Part 1: Research Question

A1) Research Question

Using a list of explanatory variables can we predict what a customer’s monthly charge will be?

A2) Objectives and Goals

The goal of this analysis is to be able to use the prediction to determine which services have the biggest impact on the monthly charge and to see which services, if any, will cause the customer to churn based on a high monthly charge.

Part 2: Method Justification

B1) Assumptions of Multiple Regression Model

The assumptions of a multiple linear regression model include:

* That there is a linear relationship between the response variable and the explanatory variables
* That the explanatory variables are not too highly correlated with each other
* That the yi observations are selected independently and randomly from the population
* That the residuals should be normally distributed with a mean of zero

B2) Benefits of Tools

For this analysis I will be using Python for the coding and Jupyter notebook as the IDE.

While R is a powerful language for data analysis, Python is a more versatile language considering it is primarily an object-oriented language. For me personally, when starting out with programming I worked primarily with C# and Java, as I am sure many other people do. Using Python feels more natural coming from a background using other object-oriented languages, and Python has many different libraries and packages that are making it one of the strongest tools for data analysis. Jupyter notebook is also simple to use, as it runs right in the browser and has a clean user interface. [6]

B3) Appropriate Technique

Multiple linear regression is the appropriate technique to use for the analysis of my research question because the response variable I am predicting is numeric.

Part 3: Data Preparation

C1) Describe Data Goals

The goal of the data preparation is to manipulate the data to better fit using multiple linear regression.

To achieve this goal, first I will load the data set into python and evaluate the data to gain a better understanding of the data set. Normally the first part of manipulation would be to re-name the final 8 survey questions for better visibility, but since those will be dropped as less meaningful, I am going to skip that part and go right to dropping all the less meaningful columns from the data set. The next step is to check the data for any fields of missing or null values, should any be found, I will impute the missing records using either the mean, median, or mode depending on the type of data that specific variable is. Any column that has categorical variables of yes or no will have their values replaced by 0 and 1. Out of the remaining columns of categorical values, for ‘Contract’ and ‘InternetService’ I will use the pandas method get\_dummies to split those columns into 0’s and 1’s. The remaining columns with categorical values will be dropped, such as the column ‘State’. Finally, I will create some univariate and bivariate graphs before saving the prepared data set.

C2) Discuss Summary Statistics

When loaded into pandas, we see that the data set consists of 50 columns each with 10,000 records. For this analysis, my target variable is the numeric variable ‘MonthlyCharge’. As for the predictor variables, I dropped the less meaningful columns that likely would not have any effect on our response variable. These columns include less meaningful data and customer demographic data such as:

* CaseOrder
* Customer\_id
* Interaction, UID
* City, State, Country, Zip, Lat, Lng
* Population, Area type
* TimeZone
* Marital status
* Gender
* Email (number of emails sent to customer)
* Contacts (number of times customer contacted tech support)
* Techie (whether customer considers themselves technically inclined)
* TechSupport
* PaperlessBilling
* PaymentMethod
* All 8 survey questions

When looking through the data set, I found that it appeared to be cleaned with no missing data. Finally, all remaining categorical values with fields of ‘yes’ and ‘no’ were replaced by 1 and 0, respectively. The variable ‘Contract’ had 3 fields being ‘Month-to-month’, ‘One year’, and ‘Two Year’ while the variable ‘InternetService’ also had 3 fields being ‘None’, ‘DSL’, and ‘Fiber Optic’. Both will have dummy values created to split them since they are not in any specific order, then the original columns will be dropped. This leaves the remaining columns of:

* Children
* Age
* Income
* Churn
* Outage\_sec\_perweek
* Yearly\_equip\_failure
* Contract (split with dummy values)
* Port\_modem
* Tablet
* InternetService (split with dummy values)
* Phone
* Multiple
* OnlineSecurity
* OnlineBackup
* DeviceProtection
* StreamingTV
* StreamingMovies
* Tenure
* Bandwidth\_GB\_Year

C3) Data Preparation Steps

* Load the data set into pandas dataframe
* Examine the data set
* Drop less meaningful columns
* Search for missing or null values
* Impute missing fields with mean, median, or mode if necessary
* Replace categorical values with numeric values
* Create dummy values for the two columns above
* Create histograms and boxplots for univariate visualizations
* Create scatterplots for bivariate visualizations
* Extract the now prepared data set as ‘churn\_prepared.csv’

