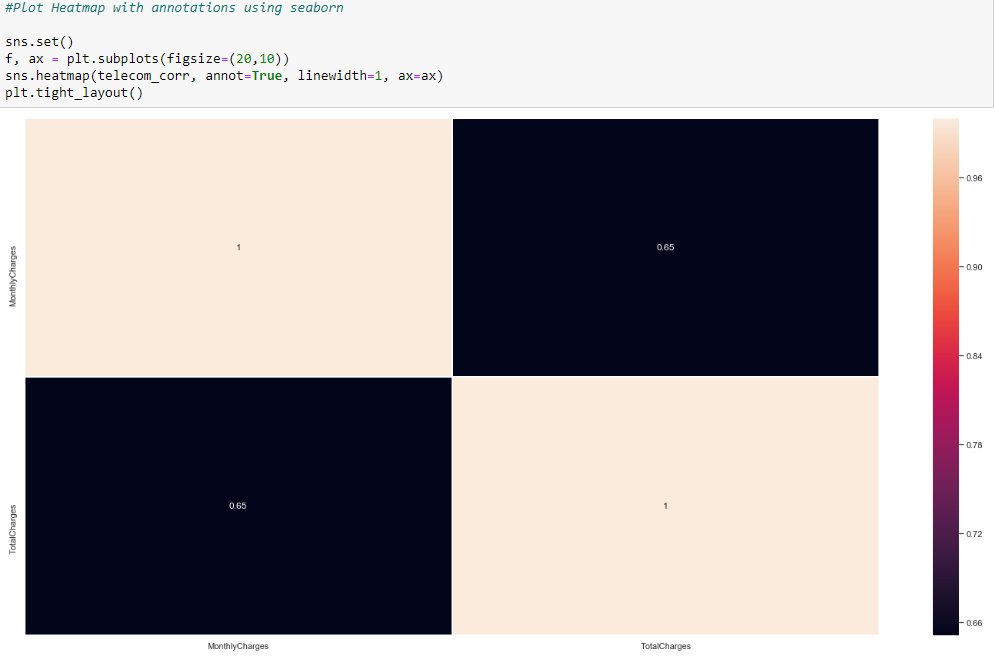


It can be observed from the below scatterplot, that a positive relationship does exist between the two variables.



A correlation matrix and the corresponding plot is created next.





It can be observed that there is a positive correlation between the two variables, but it is not a very strong correlation. Thus, these two variables are retained in the *telecom* data frame at this stage and will be looked at further if the dimension reduction algorithm removes one or both features in further sections.

# Evaluative models for Predictions

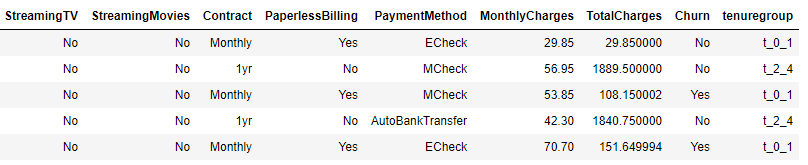
There are a bunch of predictive methods available to successfully perform predictions for the binary response variable. Since the predictors, in this case, are a combination of binary, categorical and continuous variables, two of the best algorithm choices are Logistic Regression and Decision Trees (Tuffery, *Table 6.2 Predictive Methods* 2011). As a best practice, several different algorithms are tried and tested before finalizing the model to be deployed. In this case as well, both Logistic Regression and a refined version of the Decision Trees – Random Forest algorithm are implemented, and the best performing model is chosen for recommendation.

The decision to use these two algorithms out of all the available choices was made because both these algorithms work well with a combination of qualitative and quantitative predictor variables. Both algorithms can be used for the classification type of problems. In addition to these factors, Logistic Regression requires less computational resources and is very efficient to train (Kumar, *Advantages, and Disadvantages of Random Forest Algorithm in Machine Learning* 2019). Whereas Random Forest requires more computational power and resources but reduces overfitting thus reducing variance and improving accuracy (Kumar, *Advantages, and Disadvantages of the Random Forest Algorithm in Machine Learning* 2019). These two algorithms provide a good mix of options and these make them an obvious choice for implementation and validations.

## Data Pre-Processing

Many machine learning algorithms perform better when numerical input variables are scaled to a standard range (Brownlee, How to Use StandardScaler and MinMaxScaler Transforms in Python). Any algorithm which depends on the “vector distances”, for example, eigenvalues, is recommended, and in some cases necessary to perform transformations on the training & test data sets. Logistic Regression is one such algorithm, which performs better when the data is transformed into numeric input variables. A check for the top 5 sample records in the telecom data frame is made before the transformations are applied.





In addition to checks above, a copy of the current data of the *telecom* data frame is made to a new data frame named *original\_telecom* in case cleansed data needs to be referenced.



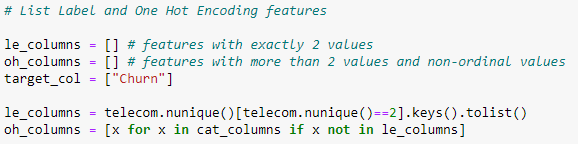
### Transformation of both Predictor and Response Variables

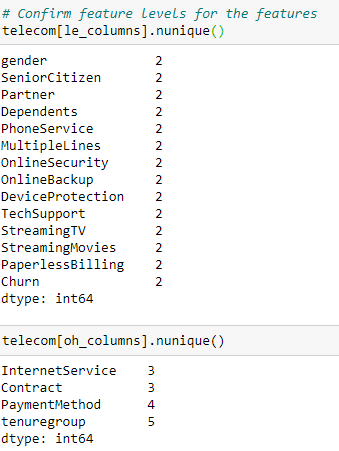
Label Encoding encodes the labels into model-understandable numerical data, between *0* and the *number of classes-1*. In the case of the *telecom* data frame, all the binary categorical predictor variables will be label encoded (Srinidhi, *Label Encoder vs. One Hot Encoder in Machine Learning*).

Another type of encoding is the one that converts categorical features as one-hot numeric arrays. The encoder derives the categories based on the unique values in each feature. This technique is called One-Hot Encoding. This will be applied to all non-binary categorical predictor variables in the *telecom* data frame (Srinidhi, *Label Encoder vs. One Hot Encoder in Machine Learning*).

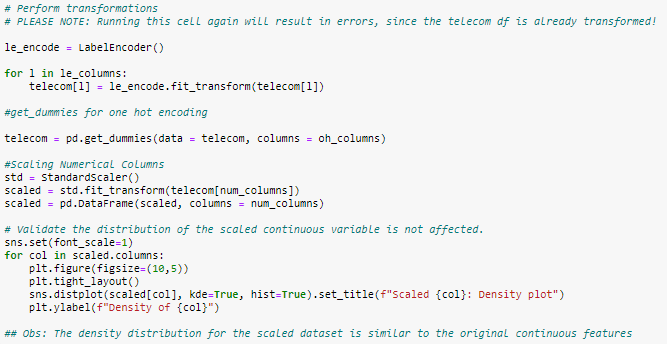
Standardization scales each numerical input variable separately by subtracting the mean (called centering) and dividing by the standard deviation to shift the distribution to have a mean of zero and a standard deviation of one (Brownlee, How to Use StandardScaler and MinMaxScaler Transforms in Python). In the case of the *telecom* data frame, the numeric features *MonthlyCharges* and *TotalCharges* will be standardized using the Scikit-learn package’s StandardScaler(). Generally, outliers have more impact on StandardScaler than MinMaxScaler, but since there are no outliers in these numeric variables, StandardScaler() function is used.

Two lists with column names for which label encoding and one-hot encoding must be applied are created for easy lookups. Also, validate the number of unique records for these columns.

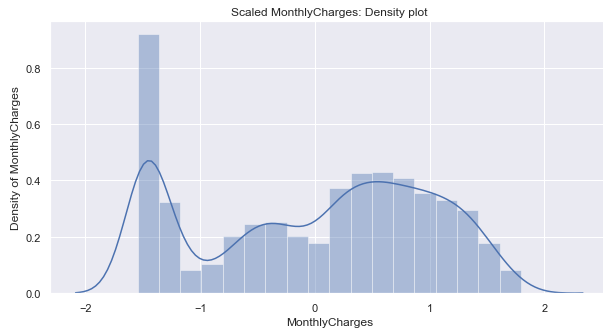


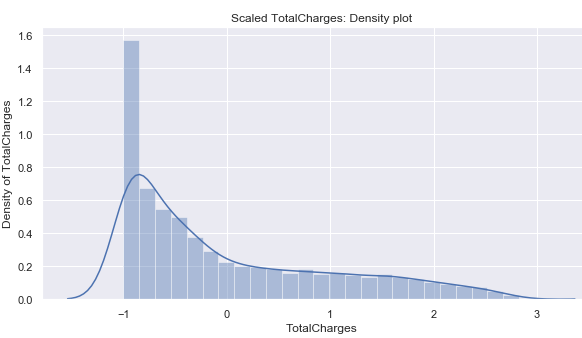


Now, perform all three transformations discussed above and check the distributions of numeric predictor variables to make sure if the distribution remained the same.



