D208_Performance_Assessment

Multiple Regression Analysis

for Predictive Modeling

September 22nd,2021

# Table of Contents

Performance Evaluation of Predictive Modeling – NBM2, D208 ........ 3
A. Introduction to Predictive Modeling -NBM2 ........ 3
A1. Analytical Question: ........ 3
A2. Goals and Objectives: ........ 3
B. Justification for the method ........ 3
B1. Assumptions Summary: Multi Linear Regression ........ 3
B2. Advantages/Benefit of the Tool ........ 4
B3. Appropriate Methodology: ........ 5
C. Data Objectives: ........ 5
C1. The following will be part of my strategy: ........ 5
C2. Statistics in Brief: ........ 6
C3. Data Preparation Procedures: ........ 7
C4: Imagination/Visualization ........ 15
C5: Prepared Data set ........ 25
D. Analysis and Comparison of Models ........ 26
D1. Initial Model of regression from all predictors: ........ 26
D2. Using model with all categorical dummy variables: ........ 27
D3: Model Reduction Justification: ........ 30
E. Perform the following for reduced multiple regression model: ........ 33
E3. Code ........ 33
F. Do the following to summarize your findings and assumptions: ........ 34
F1. Conclusions ........ 34
F2. Suggestions ........ 34
G. Documentary Evidence ........ 34
G1. Panopto recording: ........ 34
G2. Third Party Evidence: ........ 35
G3. References: ........ 35

# Performance Evaluation of Predictive Modeling – NBM2, D208

Name: Rekha Alle
Student ID: 000778673
Course: Masters Data Analytics
Date: 09/22/2021
Program Mentor: David Gagner
Contact: 3854282643
Email: David.gagner@wgu.edu

## A. Introduction to Predictive Modeling -NBM2

Finding reasons for client loss, gauging customer loyalty, and recovering customers have all become highly essential ideas for many businesses. Companies conduct a variety of studies and efforts to prevent losing consumers rather than gaining new ones.

Due to fast renewable technology, an increase in the number of users, and value-added services, the telecommunications business collects massive amounts of data. Due to the uncontrolled and rapid expansion of this area, considerable losses are incurred because of fraud and technical challenges. As a result, the creation of new analysis methodologies has become a necessity. The number one business goal for many providers is to keep extremely profitable customers. Telecommunications businesses must predict which consumers are at high risk of churning to reduce customer churn.

### A1. Analytical Question:

How many GBs of data will a consumer consume per year? Is it possible to forecast this accurately using a set of explanatory variables?

### A2. Goals and Objectives:

Everybody in the organization will profit if they can forecast how much data a consumer will utilize with reasonable certainty. This will help analyze the pros and cons of expanding client data limitations, providing limitless (or metered) media streaming, and expanding firm cloud computing capabilities to meet rising bandwidth demands.

## B. Justification for the method

### B1. Assumptions Summary: Multi Linear Regression

The result variable and the independent factors must have a linear relationship. Scatterplots can reveal whether a relationship is linear or curvilinear. A curvilinear relationship (left) and a linear relationship (right) are depicted in the following two instances below

**Multi-Variate Normality**: The residuals in multiple regression are assumed to be regularly distributed.

**No multi-collinearity:** The independent variables in multiple regression are assumed to be unrelated to one another. The variance inflation factor (VIF) measurements are used to test this hypothesis.

**Multi-linear regression**: It necessitates the presence of two independent variables, which are nominal, ordinal, or interval/rational in nature. Regression analysis requires at least 20 examples per independent variable in the analysis, according to a rule of thumb.

## B2. Advantages/Benefit of the Tool

**Tools will be used:**

For this assessment, I'll use Python because the study will be supported by Jupyter notebooks in Python and I Python. Python includes many established data science and machine learning tools, , straightforward, and extensible programming style, and grammar. Python is cross-platform, so it will function whether the analysis is viewed on a Windows PC or a MacBook laptop. When compared to other programming languages such as R or MATLAB, it is quick (Massaron, p. 8[1]). In addition, Python is often regarded in popular media as the most widely used programming language for data science and media (CBTNuggets, p. 1). [2]

**NumPy** used to work with arrays,
**Pandas** used to load datasets,
**Matplotlib** used to plot charts,
**Scikit-learn** used for machine learning model classes,
**SciPy** used for mathematical problems, specifically linear algebra transformations, and
**Seaborn** used for a high-level interface and appealing visualizations.
Using the Pandas library and its accompanying "read csv" function to transform our data as a dataframe is a quick, exact example of loading a dataset and constructing a variable efficiently: imported pandas as pd, df(dataframe) = pd.read csv('ChurnData.csv')

## B3. Appropriate Methodology:

In this assessment, we'll begin by looking at the data and deciding on a target and independent variable. We'll next use univariant and bivariant statistics to investigate this variable. We'll also look for outliers and missing numbers, as well as update variable types for future analysis. We will conduct multiple linear regression when the data has been cleansed.

Because our objective variable, forecasting (how much data is used) a continuous variable that represents an actual quantity of GBs each year, multiple regression is an effective technique for analyzing the study topic. Also, when attempting to anticipate how much data a client will use each year, there may be multiple (rather than just one) explanatory variables (area type, employment, income, age, children, etc.) this will help us to understand more. When we add or remove independent variables from our regression equation, we'll see its a positive or negative association with our target variable, and how that can affect marketing segmentation decisions within the organization.