C3) Code

*#Import all packages*

*import pandas as pd*

*import numpy as np*

*from pandas import Series, DataFrame*

*import seaborn as sns*

*import matplotlib.pyplot as plt*

*%matplotlib inline*

*import statsmodels.api as sm*

*import statistics*

*from scipy import stats*

*import sklearn*

*from sklearn import preprocessing*

*from sklearn.linear\_model import LinearRegression*

*from sklearn.model\_selection import train\_test\_split*

*from sklearn import metrics*

*from sklearn.metrics import classification\_report*

*#Load the data set into Pandas*

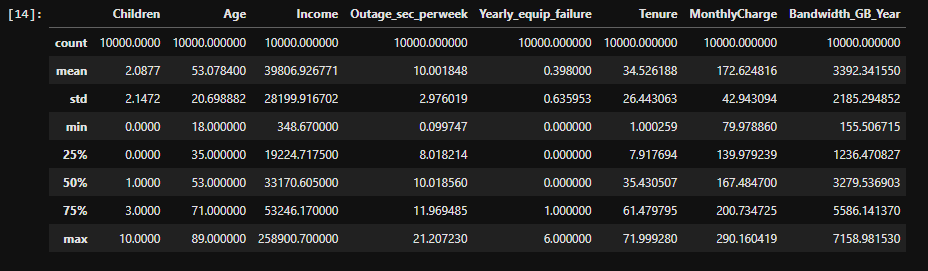
*df = pd.read\_csv('churn\_clean.csv', index\_col = 0)*

*df.describe()*

*#Drop the less meaningul columns from the data set*

*df = df.drop(columns = ['Customer\_id', 'Interaction', 'UID', 'City', 'State', 'County', 'Zip', 'Lat', 'Lng', 'Population', 'Area', 'TimeZone', 'Job', 'Marital', 'Gender', 'Email', 'Contacts', 'Techie', 'TechSupport', 'PaperlessBilling', 'PaymentMethod', 'Item1', 'Item2', 'Item3', 'Item4', 'Item5', 'Item6', 'Item7', 'Item8'])*

*df.describe()*

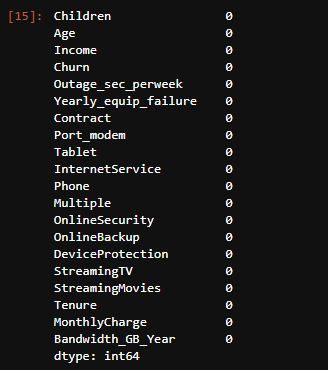
**

*#Drop the less meaningul columns from the data set*

*df = df.drop(columns = ['Customer\_id', 'Interaction', 'UID', 'City', 'State', 'County', 'Zip', 'Lat', 'Lng', 'Gender', 'Population', 'Area', 'TimeZone', 'Job', 'Marital', 'Email', 'Contacts', 'Techie', 'TechSupport', 'PaperlessBilling', 'PaymentMethod', 'Item1', 'Item2', 'Item3', 'Item4', 'Item5', 'Item6', 'Item7', 'Item8'])*

*#Search for missing data*

*df.isnull().sum()*

**

*#No missing data, now to use ordinal encoding to replace the categorical values with numeric ones*