The plots for scaled numeric values are below.

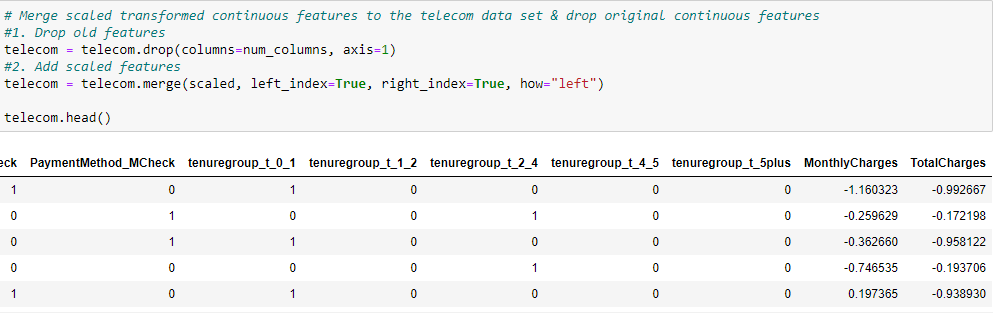




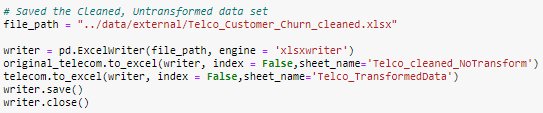
Validate one record from the cleansed data set and transformed data set to make sure if the data looks the same after transformations. The entries seem to match.

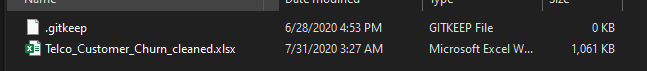


The transformed telecom data frame is now finalized, by dropping the old numeric variables and merging it with scaled values.



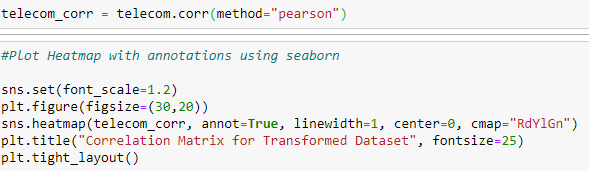
Please note, that the number of features has increased significantly from 21 features to 31, including the response variable. the final cleansed and transformed data can now be saved in an Excel workbook, with two worksheets, “Telco\_cleaned\_NoTransform” and “Telco\_TransformedData”. This workbook is saved on the path – Customer\_Churn/data/external

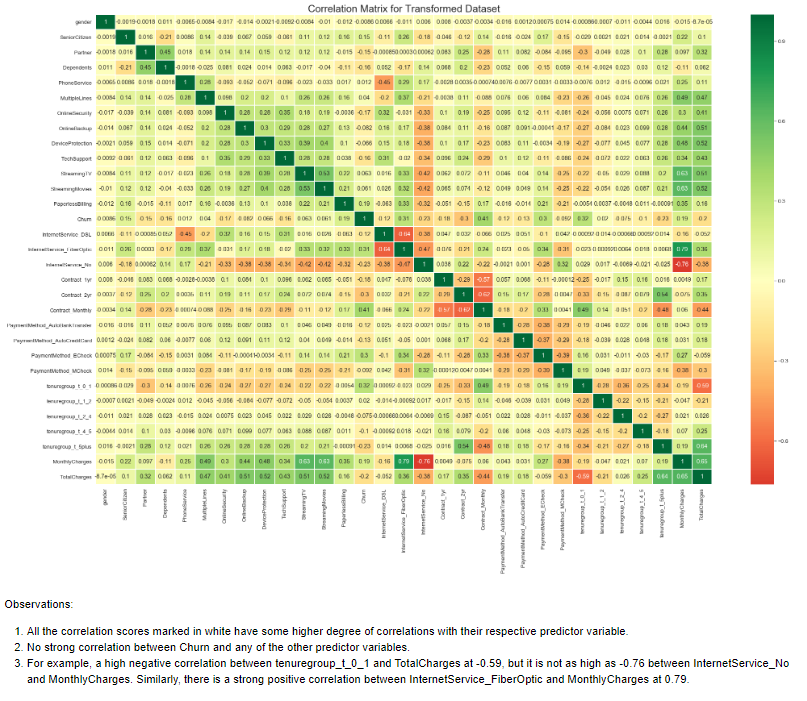




## Check Correlations between Transformed Variables

A quick check for correlations between the newly transformed variables can be performed to see if there are any multicollinear relationship among any of these variables. The code to generate the correlation matrix and plot the matrix, with the scores is below.



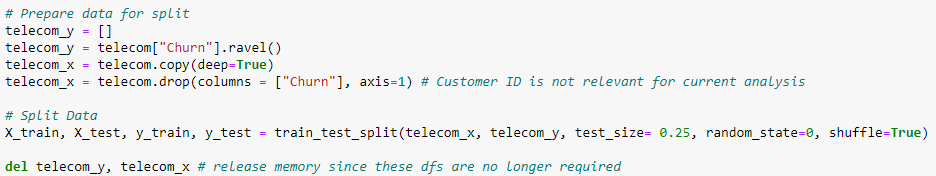


As it can be observed, there is no strong relationship between Churn and any of the other predictor variables. Also, all the correlation scores marked in white have some higher degree of correlations with their respective predictor variable. For example, a high negative correlation between *tenuregroup\_t\_0\_1* and *TotalCharges* at -0.59, but it is not as high as -0.76 between *InternetService\_No* and *MonthlyCharges*. Similarly, there is a strong positive correlation between *InternetService\_FiberOptic* and *MonthlyCharges* at 0.79.

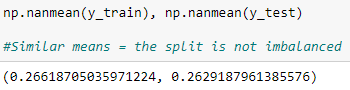
These highly correlated features can be removed from the data frame now manually or a sophisticated algorithm called *Recursive Feature Elimination (RFE)* can be used to systematically reduce the number of features within the data set to build a meaningful logistic regression model. A latter choice of using the RFE algorithm is made at this point to pick the top 15 predictor variables among 31 features. This list of features can later be refined by checking the p-values of these selected 15 features and the number of features can be further reduced, depending on the values.

## Train-Test Split of an initial training data set

Before building the models, the training data set is split into train and test sub-sets (Prabhakaran, "Linear Regression - A Complete Introduction in R with Examples", 2019). This is optional, but it is considered a best practice to perform this split. The simple reason is to check the performance of the model and build the best prediction model. This split is performed by the below code. This command set will perform the 75-25 percent split on the original training data.



The *X\_train* and *X\_test* matrices now have the predictor variables data and *y\_train* and *y\_test* have response variable data. A check is performed to know whether the split was balanced for the response variable or not. The means for both the lists are almost similar, which is a good sign.

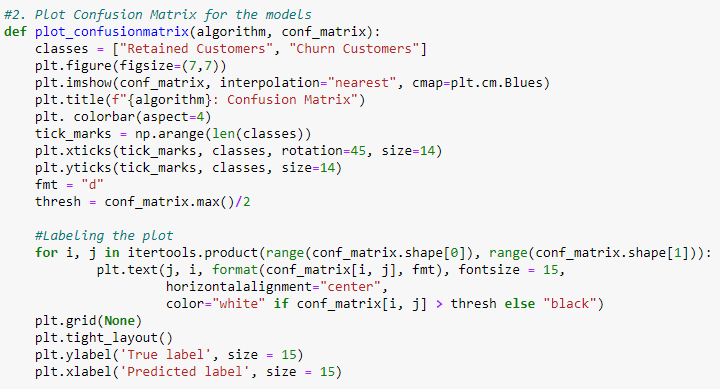


## Define Custom Functions

Since there would be a few different models created for comparisons, some tasks would have to be repeated to calculate scores and plot the necessary graphs. Instead of writing the same code repeatedly, few custom functions are implemented to speed up the analysis process.