# C. Data Objectives:

## C1. The following will be part of my strategy:

1. Using Pandas' read csv command, read the data collection into python programming.
2. Examine the data structure for a better understanding of the data collection process.
3. Using the variable "churn df" to name the dataset, and "df" to name the data frame's subsequent usable slices.
4. Check for misspellings, strange variable names, and data that is missing.
5. Identify outliers that may create or obscure statistical significance using histograms.
6. Computing replaces missing data with relevant central tendency measures (mean, median, or mode) or just Outliers a few standard deviations above the mean are removed.

Using "Bandwidth GB Year" (the average) is a dependent variable of yearly quantity of data consumed, per customer, in GB), it will be our long- term target variable, is the most important to our decision-making process. To construct a model that will give us an indication of data a customer may use given the quantities utilized by known customers given their individual data points for selected predictor variables, on our given dataset, we must first train and then test our machine.

The categorical variables (all binary Predictor except for categorical variable with two values, "Yes"/ "No," were stated) may be shown to be significant: * Churn: Whether the consumer stopped using the service in the previous month (yes, no)* Techie: Whether or not the customer perceives themselves to be technically savvy (as determined by a customer questionnaire completed * Contract (at the time of signing up for services) (yes, no):The customer's contract term (one year , two years or month-on-month). * Port_modem: consumer using a portable modem (yes/no) * the consumer possesses a tablet, such as or a Surface or an iPad (yes, no) * Internet Service: The internet service provider for the customer (DSL, fiber optic, None) * Phone: Is there a phone service for the consumer (yes, no)? * Multiple: If the customer has more than one line (yes, no) * Online Security: Whether the consumer has an add-on for online security (yes, no) * Online Backup: Whether the consumer has purchased an add-on for internet backup (yes, no) * Device Protection: Is any consumer device protection add-on? (Yes, no) * Tech Support: Is there a technical assistance add-on for the customer (yes, no) * Streaming TV: Whether the consumer has access to streaming television (yes, no) * Streaming Movies: If the customer has access to on-demand movies (yes, no).

In the decisionmaking process, discrete ordinal predictor variables created from consumer surve y responses about various customer service attributes could be valuable. Customers in the surve ys provided ordinal numerical data by rating eight customer service aspects on a scale of 8 to 1 (8 being the most essential and 1 being the least important):

- Item1: Evidence of active listening
- Item2: Courteous exchange
- Item3: Respectful response
- Item4: Options
- Item5: Reliability
- Item6: Timely replacements
- Item7: Timely fixes
- Item8: Timely response

## C2. Statistics in Brief:

The dataset has 50 original columns and 10,000 records, as shown in the Python pandas data frame techniques are as follows.

Especial user IDs and statics categorical variables (‘Customer id’,'Case Order, 'Interaction, 'City,'State,'County,'Zip,'Lat,'UID, 'Area,'Lng’,'PaymentMethod','Population, 'TimeZone,'Job,'Marital) not included in the data frame for this research. In addition, binomial "Yes"or "No" / "Male"or"Female" variables encoded to 1 or 0. This left 34 numerical independent predictor factors, including the target variable, to be determined. There appeared to be no nulls, NAs, or missing data points in the dataset, indicating that it had been well cleaned. Ordinary distributions were discovered for "Outage sec per week,” Email" and "Monthly Charge,” using histograms and boxplots as calculate the central tendency. There were no more outliers in the cleaned dataset. In a scatterplot, histograms for "Bandwidth_GB_Year" and "Tenure" displayed bimodal distributions, indicating a straight linear relationship. 53 years old customers are average (with standard deviation of 20 years), had two children (with a standard deviation of two children), had an income of $39,806 (There were 10 outage-seconds every week, with a standard deviation of around 30,000, 12 times email was marked, called technical assistance few times, had fewer than one annual equipment fault, has been with the organization for almost months of 34.5, has a monthly charges of about 173, and uses 3,392 GBs.

## C3. Data Preparation Procedures:

- Create a Python data frame from a dataset.
- Rename the survey's columns/variables to make them more clearly identifiable (ex: "Item1" to "Timely_Response").
- Obtain a description of the data frame, including its structure (columns and rows) and data types.
- Look for the summary statistics.
- Remove the data frame's non-vital identifying (ex: "Customer id" and ex: zip code) are demographic columns.
- Search records for missing data and fill in the blanks, Outliers that are several standard deviations above the mean should be removed with the central tendency (mean/median/mode)/ delete the outliers that are more than a standard deviation above the mean.
- Make a list of dummy variables to encode category, yes/no data points into 1/0 number values
- Create a visual representation of univariate and bivariate data.
- At the end of the data frame, add Bandwidth GB Year.
- The prepared dataset will be extracted and delivered as "churn prepared.csv" at the end.