*#Yes to 1, No to 0*

*df['Churn\_num'] = df['Churn']*

*df['Tablet\_num'] = df['Tablet']*

*df['Phone\_num'] = df['Phone']*

*df['Multiple\_num'] = df['Multiple']*

*df['OnlineSecurity\_num'] = df['OnlineSecurity']*

*df['OnlineBackup\_num'] = df['OnlineBackup']*

*df['DeviceProtection\_num'] = df['DeviceProtection']*

*df['StreamingTV\_num'] = df['StreamingTV']*

*df['StreamingMovies\_num'] = df['StreamingMovies']*

*df['Port\_modem\_num'] = df['Port\_modem']*

*#Set up dictionary for converting to numeric values*

*dict\_churn = {"Churn\_num" : {"Yes" : 1, "No" : 2}}*

*dict\_tablet = {"Tablet\_num" : {"Yes" : 1, "No" : 2}}*

*dict\_phone = {"Phone\_num" : {"Yes" : 1, "No" : 2}}*

*dict\_multiple = {"Multiple\_num" : {"Yes" : 1, "No" : 2}}*

*dict\_security = {"OnlineSecurity\_num" : {"Yes" : 1, "No" : 2}}*

*dict\_backup = {"OnlineBackup\_num" : {"Yes" : 1, "No" : 2}}*

*dict\_protection = {"DeviceProtection\_num" : {"Yes" : 1, "No" : 2}}*

*dict\_tv = {"StreamingTV\_num" : {"Yes" : 1, "No" : 2}}*

*dict\_movies = {"StreamingMovies\_num" : {"Yes" : 1, "No" : 2}}*

*dict\_modem = {"Port\_modem\_num" : {"Yes" : 1, "No" : 2}}*

*#Replace the variables values*

*df.replace(dict\_churn, inplace = True)*

*df.replace(dict\_tablet, inplace = True)*

*df.replace(dict\_phone, inplace = True)*

*df.replace(dict\_multiple, inplace = True)*

*df.replace(dict\_security, inplace = True)*

*df.replace(dict\_backup, inplace = True)*

*df.replace(dict\_protection, inplace = True)*

*df.replace(dict\_tv, inplace = True)*

*df.replace(dict\_movies, inplace = True)*

*df.replace(dict\_modem, inplace = True)*

*#Now that we have those as numeric, we can drop the original columns*

*df = df.drop(columns = ['Churn', 'Port\_modem', 'Tablet', 'Phone', 'Multiple', 'OnlineSecurity','OnlineBackup', 'DeviceProtection','StreamingTV', 'StreamingMovies'])*

*df.info()*

*Graphical user interface, text

Description automatically generated*

*#Now we use dummies and one-hot encoding for the categorical variables with n-levels*

*contract = pd.get\_dummies(df['Contract'], drop\_first = True)*

*internet = pd.get\_dummies(df['InternetService'], drop\_first = True)*

*df = df.join(contract)*

*df = df.join(internet)*

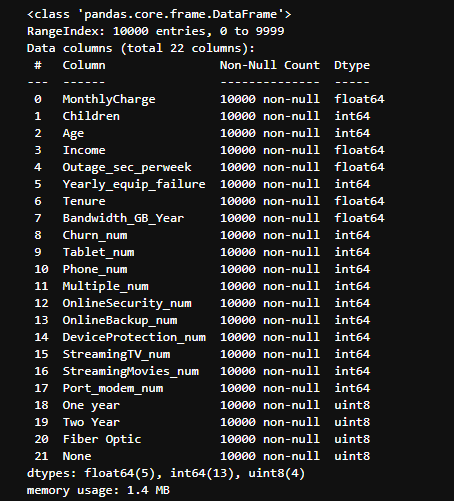
*#Drop the original columns*

*df = df.drop(columns = ['Contract', 'InternetService'])*

*#Move MonthlyCharge to top of the data set*

*df = df.set\_index('MonthlyCharge').reset\_index()*

*df.info()*

**

As we can see here, every variable is numeric and both ‘Contract’ and ‘InternetService’ have been split using pandas .get\_dummies() method with drop\_first = True so we can get the n-1 columns.

C4) Univariate and Bivariate Visualizations

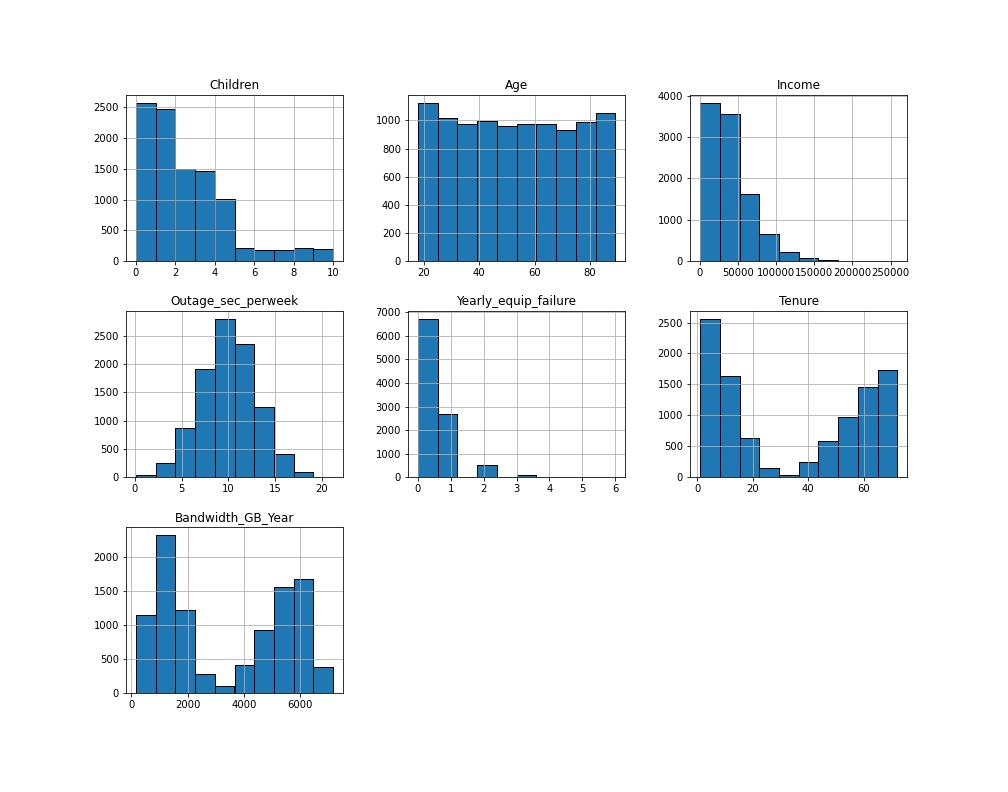
For the univariate visualizations I made histograms for the numeric variables, along with a couple of boxplots. For the bivariate visualizations I made scatterplots of several variables with my target variable, MonthlyCharge, on the y-axis of each scatterplot.