### Plot Confusion Matrix

A custom function is written to plot the confusion matrix for the model, along with the values per class. This function accepts model name and confusion matrix numbers as parameters and plots the confusion matrix as an output.



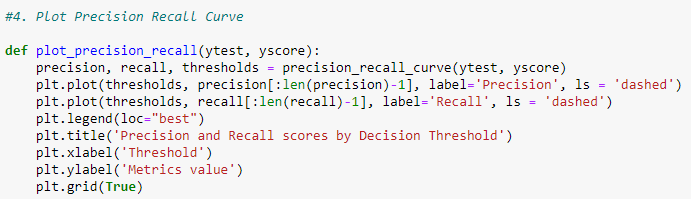
### Plot ROC-AUC curve

A custom function to plot the ROC-AUC curve and display the area under the curve is written below. This function accepts the model name, an area under curve score, false-positive rates, and true positive rates as parameters and plots the ROC-AUC curve for the model.



### Plot Precision-Recall Curve

A custom function to plot the precision-recall curve for the model is written below. The function accepts actual response variable values from the test set and score of the model from the decision\_function() of the model.



### Model Scores

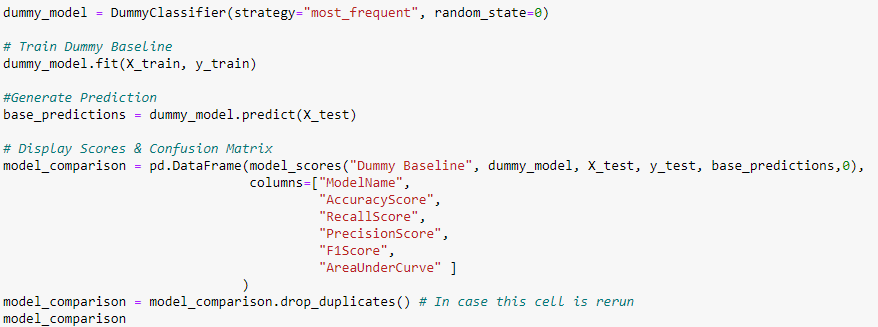
A custom function that displays model scores, classification report, and calls other programs for plotting the graphs, such as confusion matrix and AUC curve. It accepts the model name, an actual model for which the scores are calculated, test predictor variable matrix, actual test response variable data, model predictions array, and the index number to add scores to the data frame. It returns a data frame, which finally gets added to a *model\_comparison* data frame outside this function.



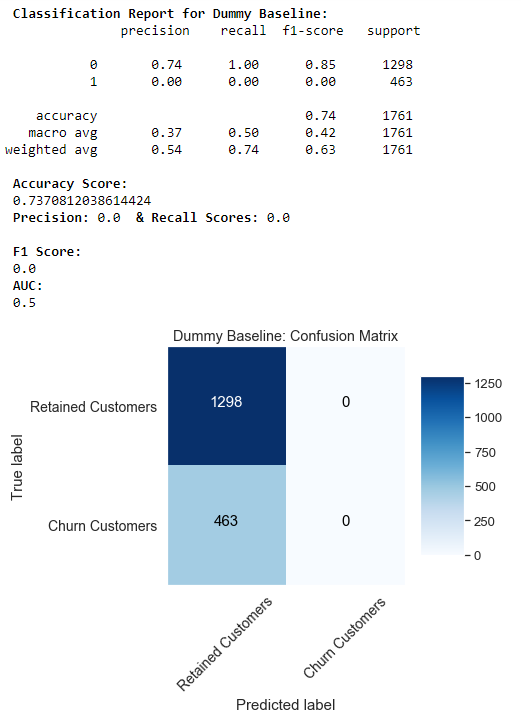
# Create Predictive Models

## Dummy Baseline Model

Although it is mandatory to create a dummy baseline model, it is generally considered a best practice to create these models for performance comparison purposes. These models are generally created to predict with a set strategy, like most frequently occurring responses. This model can be created using *DummyClassifier()* function from the *sklearn* library.



The output of running the above code is below.





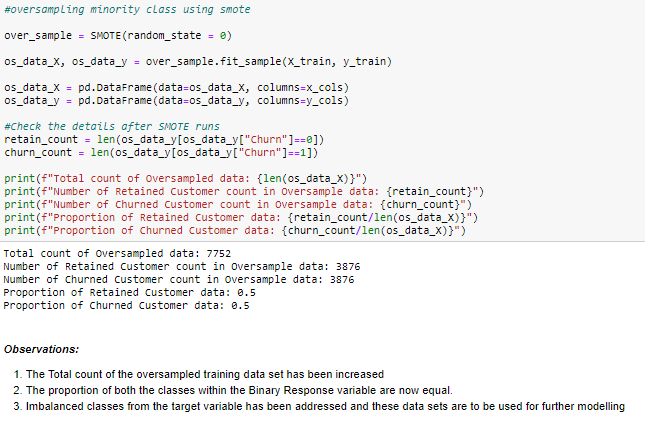
The accuracy score is 73.70% and precision, recall, and F1 score are all 0.0 since the model predicted only Churn value as “0”, which is the most frequently occurring entry within the response variable. This also results in AUC of 0.50 only. These scores although very bad, are expected since it is only a dummy baseline model without any inherent prediction logic.

## Synthetic Minority Oversampling Technique

The challenge of working with imbalanced datasets is that most machine learning techniques will ignore, and in turn have poor performance on, the minority class, although typically it is performance on the minority class that is most important (Brownlee, *SMOTE for Imbalanced Classification with Python* 2020). In this case, the customers churning (“Yes” or 1) is the minority class in the response variable Churn.

One approach to address the imbalanced datasets is to oversample the minority class. The simplest approach involves duplicating examples in the minority class, although these examples do not add any new information to the model. Instead, new examples can be “synthesized” from the existing examples. This is a type of data augmentation for the minority class and is referred to as the *Synthetic Minority Oversampling Technique (SMOTE)*.

The SMOTE function is available in the *imblearn* library of Python. Below is the code that is used to “up-sample” minority class “1” by synthesizing from existing Churn samples.

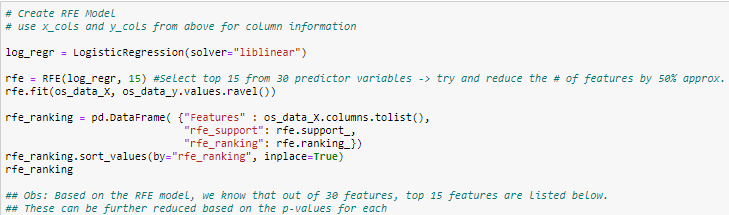


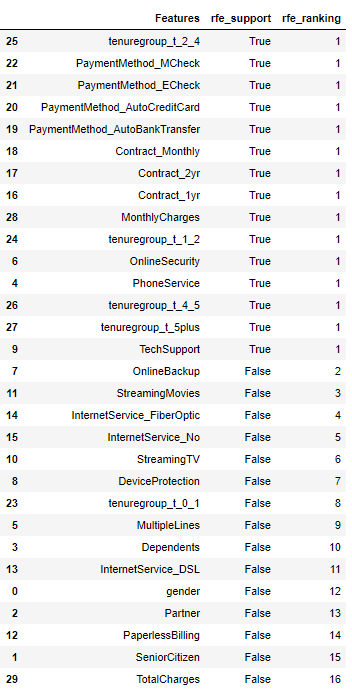
As it can be observed, the total number of records has been increased to 7752 from originally 7043 records. Also, the proportions between churned and retained customers are both at 50%. This new data set will be used for all the model building activities going forward.

## Dimension Reduction with Recursive Feature Elimination

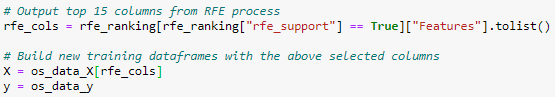
Not all features contribute to the prediction variable. Removing unimportant features can improve accuracy and reduce both, model complexity and overfitting. The training time can also be reduced for very large datasets. *Recursive Feature Elimination (RFE)*, as the title suggests, recursively removes features, builds a model using the remaining attributes, and calculates model accuracy. RFE can work out the combination of attributes that contribute to the prediction of the target variable (or class). Import RFE from *sklearn.feature\_selection* library and passing the Logistic Regression classifier model to the RFE() method with the number of features to select. In this case, the focus is on choosing the top 15 features out of 31 features from the training data set. Although the best practice is to run a *StratifiedKFold() RFE Cross-validation* to get the number of features that should be selected. For simplicity, the top 15 features will be selected.