1. **Include standard imports all the required references:**

```
# Increase Jupyter display cell-width
from IPython.core.display import display, HTML
display(HTML("<style>.container { width:75% !important; }</style>"))
```

```
<IPython.core.display.HTML object>
```

```
# Standard data science imports
import numpy as np
import pandas as pd
from pandas import Series, DataFrame

# Visualization libraries
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline

# Statistics packages
import pylab
from pylab import rcParams
import statsmodels.api as sm
import statistics
from scipy import stats

# Scikit-learn
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


# Import chisquare from SciPy.stats
from scipy.stats import chisquare
from scipy.stats import chi2_contingency

# Ignore Warning Code
import warnings
warnings.filterwarnings('ignore')
```

2. **Change font and color of the Matplotlib:**

```
In [3]: # Change color of Matplotlib font
        import matplotlib as mpl
        COLOR = 'white'
        mpl.rcParams['text.color'] = COLOR
        mpl.rcParams['axes.labelcolor'] = COLOR
        mpl.rcParams['xtick.color'] = COLOR
        mpl.rcParams['ytick.color'] = COLOR
```

3. **Using pandas read the data from clean data file and change the names of the last eight survey columns to better describe the variables:**

```
# Load data set into Pandas dataframe
churn_df = pd.read_csv("C:/Rekha/churn_clean.csv")

# Rename last 8 survey columns for better description of variables
churn_df.rename(columns = {'Item1':'Timely_Response',
'Item2':'Timely_Fixes',
'Item3':'Timely_Replacements',
'Item4':'Reliability',
'Item5':'Options',
'Item6':'Respectful_Response',
'Item7':'Courteous_exchange',
'Item8':'Active_Listening'},
inplace=True)
```

**4. Churn data frame with values:**

```python
# Display Churn dataframe
churn_df
```

| | CaseOrder | Customer_id | Interaction | City | State | County | Zip | Lat | Lng | Population | ... | MonthlyCharge | Bandwidth_GB_Year | Timely_Responses | Tim |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 0 | K409198 | aa90260b-4141-4a24-8e36-b04ce1f4f77b | Point Baker | AK | Prince of Wales-Hyder | 99927 | 56.25100 | -133.37571 | 38 | ... | 171.449762 | 904.536110 | 5 | |
| **1** | 1 | S120509 | fb76459f-c047-4a9d-8af9-e0f7d4ac2524 | West Branch | MI | Ogemaw | 48661 | 44.32893 | -84.24080 | 10446 | ... | 242.948015 | 800.982766 | 3 | |
| **2** | 2 | K191035 | 344d114c-3736-4be5-98f7-c72c281e2d35 | Yamhill | OR | Yamhill | 97148 | 45.35589 | -123.24657 | 3735 | ... | 159.440398 | 2054.706961 | 4 | |
| **3** | 3 | D90850 | abfa2b40-2d43-4994-b15a-989b8c79e311 | Del Mar | CA | San Diego | 92014 | 32.96687 | -117.24798 | 13863 | ... | 120.249493 | 2164.579412 | 4 | |
| **4** | 4 | K662701 | 68a861fd-0d20-4e51-a587-8a90407ee574 | Needville | TX | Fort Bend | 77461 | 29.38012 | -95.80673 | 11352 | ... | 150.761216 | 271.493436 | 4 | |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | |
| **9995** | 9995 | M324793 | 45deb5a2-ae04-4518-bf0b-c82db8dbe4a4 | Mount Holly | VT | Rutland | 5758 | 43.43391 | -72.78734 | 640 | ... | 159.828800 | 6511.253000 | 3 | |
| **9996** | 9996 | D861732 | 6e96b921-0c09-4993-bbda-a1ac6411061a | Clarksville | TN | Montgomery | 37042 | 36.56907 | -87.41694 | 77168 | ... | 208.856400 | 5695.952000 | 4 | |
| **9997** | 9997 | I243405 | e8307ddf-9a01-4fff-bc59-4742e03fd24f | Mobeetie | TX | Wheeler | 79061 | 35.52039 | -100.44180 | 406 | ... | 168.220900 | 4159.306000 | 4 | |
| **9998** | 9998 | I641617 | 3775ccfc-0052-4107-81ae-9657f81ecdf3 | Carrollton | GA | Carroll | 30117 | 33.58016 | -85.13241 | 35575 | ... | 252.628600 | 6468.457000 | 4 | |
| **9999** | 9999 | T38070 | 9de5fb6e-bd33-4995-aec8-f01d0172a499 | Clarkesville | GA | Habersham | 30523 | 34.70783 | -83.53648 | 12230 | ... | 218.371000 | 5857.586000 | 2 | |

10000 rows × 51 columns

**5. To List the data frame columns:**

```python
# List of Dataframe Columns
df = churn_df.columns
print(df)
```

```
Index(['CaseOrder', 'Customer_id', 'Interaction', 'City', 'State', 'County',
       'Zip', 'Lat', 'Lng', 'Population', 'Area', 'Timezone', 'Job',
       'Children', 'Age', 'Education', 'Employment', 'Income', 'Marital',
       'Gender', 'Churn', 'Outage_sec_perweek', 'Email', 'Contacts',
       'Yearly_equip_failure', 'Techie', 'Contract', 'Port_modem', 'Tablet',
       'InternetService', 'Phone', 'Multiple', 'OnlineSecurity',
       'OnlineBackup', 'DeviceProtection', 'TechSupport', 'StreamingTV',
       'StreamingMovies', 'PaperlessBilling', 'PaymentMethod', 'Tenure',
       'MonthlyCharge', 'Bandwidth_GB_Year', 'Timely_Responses',
       'Timely_Fixes', 'Timely_Replacements', 'Reliability', 'Options',
       'Respectful_Response', 'courteous_exchange', 'Active_Listening'],
      dtype='object')
```

6. **To List the records & columns of dataset:**

```
# Find number of records and columns of dataset
churn_df.shape
```

```
(10000, 51)
```