Also, for the bivariate visualizations I decided to make a subset of a random sampling of 25% of the data set so the plots have better visualizations.

C4) Code

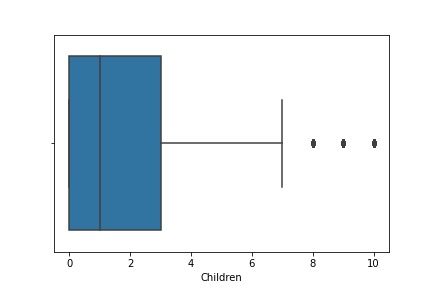
*#For univariate statistics, create histograms for the continuous and categorical variables*

*df[['Children', 'Age', 'Income', 'Outage\_sec\_perweek', 'Yearly\_equip\_failure', 'Tenure', 'Bandwidth\_GB\_Year']].hist(ec = "black", figsize = (14, 11))*

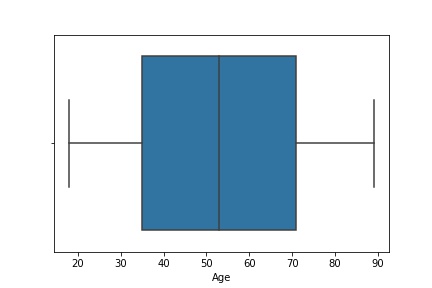
**

*#Create a couple boxplots*

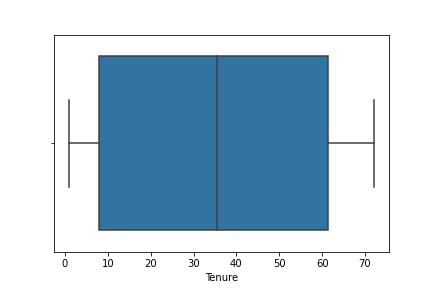
*sns.boxplot(x = df["Children"])*

**

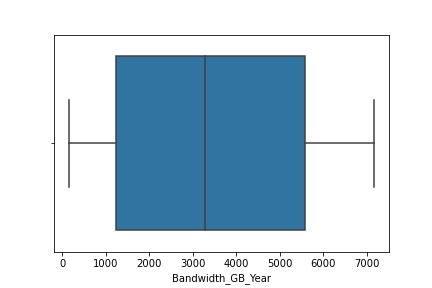
*sns.boxplot(x = df["Age"])*

**

*sns.boxplot(x = df["Tenure"])*

**

*sns.boxplot(x = df["Bandwidth\_GB\_Year"])*

**

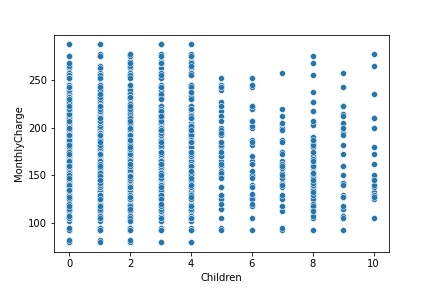
*#For bivariate statistics create some scatterplots with a few variables with our response variable as the y-axis*

*#First, creating a random sampling of 25% of the data set for the scatterplots to improve visibility[3]*

*subset = df.sample(frac = 0.25)*

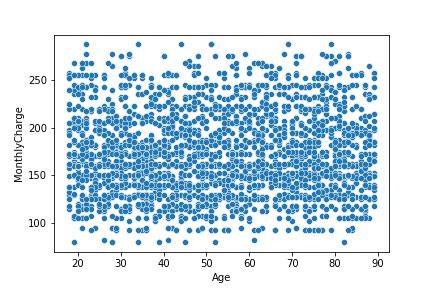
*sns.scatterplot(data = subset, x = "Children", y = "MonthlyCharge")*