The code that performs this activity, is listed below. The output of this step is also listed below.



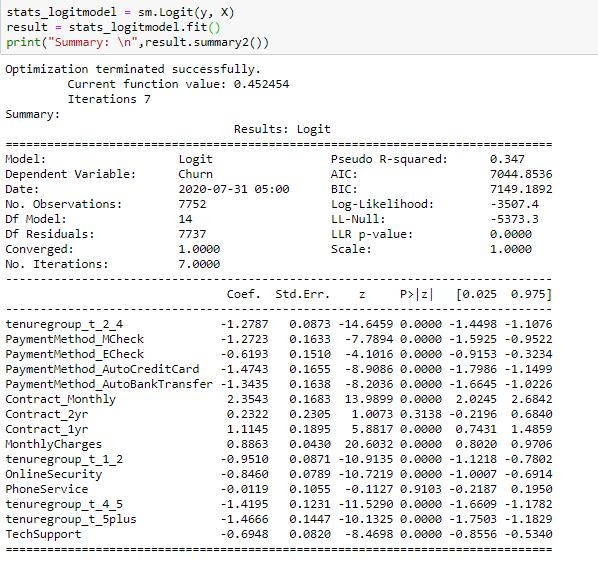


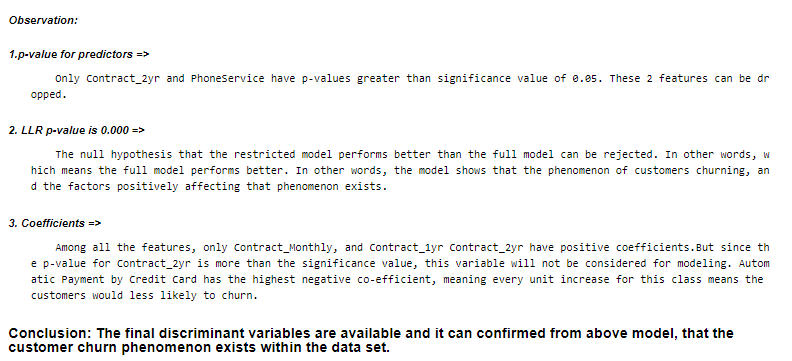
From the list above, the features marked with “*rfe\_support*” as *True* and “*rfe\_ranking*” as *1*, can be selected. The rest of the features can be dropped from the analysis. This is done by restricting the up-sampled training data to only these selected features, as shown below.



## Stats Model for Discriminant Analysis and Finding Phenomenon

A Logit statistic model can be created using the *statsmodel.api* library. The Logit function within the stats model in Python can handle both categorical and continuous predictor variables easily. This model will give additional insights into the most discriminating features, that are going to be included in the model. Also, it will provide additional details on whether the phenomenon exists or not. It is implemented as shown below.



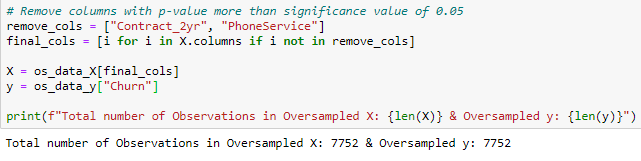


### Interpreting the parameters

1. Log-Likelihood – which is the maximized value of the log-likelihood function, is -3507.4
2. LL-Null – the result of maximized log-likelihood when only the intercept is included, is -5373.3
3. Pseudo-R2 – indicates the fit of the model, which is 0.347
4. LLR p-value – since it is 0.000, the null hypothesis that the restricted model performs better than the full model can be rejected, which means the full model performs better. In other words, the model shows that the phenomenon of customers churning, and the factors positively affecting that phenomenon exists.
5. p-values for features – every feature in the above has a p-value of 0.000, with an exception of *Contract\_2yr* and *PhoneService*. These two features can be dropped from the model.
6. Coefficients – Among all the features, only *Contract\_Monthly,* and *Contract\_1yr* *Contract\_2yr* have positive coefficients. This means, every unit increase in these classes, could mean customers in these classes would get progressively more likely to churn than other features, which have negative coefficients. It also shows that customers with automatic credit card payments have the highest negative coefficient. Meaning, every unit increase in this class could mean that customers paying their bills with a credit card automatically are most likely to stay. But since the p-value for *Contract\_2yr* is more than the significance value, this variable will not be considered for modeling.

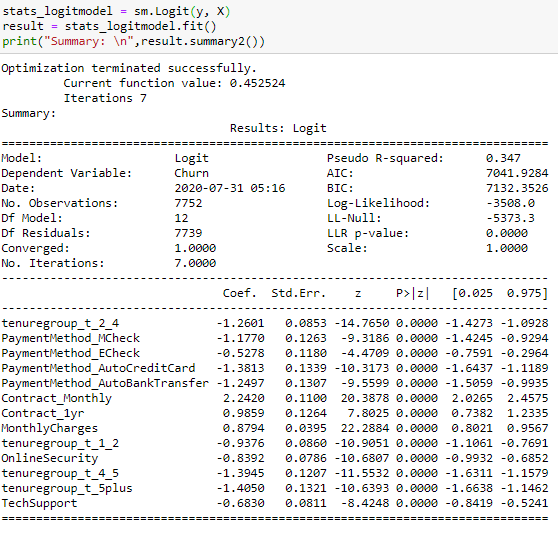
### Remove additional features

The above two features will have to be removed from the data set for future models. This is shown below.



### Validate Stats model

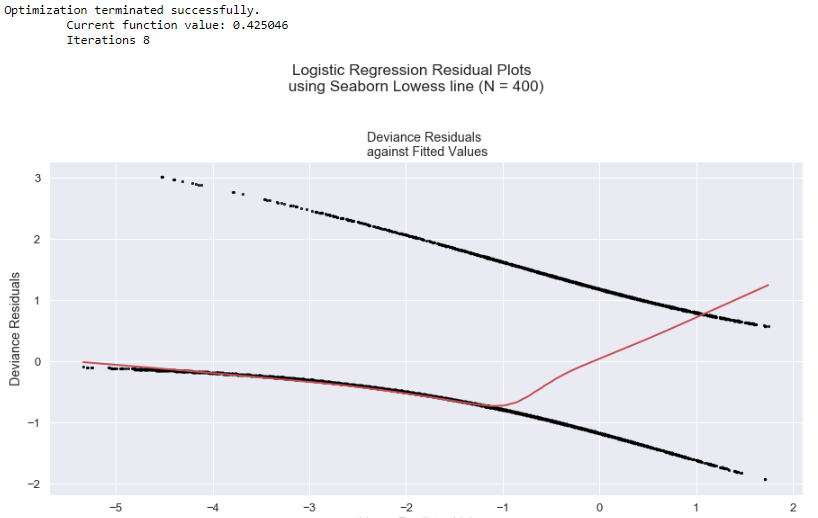
The stats model is created again to check whether any p-values for any other features were affected. It should be noted that p-values for no other feature were affected.

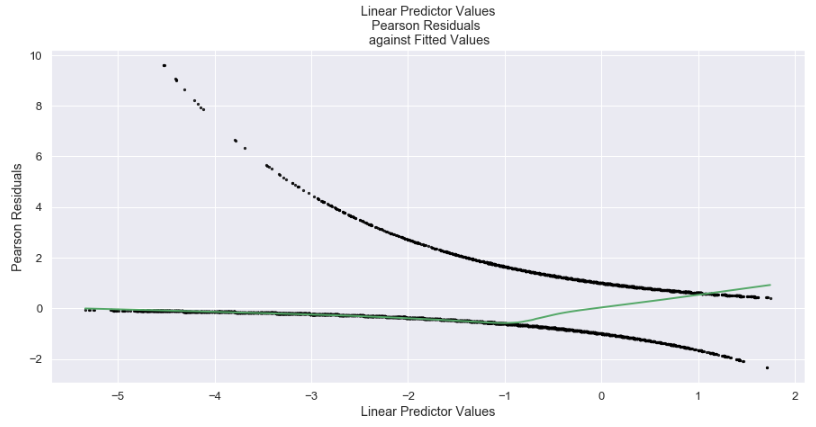


## Check the Logistic Regression Residuals Assumptions

In linear regression, one assesses the residuals as is. However, the residuals from the logistic regression model need to be transformed to be useful. This is because the dependent variable is binary (0 or 1). Due to the binary nature of the outcome, the residuals will not be normally distributed, and their distribution is unknown. The residuals assessed then are either the Pearson residuals, studentized Pearson residuals, and/or the deviance residuals (Data Science, *Python for Data Science*). Please note, this assumption was checked only for the selected features to find inadequacies in the model.