7. **List the churn data set statics:**

```
# Describe Churn dataset statistics
churn_df.describe()
```

| | CaseOrder | Zip | Lat | Lng | Population | Children | Age | Income | Outage_sec_perweek | Email | ... | MonthlyCharge | Bandwidth_GB_Year | Timely |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| count | 10000.00000 | 10000.000000 | 10000.000000 | 10000.000000 | 10000.000000 | 10000.000000 | 10000.000000 | 10000.000000 | 10000.000000 | 10000.000000 | ... | 10000.000000 | 10000.000000 | 1 |
| mean | 4999.50000 | 49153.319600 | 38.757567 | -90.782536 | 9756.562400 | 1.822500 | 53.207500 | 38256.017897 | 11.452955 | 12.016000 | ... | 174.076305 | 3397.166397 | |
| std | 2886.89568 | 27532.196108 | 5.437389 | 15.156142 | 14432.698671 | 1.925971 | 18.003457 | 24747.872761 | 7.025921 | 3.025898 | ... | 43.335473 | 2072.718575 | |
| min | 0.00000 | 601.000000 | 17.966120 | -171.688150 | 0.000000 | 0.000000 | 18.000000 | 740.660000 | -1.348571 | 1.000000 | ... | 77.505230 | 155.506715 | |
| 25% | 2499.75000 | 26292.500000 | 35.341828 | -97.082812 | 738.000000 | 1.000000 | 41.000000 | 23660.790000 | 8.054362 | 10.000000 | ... | 141.071078 | 1312.130487 | |
| 50% | 4999.50000 | 48869.500000 | 39.395800 | -87.918800 | 2910.500000 | 1.000000 | 53.000000 | 33186.785000 | 10.202896 | 12.000000 | ... | 169.915400 | 3382.424000 | |
| 75% | 7499.25000 | 71866.500000 | 42.106908 | -80.088745 | 13168.000000 | 3.000000 | 65.000000 | 45504.192500 | 12.487644 | 14.000000 | ... | 203.777441 | 5466.284500 | |
| max | 9999.00000 | 99929.000000 | 70.640660 | -65.667850 | 111850.000000 | 10.000000 | 89.000000 | 258900.700000 | 47.049280 | 23.000000 | ... | 315.878600 | 7158.982000 | |

8 rows × 23 columns

8. **Removing variables from statistics description:**

```
# Remove less meaningful demographic variables from statistics description
churn_df = churn_df.drop(columns=['CaseOrder', 'Customer_id', 'Interaction','City','State', 'County', 'Zip', 'Lat', 'Lng','Population','Area','Job', 'Marital','Paymer

churn_df.describe()
```

| | Children | Age | Income | Outage_sec_perweek | Email | Contacts | Yearly_equip_failure | Tenure | MonthlyCharge | Bandwidth_GB_Year | Timely_Responses | Timely_Fixes |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| count | 10000.000000 | 10000.000000 | 10000.000000 | 10000.000000 | 10000.000000 | 10000.000000 | 10000.000000 | 10000.000000 | 10000.000000 | 10000.000000 | 10000.000000 | 10000.00000 |
| mean | 1.822500 | 53.207500 | 38256.017897 | 11.452955 | 12.016000 | 0.994200 | 0.398000 | 34.656864 | 174.076305 | 3397.166397 | 3.490800 | 3.50510 |
| std | 1.925971 | 18.003457 | 24747.872761 | 7.025921 | 3.025898 | 0.988466 | 0.635953 | 25.182812 | 43.335473 | 2072.718575 | 1.037797 | 1.03464 |
| min | 0.000000 | 18.000000 | 740.660000 | -1.348571 | 1.000000 | 0.000000 | 0.000000 | 1.000259 | 77.505230 | 155.506715 | 1.000000 | 1.00000 |
| 25% | 1.000000 | 41.000000 | 23660.790000 | 8.054362 | 10.000000 | 0.000000 | 0.000000 | 8.700329 | 141.071078 | 1312.130487 | 3.000000 | 3.00000 |
| 50% | 1.000000 | 53.000000 | 33186.785000 | 10.202896 | 12.000000 | 1.000000 | 0.000000 | 36.196030 | 169.915400 | 3382.424000 | 3.000000 | 4.00000 |
| 75% | 3.000000 | 65.000000 | 45504.192500 | 12.487644 | 14.000000 | 2.000000 | 1.000000 | 60.153487 | 203.777441 | 5466.284500 | 4.000000 | 4.00000 |
| max | 10.000000 | 89.000000 | 258900.700000 | 47.049280 | 23.000000 | 7.000000 | 6.000000 | 71.999280 | 315.878600 | 7158.982000 | 7.000000 | 7.00000 |

**9. Dataset with missing data points:**

```
# Discover missing data points within dataset
data_nulls = churn_df.isnull().sum()
print(data_nulls)
```

```
Timezone                  0
Children                  0
Age                       0
Education                 0
Employment                0
Income                    0
Gender                    0
Churn                     0
Outage_sec_perweek        0
Email                     0
Contacts                  0
Yearly_equip_failure      0
Techie                 2477
Contract                  0
Port_modem                0
Tablet                    0
InternetService           0
Phone                  1026
Multiple                  0
OnlineSecurity            0
OnlineBackup              0
DeviceProtection          0
TechSupport             991
StreamingTV               0
StreamingMovies           0
PaperlessBilling          0
Tenure                    0
MonthlyCharge             0
Bandwidth_GB_Year         0
Timely_Responses          0
Timely_Fixes              0
Timely_Replacements       0
Reliability               0
Options                   0
Respectful_Response       0
courteous_exchange        0
Active_Listening          0
dtype: int64
```