*plt.show()*

**

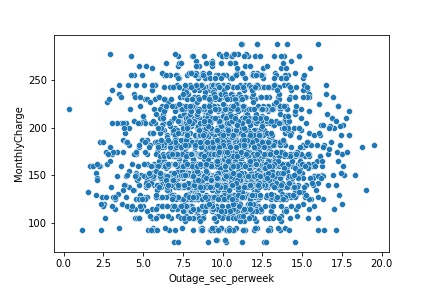
*sns.scatterplot(data = subset, x = "Age", y = "MonthlyCharge")*

*plt.show()*

**

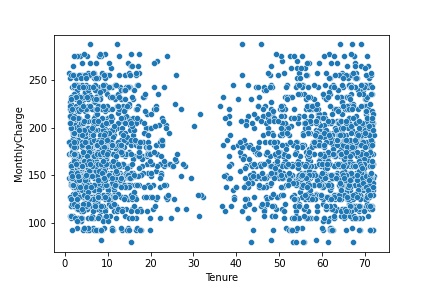
*sns.scatterplot(data = subset, x = "Outage\_sec\_perweek", y = "MonthlyCharge")*

*plt.show()*

**

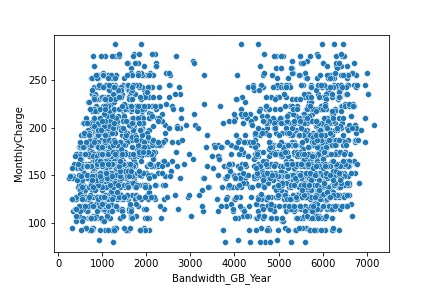
*sns.scatterplot(data = subset, x = "Tenure", y = "MonthlyCharge")*

*plt.show()*

**

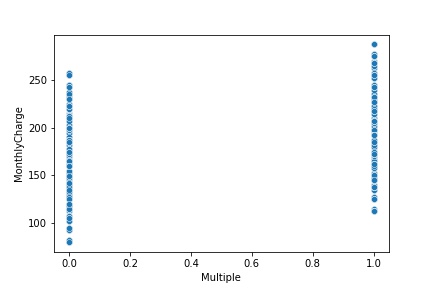
*sns.scatterplot(data = subset, x = "Bandwidth\_GB\_Year", y = "MonthlyCharge")*

*plt.show()*

**

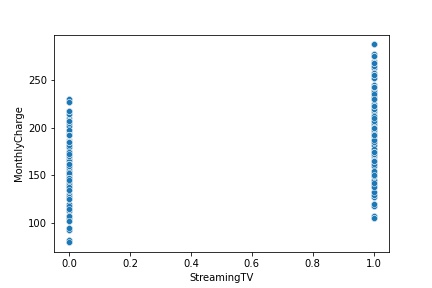
*sns.scatterplot(data = subset, x = "Multiple\_num", y = "MonthlyCharge")*

*plt.show()*

**

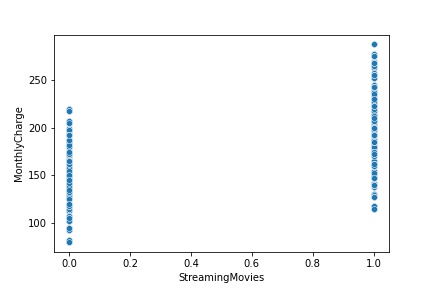
*sns.scatterplot(data = subset, x = "StreamingTV\_num", y = "MonthlyCharge")*

*plt.show()*

**

*sns.scatterplot(data = subset, x = "StreamingMovies\_num", y = "MonthlyCharge")*

*plt.show()*

**

C5) Provide Copy of the Prepared Data Set

*#Extract prepared dataset*

*df.to\_csv('churn\_prepared.csv')*

Part 4: Model Comparison and Analysis

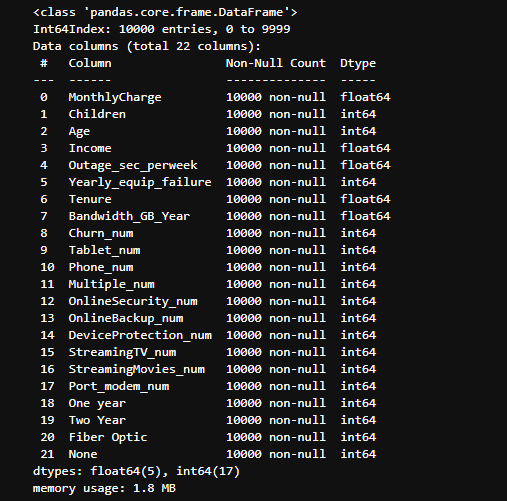
D1) Construct Initial Multiple Linear Regression Model