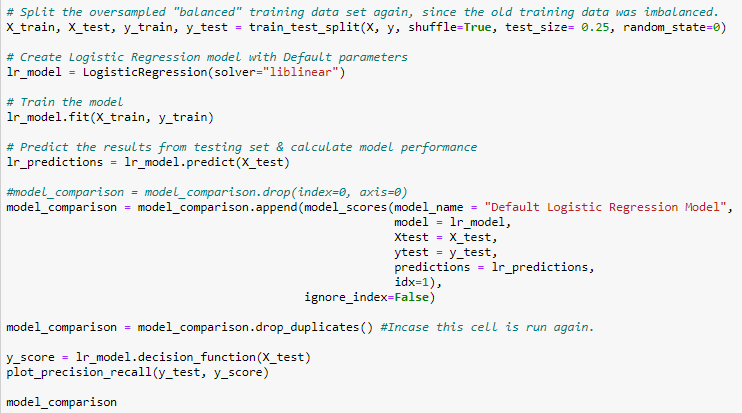




It appears the plots do an approximately horizontal line with 0-intercept. This suggests that there is no significant model inadequacy.

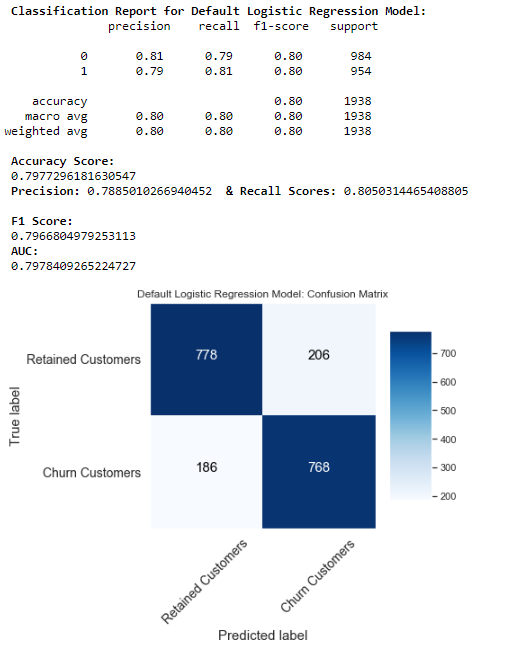
## Logistic Regression Model for Prediction – with Default Parameters

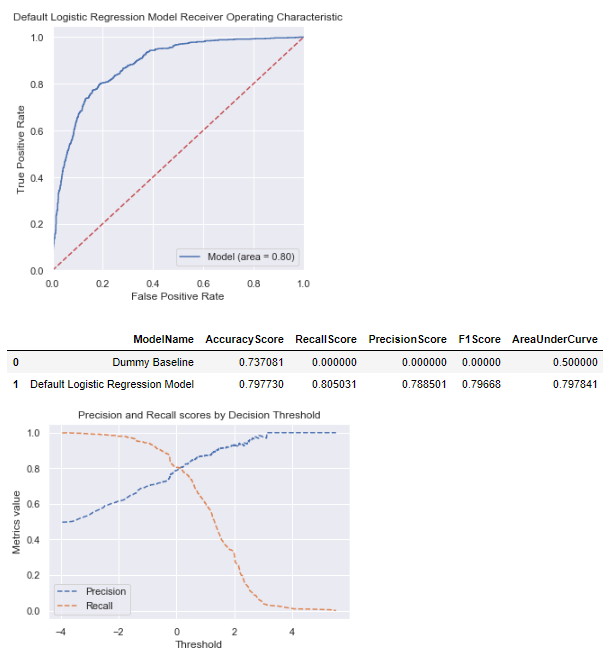
Once all the independent predictor variables have been decided, the assumptions for logistic regression satisfied, the predictive model can now be implemented. Since the training was synthesized to up-sample minority class, the train-test split needs to happen again.

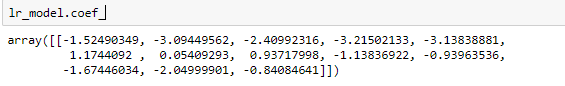


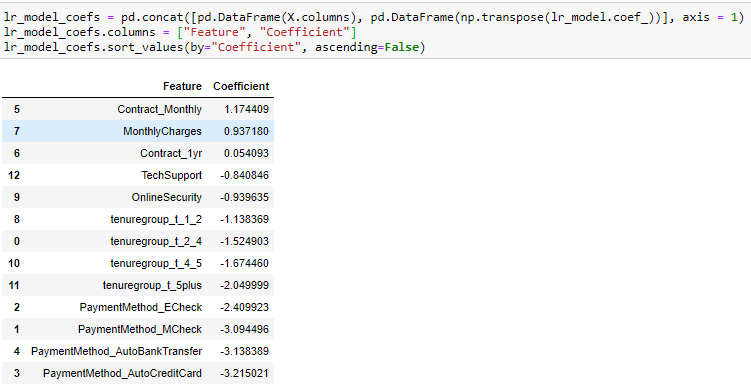
The above code creates a Logistic Regression model – “lr\_model”, with default parameters. The only parameter passed was the type of solver to use as a traditional “liblinear”. This solver was used specifically since the default “lbfgs” can sometimes cause a non-convergence error. The liblinear solver is much more predictable. The model is then fit or trained using the generated training set from the train-test split.

This model then predicts the outcome for the test set from the train-test split. The *model\_scores*, *plot\_precision\_recall* functions then calculate the scores, plots the required graphs, and appends the score for this model. The output scores of this model are below.



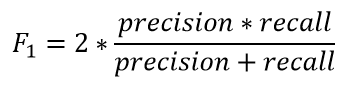






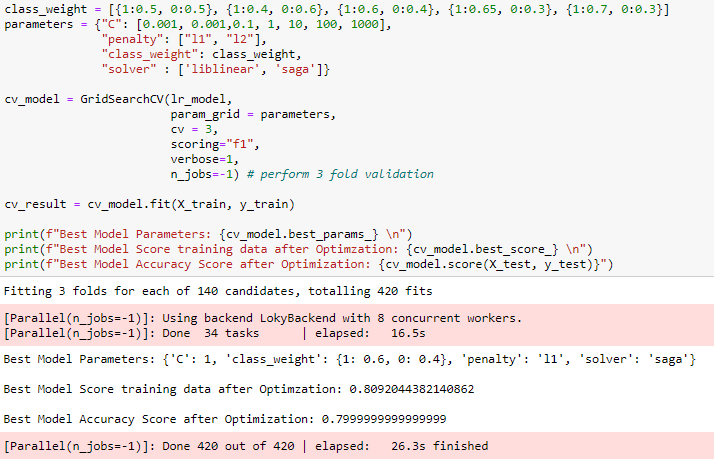
This is the list of coefficients that show the detection of the phenomena that needed to be found and prompted this analysis. Here we see what factors are most associated with customer churn and from there we can infer what is most likely causing customers to churn. As a 1 indicates a positive customer churn, the positive coefficients are the ones that push the probability closer to churn, and the negative coefficients push the probability closer to not churning. In addition to the above coefficients, the model accuracy score is 79.77%, whereas the recall score is at 80.50%.

Although accuracy is one of the most important scores among all different model scores, the focus can be changed to other scores depending on the problem that is being addressed. For example, a higher recall score means that the model tends to have a high number of true-positive and a smaller number of false-negatives, meaning finding the number of customers who are more likely to churn. Although this is a logical choice, it is a better practice to focus on finding a balance between precision score and recall score, so the rate of false positives does not increase. This is achieved by tuning the model at F1-Score, which is a harmonic mean of precision and recall scores (Koehrsen, *Beyond Accuracy: Precision and Recall*). Also, having a higher Area Under Curve (AUC) score means the model is better at predicting correct results. In this case, the AUC score is 0.80.