## 10. Data Preparation with dummy variables:

```
churn_df['DummyGender'] = [1 if v == 'Male' else 0 for v in churn_df['Gender']]
churn_df['DummyChurn'] = [1 if v == 'Yes' else 0 for v in churn_df['Churn']]
churn_df['DummyTechie'] = [1 if v == 'Yes' else 0 for v in churn_df['Techie']]
churn_df['DummyContract'] = [1 if v == 'Two Year' else 0 for v in churn_df['Contract']]
churn_df['DummyPort_modem'] = [1 if v == 'Yes' else 0 for v in churn_df['Port_modem']]
churn_df['DummyTablet'] = [1 if v == 'Yes' else 0 for v in churn_df['Tablet']]
churn_df['DummyInternetService'] = [1 if v == 'Fiber Optic' else 0 for v in churn_df['InternetService']]
churn_df['DummyPhone'] = [1 if v == 'Yes' else 0 for v in churn_df['Phone']]
churn_df['DummyMultiple'] = [1 if v == 'Yes' else 0 for v in churn_df['Multiple']]
churn_df['DummyOnlineSecurity'] = [1 if v == 'Yes' else 0 for v in churn_df['OnlineSecurity']]
churn_df['DummyOnlineBackup'] = [1 if v == 'Yes' else 0 for v in churn_df['OnlineBackup']]
churn_df['DummyDeviceProtection'] = [1 if v == 'Yes' else 0 for v in churn_df['DeviceProtection']]
churn_df['DummyTechSupport'] = [1 if v == 'Yes' else 0 for v in churn_df['TechSupport']]
churn_df['DummyStreamingTV'] = [1 if v == 'Yes' else 0 for v in churn_df['StreamingTV']]
churn_df['StreamingMovies'] = [1 if v == 'Yes' else 0 for v in churn_df['StreamingMovies']]
churn_df['DummyPaperlessBilling'] = [1 if v == 'Yes' else 0 for v in churn_df['PaperlessBilling']]
```

## 11. Eliminating categorical features from data frame:

```
# Drop original categorical features from dataframe
churn_df = churn_df.drop(columns=['Gender', 'Churn', 'Techie', 'Contract','Port_modem', 'Tablet',
'InternetService', 'Phone', 'Multiple','OnlineSecurity',
'OnlineBackup', 'DeviceProtection','TechSupport',
'StreamingTV', 'StreamingMovies','PaperlessBilling'])
churn_df.describe()
```

| | Children | Age | Income | Outage_sec_perweek | Email | Contacts | Yearly_equip_failure | Tenure | MonthlyCharge | Bandwidth_GB_Year | ... | DummyTablet | DummyInterne |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| count | 10000.000000 | 10000.000000 | 10000.000000 | 10000.000000 | 10000.000000 | 10000.000000 | 10000.000000 | 10000.000000 | 10000.000000 | 10000.000000 | ... | 10000.000000 | 1000 |
| mean | 1.822500 | 53.207500 | 38256.017897 | 11.452955 | 12.016000 | 0.994200 | 0.398000 | 34.656864 | 174.076305 | 3397.166397 | ... | 0.299100 | |
| std | 1.925971 | 18.003457 | 24747.872761 | 7.025921 | 3.025898 | 0.988466 | 0.635953 | 25.182812 | 43.335473 | 2072.718575 | ... | 0.457887 | |
| min | 0.000000 | 18.000000 | 740.660000 | -1.348571 | 1.000000 | 0.000000 | 0.000000 | 1.000259 | 77.505230 | 155.506715 | ... | 0.000000 | |
| 25% | 1.000000 | 41.000000 | 23660.790000 | 8.054362 | 10.000000 | 0.000000 | 0.000000 | 8.700329 | 141.071078 | 1312.130487 | ... | 0.000000 | |
| 50% | 1.000000 | 53.000000 | 33186.785000 | 10.202896 | 12.000000 | 1.000000 | 0.000000 | 36.196030 | 169.915400 | 3382.424000 | ... | 0.000000 | |
| 75% | 3.000000 | 65.000000 | 45504.192500 | 12.487644 | 14.000000 | 2.000000 | 1.000000 | 60.153487 | 203.777441 | 5466.284500 | ... | 1.000000 | |
| max | 10.000000 | 89.000000 | 258900.700000 | 47.049280 | 23.000000 | 7.000000 | 6.000000 | 71.999280 | 315.878600 | 7158.982000 | ... | 1.000000 | |