*#Load the data set into Pandas*

*df = pd.read\_csv('churn\_prepared.csv')*

*df.describe()*

*df.info()*

**

*#Prepare data [5][6]*

*feature = ['Children', 'Age', 'Income', 'Outage\_sec\_perweek', 'Yearly\_equip\_failure', 'Tenure', 'Bandwidth\_GB\_Year', 'Churn\_num', 'Tablet\_num', 'Phone\_num', 'Multiple\_num', 'OnlineSecurity\_num', 'OnlineBackup\_num', 'DeviceProtection\_num', 'StreamingTV\_num', 'StreamingMovies\_num', 'Port\_modem\_num', 'One year', 'Two Year', 'Fiber Optic', 'None']*

*X = df[feature]*

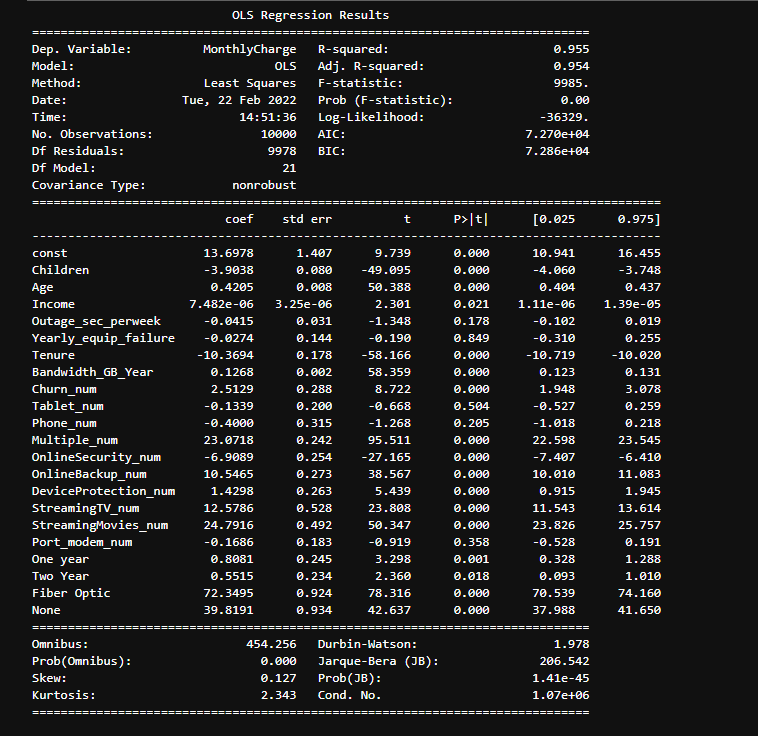
*y = df['MonthlyCharge']*

*X = sm.add\_constant(X)*

*#Create model [6]*

*model = sm.OLS(y, X).fit()*

*print(model.summary())*

**

D2) Selection Procedure

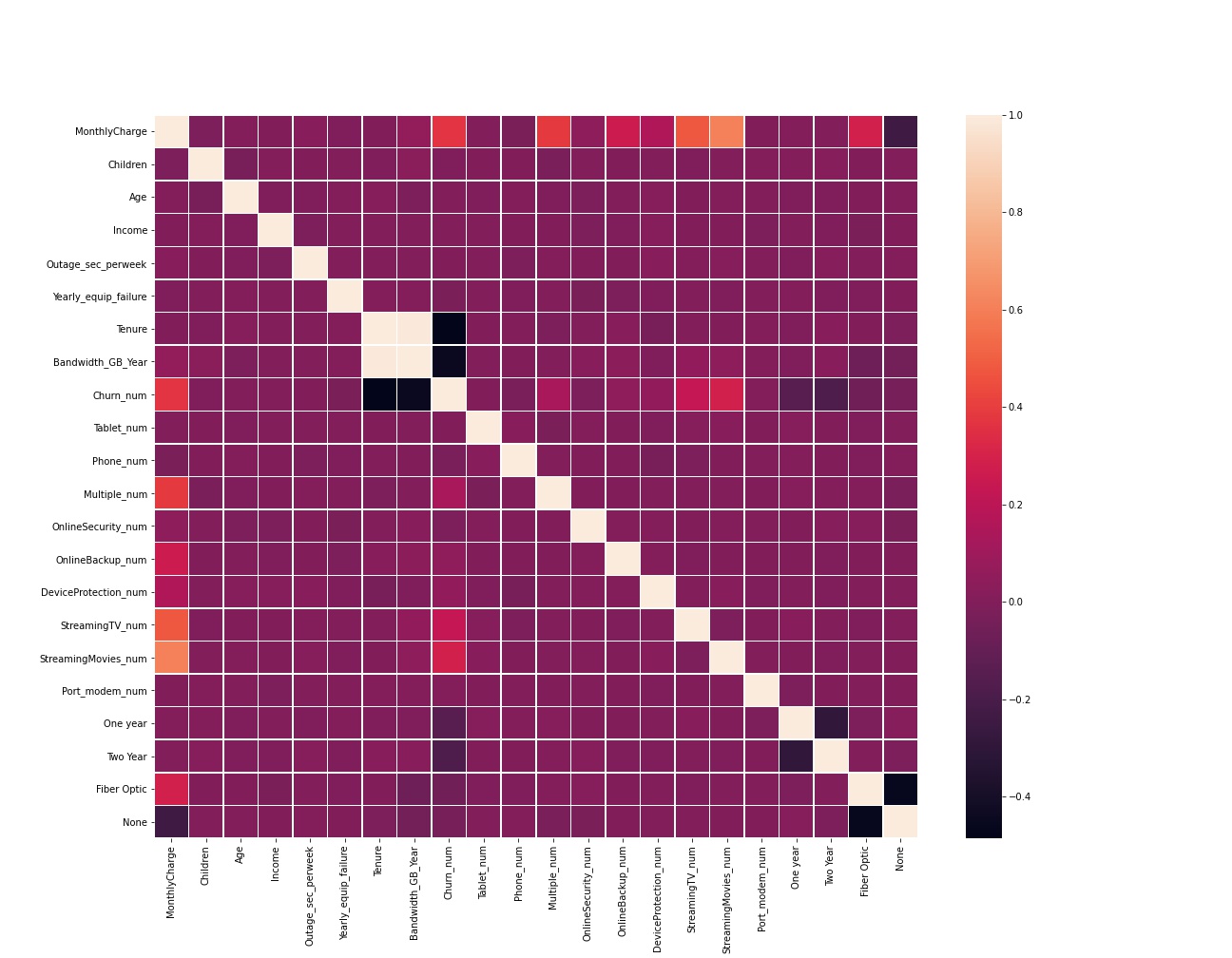
For the initial variable selection, we can look at the summary and disregard the variables with p-values above 0.05. We can also create a heatmap for the correlations between the variables to see which are more significant for our target variable.

*#For variable selection and model evaluation, use a seaborn heatmap to see correlation between the columns [1][2]*

*plt.figure(figsize = (18, 14))*

*sns.heatmap(df.corr(), linewidth = 0.5)*

*#plt.savefig('heatmap3.jpg')*



By looking at the heatmap we can see that Churn, Multiple, OnlineBackup, DeviceProtection, StreamingTV, StreamingMovies, Fiber Optic, and None (last two of which were dummies created from InternetService) are the variables that have the strongest effect on our target variable of MonthlyCharge.

D3) Reduced Regression Model

*#Re-prepare the data [5][6]*

*#Variable selection based on p-values and heatmap*

*Feature2 = ['Churn\_num', 'Multiple\_num', 'OnlineBackup\_num', 'DeviceProtection\_num', 'StreamingTV\_num', 'StreamingMovies\_num', 'Fiber Optic', 'None']*