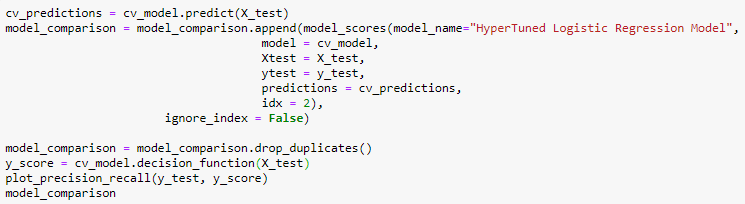


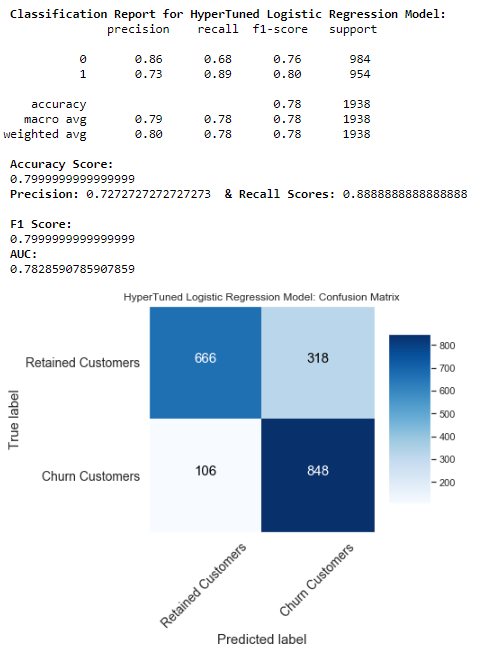
## Hyperparameter Optimization of Logistic Regression Model for Prediction

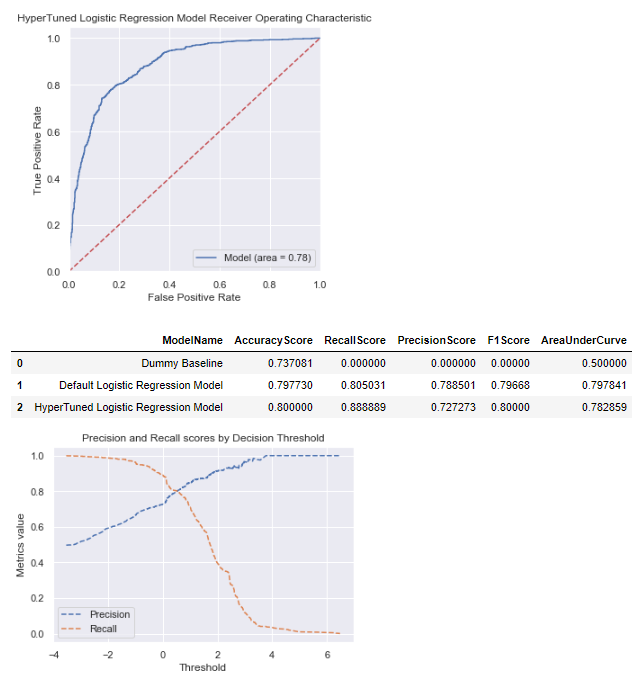
As discussed in the previous section, the model can be tuned to focus on specific scoring. The logistic regression model can be tuned on the model hyperparameters by the following code.



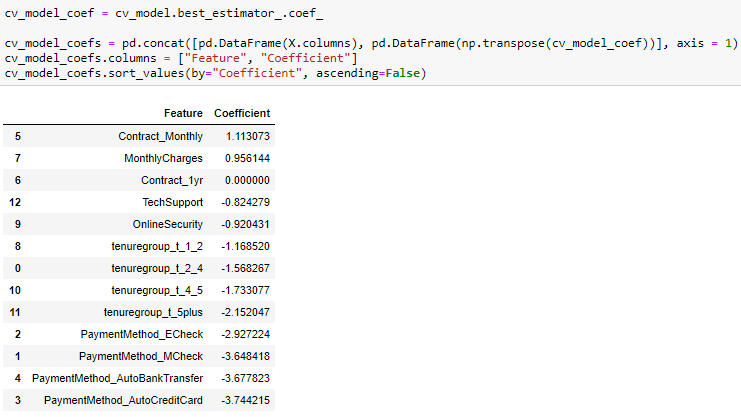
This process is performed by a “Cross-Validation” technique and progressively create different models, compare their scores and the *GridSearchCV()* function then returns the best model out of all the models generated. The output of this is as below.







It can be observed that, although there is not a lot of improvement in accuracy and F1 scores, the recall score is drastically improved. If the confusion matrices between the previous model and this model are compared, the number of False Negatives dropped, and the number of True Positives increased significantly. The coefficients after tuning the logistic regression model are below.



## Random Forest Model for Prediction – with Default Parameters

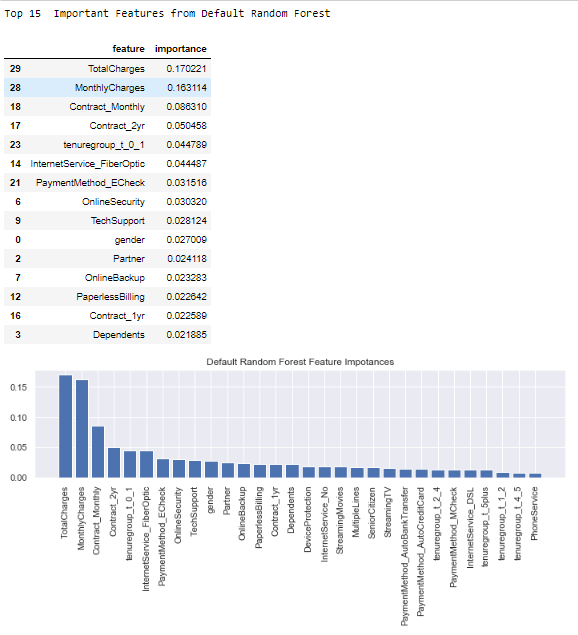
A decision tree is the building block of random forest and is an intuitive model. We can think of a decision tree as a series of yes/no questions asked about our data eventually leading to a predicted class. This is an interpretable model because it makes classifications much as we do: we ask a sequence of queries about the available data we have until we arrive at a decision (in an ideal world) (Koehrsen, *An Implementation, and Explanation of the Random Forest in Python*).

Although one can implement a single decision tree, it is very prone to overfitting. To avoid this problem, a Random Forest, which is a collection of decision trees, is chosen to predict the customer churn. Please note, this algorithm would be run on the entire set of features, since this algorithm can handle the high number of features (Koehrsen, *An Implementation, and Explanation of the Random Forest in Python*).



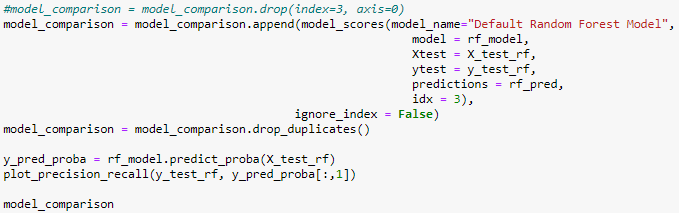


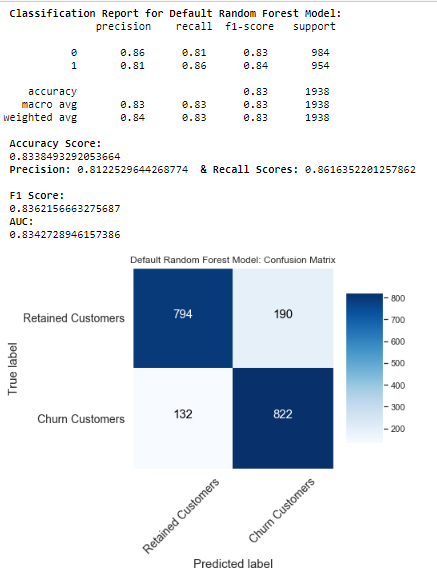
The above code performs the train-test split on the up-sampled synthesized training data set, creates a Random Forest model with default parameters. This will generate the predictions on the test data set and list out the top 15 features from the data set. The important features are displayed below.