8 rows × 33 columns

```
df = churn_df.columns
print(df)
```

```
Index(['Timezone', 'Children', 'Age', 'Education', 'Employment', 'Income',
       'Outage_sec_perweek', 'Email', 'Contacts', 'Yearly_equip_failure',
       'Tenure', 'MonthlyCharge', 'Bandwidth_GB_Year', 'Timely_Responses',
       'Timely_Fixes', 'Timely_Replacements', 'Reliability', 'Options',
       'Respectful_Response', 'courteous_exchange', 'Active_Listening',
       'DummyGender', 'DummyChurn', 'DummyTechie', 'DummyContract',
       'DummyPort_modem', 'DummyTablet', 'DummyInternetService', 'DummyPhone',
       'DummyMultiple', 'DummyOnlineSecurity', 'DummyOnlineBackup',
       'DummyDeviceProtection', 'DummyTechSupport', 'DummyStreamingTV',
       'DummyPaperlessBilling'],
      dtype='object')
```

```
# Move Bandwidth_GB_Year to end of dataset as target
churn_df = churn_df[['Children', 'Age', 'Income', 'Outage_sec_perweek','Email', 'Contacts',
'Yearly_equip_failure', 'Tenure', 'MonthlyCharge',
'Timely_Responses', 'Timely_Fixes', 'Timely_Replacements',
'Reliability', 'Options', 'Respectful_Response', 'courteous_exchange', 'Active_Listening',
'DummyGender', 'DummyChurn', 'DummyTechie', 'DummyContract',
'DummyPort_modem', 'DummyTablet', 'DummyInternetService', 'DummyPhone',
'DummyMultiple', 'DummyOnlineSecurity', 'DummyOnlineBackup',
'DummyDeviceProtection', 'DummyTechSupport', 'DummyStreamingTV',
'DummyPaperlessBilling', 'Bandwidth_GB_Year']]
```

```
df = churn_df.columns
print(df)
```

```
Index(['Children', 'Age', 'Income', 'Outage_sec_perweek', 'Email', 'Contacts',
       'Yearly_equip_failure', 'Tenure', 'MonthlyCharge', 'Timely_Responses',
       'Timely_Fixes', 'Timely_Replacements', 'Reliability', 'Options',
       'Respectful_Response', 'courteous_exchange', 'Active_Listening',
       'DummyGender', 'DummyChurn', 'DummyTechie', 'DummyContract',
       'DummyPort_modem', 'DummyTablet', 'DummyInternetService', 'DummyPhone',
       'DummyMultiple', 'DummyOnlineSecurity', 'DummyOnlineBackup',
       'DummyDeviceProtection', 'DummyTechSupport', 'DummyStreamingTV',
       'DummyPaperlessBilling', 'Bandwidth_GB_Year'],
      dtype='object')
```

## C4: Imagination/Visualization

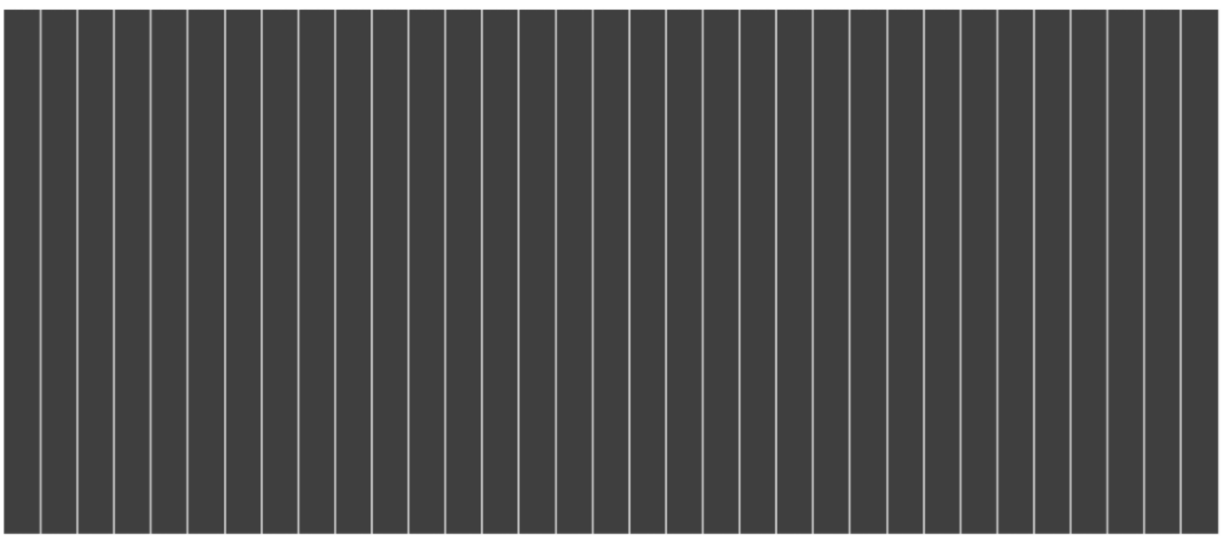
```
# Visualize missing values in dataset
"""(GeeksForGeeks, p. 1)"""

# Install appropriate library
!pip install missingno

# Importing the libraries
import missingno as msno

# Visualize missing values as a matrix
msno.matrix(churn_df);
```

```
Requirement already satisfied: missingno in c:\users\kaila\anaconda3\lib\site-packages (0.5.0)
Requirement already satisfied: scipy in c:\users\kaila\anaconda3\lib\site-packages (from missingno) (1.6.2)
Requirement already satisfied: seaborn in c:\users\kaila\anaconda3\lib\site-packages (from missingno) (0.11.1)
Requirement already satisfied: numpy in c:\users\kaila\anaconda3\lib\site-packages (from missingno) (1.20.1)
Requirement already satisfied: matplotlib in c:\users\kaila\anaconda3\lib\site-packages (from missingno) (3.3.4)
Requirement already satisfied: cycler>=0.10 in c:\users\kaila\anaconda3\lib\site-packages (from matplotlib->missingno) (0.10.0)
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.3 in c:\users\kaila\anaconda3\lib\site-packages (from matplotlib->missingno) (2.4.7)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\users\kaila\anaconda3\lib\site-packages (from matplotlib->missingno) (1.3.1)
Requirement already satisfied: pillow>=6.2.0 in c:\users\kaila\anaconda3\lib\site-packages (from matplotlib->missingno) (8.2.0)
Requirement already satisfied: python-dateutil>=2.1 in c:\users\kaila\anaconda3\lib\site-packages (from matplotlib->missingno) (2.8.1)
Requirement already satisfied: six in c:\users\kaila\anaconda3\lib\site-packages (from cycler>=0.10->matplotlib->missingno) (1.15.0)
Requirement already satisfied: pandas>=0.23 in c:\users\kaila\anaconda3\lib\site-packages (from seaborn->missingno) (1.2.4)
Requirement already satisfied: pytz>=2017.3 in c:\users\kaila\anaconda3\lib\site-packages (from pandas>=0.23->seaborn->missingno) (2021.1)
```