*X2 = df[feature2]*

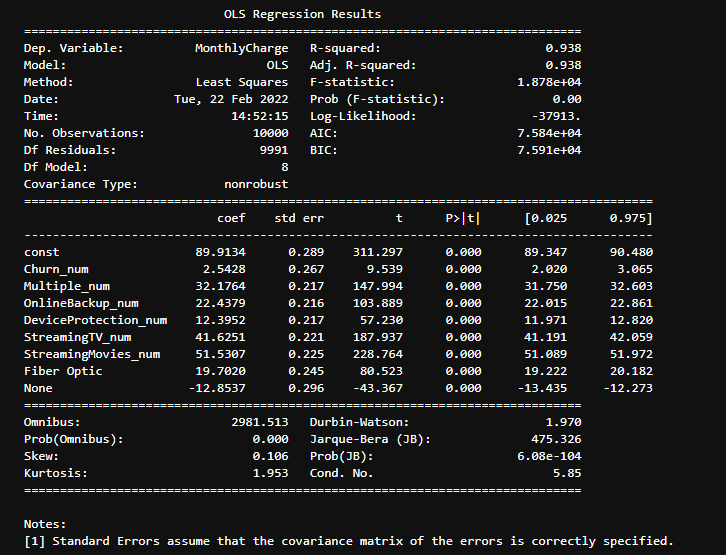
*y = df['MonthlyCharge']*

*X2 = sm.add\_constant(X2)*

*#Run reduced model [6]*

*reduced\_model = sm.OLS(y, X2).fit()*

*print(reduced\_model.summary())*

**

For the model evaluation we can look at the R^2 value from the initial model, being 0.955 and the adjusted R^2 value is 0.954. With both so close to each other and so close to 1, we can determine that the initial model is a pretty good model. Comparing it to the R^2 value of the reduced model, which is 0.938 we can see that the coefficient of determination went down a bit for the reduced model, even though the R^2 and adjusted R^2 value for the reduced model is the same. For the reasoning why it went down, I can only make an estimation at possible the fact that when ‘InternetService’ was split with .get\_dummies() and the first value dropped, perhaps that first value which was ‘DSL’ might have had a positive effect on the model.

E1) Process

For variable selection I chose to omit the variables with p-values higher than 0.05 since they are not statistically significant and used a heatmap to visualize the variables with the strongest correlation to my target variable. For the model evaluation I looked at the R^2 and adjusted R^2 values of each summary statistic to see how the reduced model held up to the initial one.

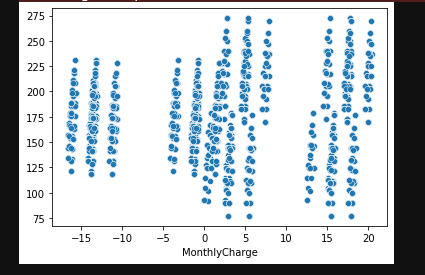
E1) Residual Plot

*y\_pred = reduced\_model.predict(X2)*

*residuals = y - y\_pred*

*sns.scatterplot(residuals, y\_pred)*

*plt.show()*



E2) Output

All output is listed above

E3) Code

All code is listed above

Part 5: Data Summary and Implications

F1) Discuss Results

The regression equation for the reduced model is as follows:

Y = 89.91 + 2.54(Churn) + 32.18(Multiple\_num) + 22.44(OnlineBackup\_num) + 12.40(DeviceProtection\_num) + 41.63(StreamingTV\_num) + 51.53(StreamingMovies\_num) + 19.70(Fiber Optic) – 12.85(None)

For this specific reduced model, each of the coefficients is negative, so this would suggest that for every 1 unit of:

Churn – MonthlyCharge will increase by 2.54

Multiple Devices – MonthlyCharge will increase by 32.18

OnlineBackup – MonthlyCharge will increase by 22.44

DeviceProtection – MonthlyCharge will increase by 12.40

StreamingTV – MonthlyCharge will increase by 41.63

StreamingMovies – MonthlyCharge will increase by 51.53

Fiber Optic (InternetService) – MonthlyCharge will increase by 19.70

None (No internet service) – Monthly charge will decrease by 12.85

The main limitation of this data analysis is that 10,000 records is small for something as large as media entertainment. As well as the fact that a variable like how much a customer will be charged is always fluid and changes due to many different factors.

F2) Recommendation

Using multiple linear regression to predict a variable like monthly charge probably would not be the best idea. In the analysis, all the variables that had the largest effect on the MonthlyCharge variable were services and addons, which almost always have set prices for most companies.

Part 6: Demonstration

G) Panopto Video

<https://wgu.hosted.panopto.com/Panopto/Pages/Viewer.aspx?id=a01de8b1-928f-41cf-93fe-ae44015fa3f0>

H) Sources for Third-Party Code

[1] “Seaborn.heatmap.” *Seaborn.heatmap - Seaborn 0.11.2 Documentation*, https://seaborn.pydata.org/generated/seaborn.heatmap.html.

[2] Carvalho, Thiago. “Heatmap Basics with Python's Seaborn.” *Medium*, Towards Data Science, 24 Sept. 2021, https://towardsdatascience.com/heatmap-basics-with-pythons-seaborn-fb92ea280a6c.

[3] Duca, Angelica Lo. “How to Sample a Dataframe in Python Pandas.” *Medium*, Towards Data Science, 7 July 2021, https://towardsdatascience.com/how-to-sample-a-dataframe-in-python-pandas-d18a3187139b.

[4] Navlani, Avinash. “Python Logistic Regression with Sklearn & Scikit.” *DataCamp Community*, <https://www.datacamp.com/community/tutorials/understanding-logistic-regression-python>.

[5] Larose, C. D., & Larose, D. T. (2019). Chapter 11: Regression Modeling. In *Data Science using python and R*. essay, Wiley Blackwell.

J) Sources

[6] “Python vs. R: What's the Difference?” *IBM*, https://www.ibm.com/cloud/blog/python-vs-r