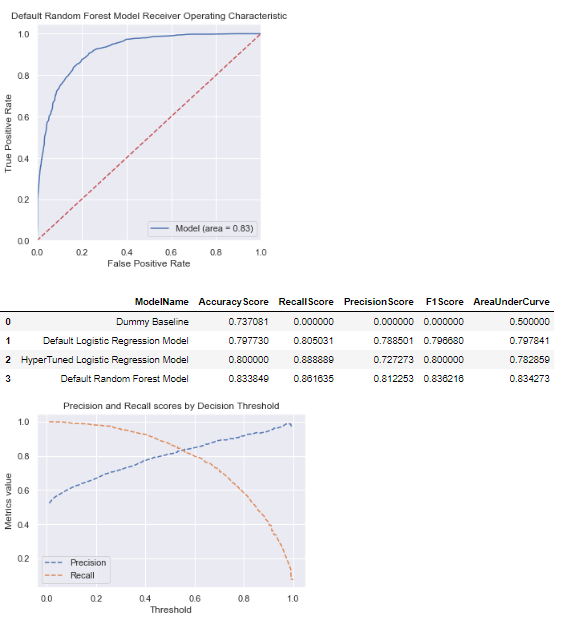


It can be noted that among the top 15 important features, *TotalCharges*, *MonthlyCharges*, *ContractMonthly*, *Contract\_2yr,* and *InternetService\_FiberOptic* are the most influential features on the probability of customer churn. This list can be cross-referenced with the features with positive coefficients list generated for the Logistic Regression, to come up with the top 5-6 features to recommend to Telco management to generate their mitigation plan, since including more number of features in their plan can get too complicated.

The model scores for the default Random Forest model are displayed below.



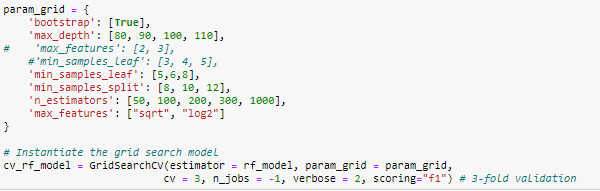


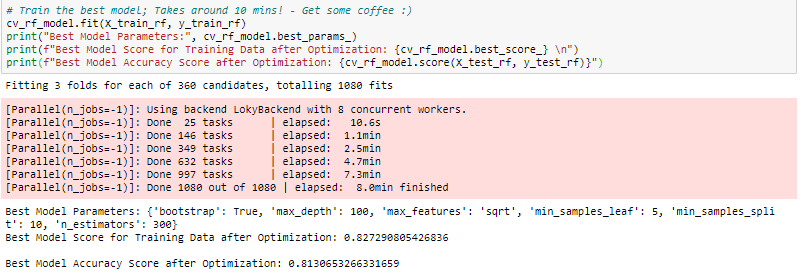


It can be noted that accuracy, precision, and F1 scores are improved compared to the hyper-tuned Logistic Regression model and the recall score was degraded slightly.

## Hyperparameter Optimization for Random Forest Model for Prediction

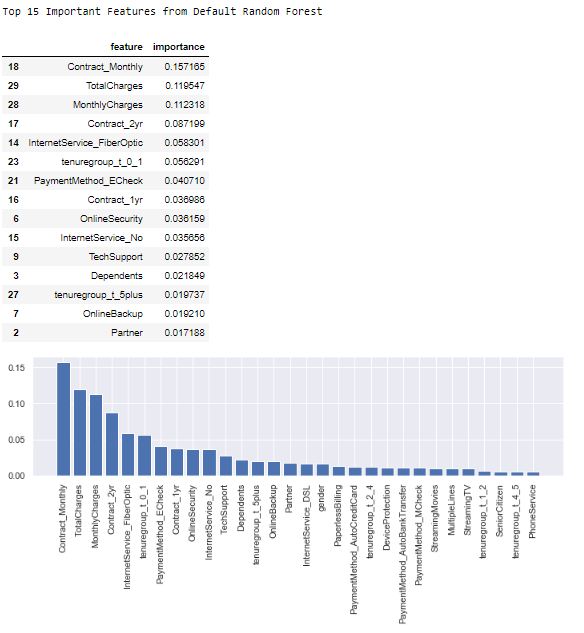
Like the Logistic Regression model, the Random Forest can also be tuned to address specific model scoring values. In this case, the random forest model is tuned to focus on the F1 score as well, so the comparison to tuned logistic regression model can be performed. This is performed by the following code.



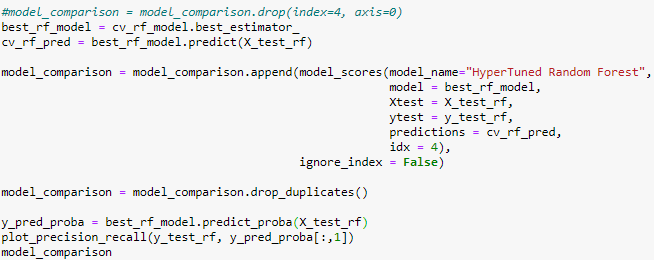


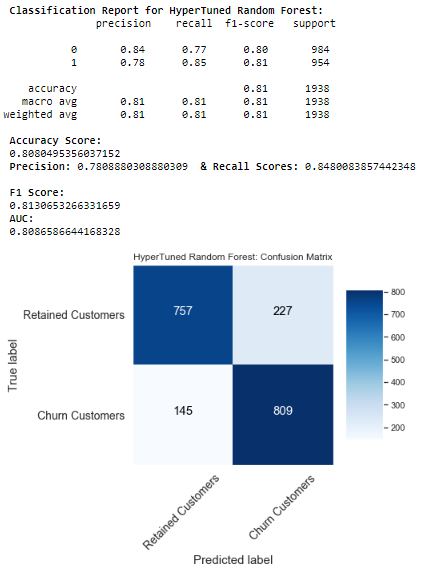
The above hyperparameter tuning leads to 1080 model fits. The output of the execution of *GridSearchCV()* is the random forest model with the best scores among the fitted models. The top 15 most important features from this model are below.

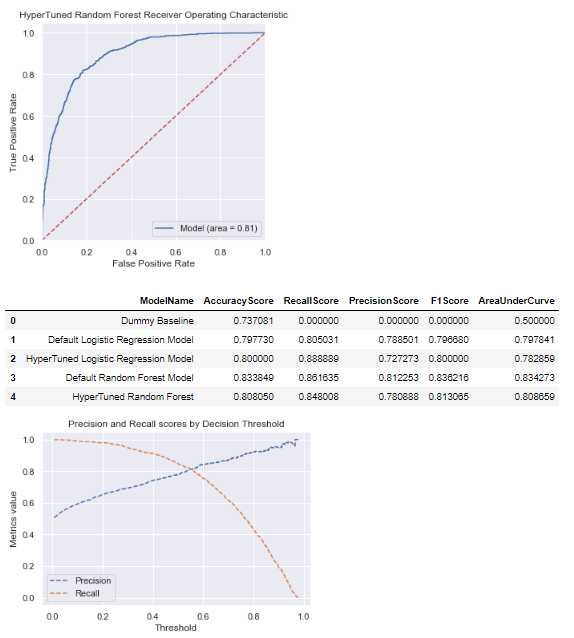




If the comparison to the earlier list is made, although the importance ranking of the top 5 among these 15 features has changed, the features remain the same. Now checking the scores of the tuned model are below.



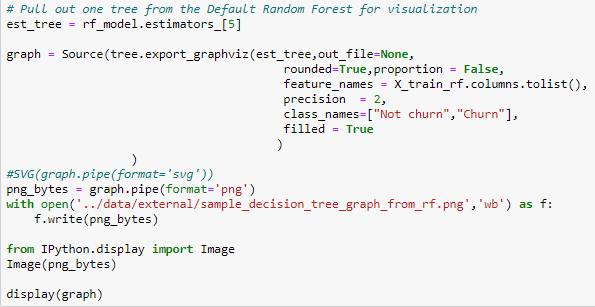


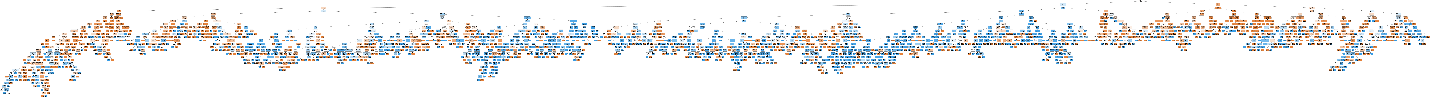


Surprisingly, all the scores have dropped as compared to the previous random forest model with default parameters.