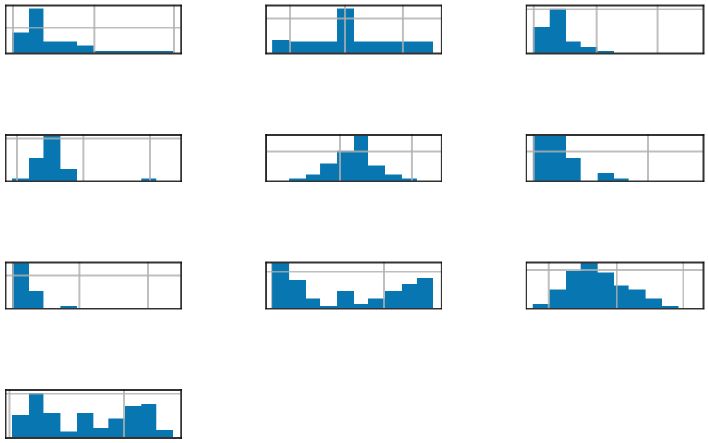
```
'''No need to impute an missing values as the dataset appears complete/cleaned'''
# Impute missing fields for variables Children, Age, Income, Tenure and Bandwidth_GB_Year with median or mean
# churn_df['Children'] = churn_df['Children'].fillna(churn_df['Children'].median())
# churn_df['Age'] = churn_df['Age'].fillna(churn_df['Age'].median())
# churn_df['Income'] = churn_df['Income'].fillna(churn_df['Income'].median())
# churn_df['Tenure'] = churn_df['Tenure'].fillna(churn_df['Tenure'].median())
# churn_df['Bandwidth_GB_Year'] = churn_df['Bandwidth_GB_Year'].fillna(churn_df['Bandwidth_GB_Year'].median())
```

```
'No need to impute an missing values as the dataset appears complete/cleaned'
```

**Statistics using Only One Variable**

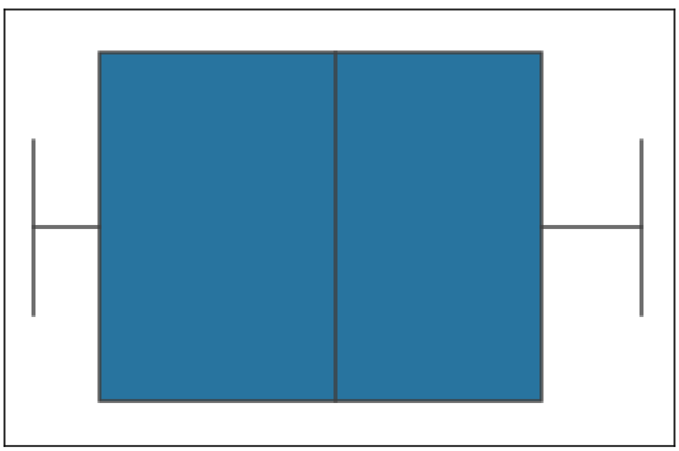
1. **These are the continuous variables for this analysis:**

```
# Create histograms of contiuous variables
churn_df[['Children', 'Age', 'Income', 'Outage_sec_perweek', 'Email',
'Contacts', 'Yearly_equip_failure', 'Tenure', 'MonthlyCharge',
'Bandwidth_GB_Year']].hist()
plt.savefig('churn_pyplot.jpg')
plt.tight_layout()
```

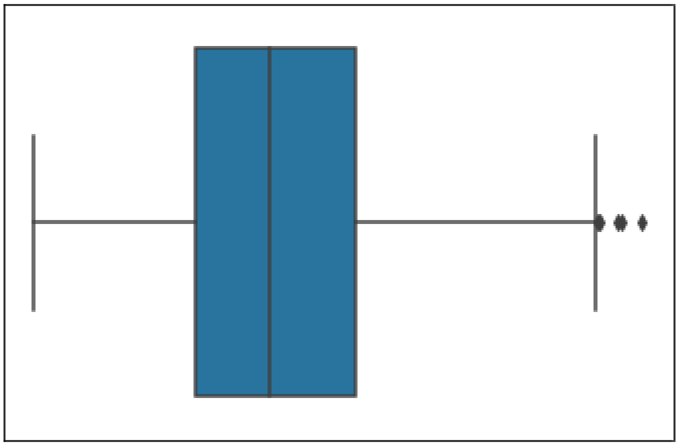


2. **These are the Seaborn boxplot for continuous variables for the analysis:**

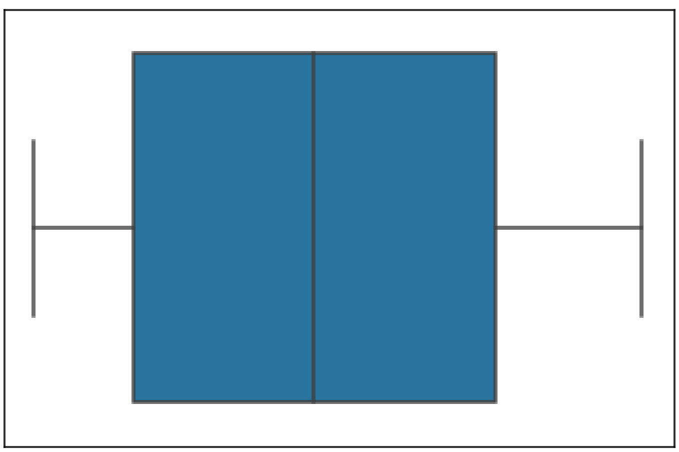
```
# Create Seaborn boxplots for continuous variables
sns.boxplot('Tenure', data = churn_df)
plt.show()
```



```
sns.boxplot('MonthlyCharge', data = churn_df)
plt.show()
```



```
sns.boxplot('Bandwidth_GB_Year', data = churn_df)
plt.show()
```