## Visualizing Sample Decision Tree from Default Random Forest

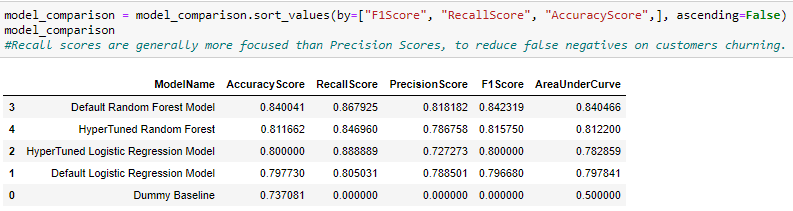
A sample decision tree can be “plucked” out of the random forest model to visualize the tree. The code to generate the decision tree visualization is below. Since it generates a very spread out tree graph, the full decision graph is saved in an external file under folder path data/external as “*sample\_decision\_tree\_graph\_from\_rf.png*”.



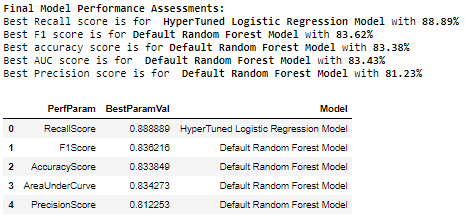


## Scores Comparison for Predictive Models

The model with the best performance must be selected before finalizing a model for predicting customer churn and making mitigation plans. Since the focus is on F1 Score and recall scores, the list is sorted by the F1 Score, Recall Score, and Accuracy Score.









The recommended model to use for prediction is Random Forest with default parameters, since that has the best scores. It predicted that 52.22% of customers from the test data are likely to churn and need to be handled accordingly.

## Top Features of Interest

Comparing the features with positive coefficients from both the hyper-tuned logistic regression models and the top 5 most important features from both the random forest models will result in finding the most important areas of focus for Telco management’s mitigation plan. This results in the below features:

1. Contract\_Monthly
2. MonthlyCharges
3. TotalCharges
4. Contract\_2yr
5. InternetService\_FiberOptic

In addition to these features from the predictive models, after checking the summary statistics of these features in earlier sections, several important observations can be included as well. These include:

1. Paperless Billing – 74.9% of churned customers have this option of invoice delivery.
2. Internet Service DSL – 69.4% of churned customers have a DSL Internet connection.
3. Payment Method Manual Check – 57.3% of churned customers pay bills by manual check
4. Tenure Group 0-1 Year – 55.5% of churned customers are new customers, which could be an alarming sign and needs to be looked into.
5. Multiple Lines – 45.5% of churned customers have multiple phone lines.
6. Streaming TV & Movies – 43.6% of churned customers like to stream live TV and movies.

## Final Output Files

The output files which are generated as a part of this analysis are:

1. Project Folder – *Customer\_Churn.zip*
2. Python Notebook - wgu\_telco\_customer\_churn\_analysis.ipynb – at Customer\_Churn/notebooks within project folder.
3. Cleansed and Transformed Data file - Telco\_Customer\_Churn\_cleaned.xlsx; Consists of two worksheets at Customer\_Churn/data/external
   1. Telco\_cleaned\_NoTransform – No Scaling or encoding
   2. Telco\_TransformedData – Scaled and Encoded data
4. Sample Decision Tree – sample\_decision\_tree\_graph\_from\_rf.png at Customer\_Churn/data/external and attached.
5. Full program run screenshots - Successful-Run-Screenshots-Telcom-Customer-Churn-Analysis.docx

# Summary and Conclusions

A combination of RFE and Logit Stats Model was used for detecting the interactions, selecting the most important and the most discriminating predictor variables. Also, the Random Forest feature importance scores were used to cross-reference with the above list of important features. These list *Contract\_Monthly* as the most influential predictor in terms of predicting customer churn, followed by *MonthlyCharges*. The Random Forest algorithm also listed out *TotalCharges* with the highest feature importance score, along with Contract\_2yr among the top 5 important features.

For example, the coefficient score for *Contract\_Monthly* from Stats model is 2.2420, Logistic Regression model: 1.174409, Hyper-tuned Logistic Regression model: 1.113073, Random Forest importance score: 0.164072 and Hyper-tuned Random Forest importance score: 0.143334. All these show that the customer churn phenomenon exists, and unit increases in this feature positively influence customer churn.

The Logit Stats model listed the p-values of the predictor variables and the ones which were not discriminant, the values for those variables were more than the significance value of 0.05. For example, *PhoneService* has a p-value of 0.9103 and thus was removed from the analysis.

Based on all the observations from the analysis, the biggest issue is with the customers with a month-to-month subscription, where 88.6% of the churned customers are month-to-month subscribers. Since these customers are on a month-to-month basis, the monthly charges that these customers are paying also plays an important role. The competitors could be providing better monthly rates for their services, which could be the reason for the high churn rate among this group of customers. To mitigate this problem, as a proactive strategy, they could be offered better rates and offer to sign a 1 or 2-year contract with the company, which could include discounted monthly charges for half of the total duration of their contract agreement.

Another issue is with customers with two-year contracts and high total charges. These customers could be towards the end of their contract term and if the customer feels that total accrued charges during their term with Telco are higher than what the competition is currently offering, they would be more influenced to churn from Telco. This can be a reactive mitigation strategy in which, if such customers request to terminate or do not wish to renew their contract, offering them competition price-match offers, discounts or a good balance of both would persuade them to stay or sign another 2-year contract with Telco.

The third set of problems is with the Internet Service department, where 69.4% of churned customers are using DSL type connections. In addition to this, the best performing prediction model ranks Fiber Optic internet service fourth most important factor in influencing the customer to churn. This business segment needs to be looked at for improving services and offerings.

Finally, customers enrolled in paperless billing, streaming live TV and movies services, multiple phone lines, along with ones paying the bills via manual checks and new customers with tenure less than a year, are also the groups of customers that deserve special attention.

Lastly, the Random Forest model can be used to predict customer churn at Telco. As a future enhancement to the prediction process, this model can be saved in a file and deployed through an API to be consumed for predicting individual cases and integrate with other enterprise applications.

# Acknowledgment

The raw data set was obtained from the Performance Assessment Task page from WGU Portal ("Performance Assessment - ACE3").

The document template for this report was provided by the Writing Center team from Student Success Centers, with additional details referred to APA style formatting from WGU Knowledge Center (WGU Knowledge Center, solene, "APA Document Formatting", 2018).

All the sources listed under the “References” section of this report were used to refer details and instructions to complete this analysis and report.

Finally, sincere gratitude to “David Gagner” and “Kelly Smith” for their exceptional guidance, support, and encouragement throughout this course.

# References

PERFORMANCE ASSESSMENT - ACE3. (n.d.). Retrieved from https://tasks.wgu.edu/student/001101075/course/11740007/task/970/overview

Brownlee, J. (2020, June 29). How to Use StandardScaler and MinMaxScaler Transforms in Python. Retrieved July 31, 2020, from https://machinelearningmastery.com/standardscaler-and-minmaxscaler-transforms-in-python/

Kumar, N. (2019, March). Advantages and Disadvantages of Random Forest Algorithm in Machine Learning. Retrieved July 31, 2020, from http://theprofessionalspoint.blogspot.com/2019/02/advantages-and-disadvantages-of-random.html

LLC, B. (2019, September 06). 2019 SAS, R, or Python Survey Update: Which Tool do Data Scientists &amp; Analytics Pros Prefer? Retrieved July 28, 2020, from https://www.burtchworks.com/2019/08/21/2019-sas-r-or-python-survey-update-which-tool-do-data-scientists-analytics-pros-prefer/

Srinidhi, S. (2020, January 09). Label Encoder vs. One Hot Encoder in Machine Learning. Retrieved July 31, 2020, from https://medium.com/@contactsunny/label-encoder-vs-one-hot-encoder-in-machine-learning-3fc273365621

Tuffery, S. (2011). Table 6.2 Predictive Methods. In Data Mining and Statistics for Decision Making (p. 170). West Sussex, England: John Wiley & Sons.

Data Science, P. (n.d.). Python for Data Science. Retrieved July 31, 2020, from https://www.pythonfordatascience.org/logistic-regression-python/

Koehrsen, W. (2018, March 10). Beyond Accuracy: Precision and Recall. Retrieved August 01, 2020, from https://towardsdatascience.com/beyond-accuracy-precision-and-recall-3da06bea9f6c

Koehrsen, W. (2018, August 31). An Implementation and Explanation of the Random Forest in Python. Retrieved August 01, 2020, from https://towardsdatascience.com/an-implementation-and-explanation-of-the-random-forest-in-python-77bf308a9b76