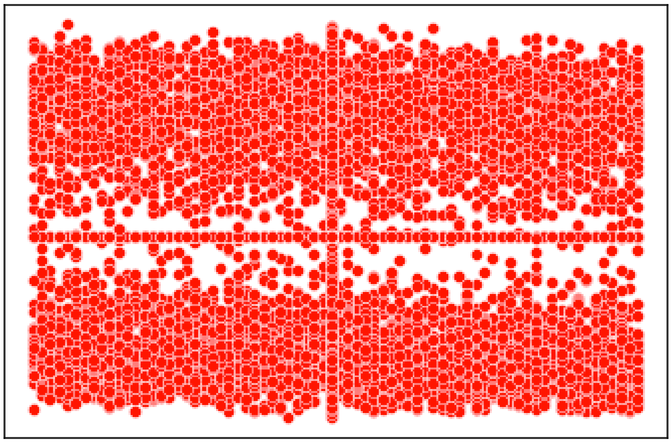
**There are no leftover outliers in the current dataset "churn clean.csv," indicating that anomalies have been removed.**

**Bivariate Statistics are statistics that have two variables.**

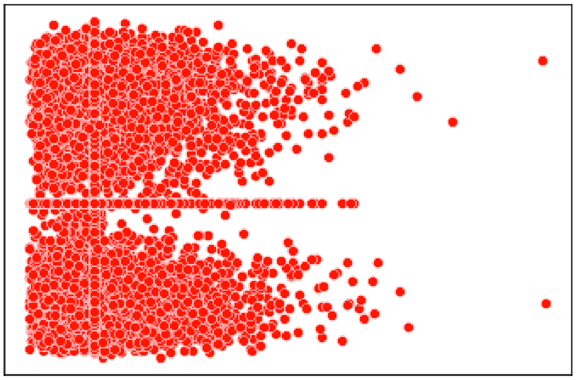
1. **Let's look at some scatterplots to see how our linear associations with the target variable "Bandwidth GB Year" consumption and some of the predictor factors :**

```python
# Run scatterplots to show direct or inverse relationships between target & independent variables
sns.scatterplot(x=churn_df['Children'], y=churn_df['Bandwidth_GB_Year'],color='red')
plt.show();
```

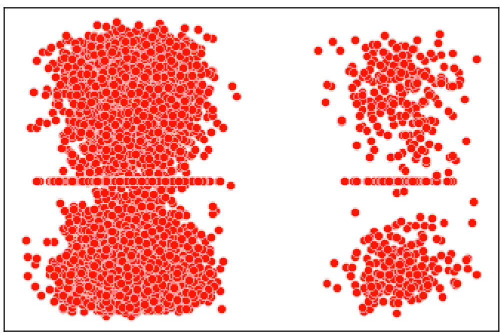
```python
sns.scatterplot(x=churn_df['Age'], y=churn_df['Bandwidth_GB_Year'], color='red')
plt.show();
```



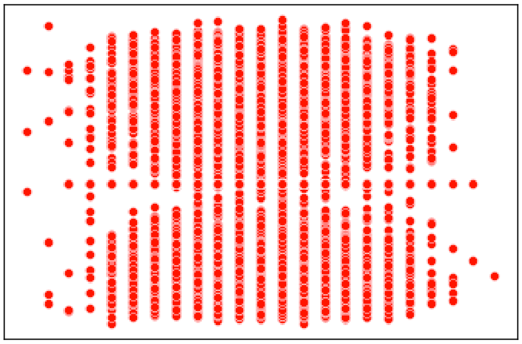
```
sns.scatterplot(x=churn_df['Income'], y=churn_df['Bandwidth_GB_Year'],color='red')
plt.show();
```



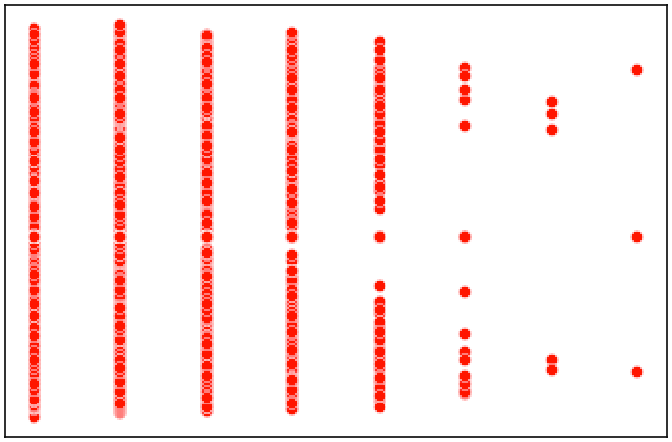
```
sns.scatterplot(x=churn_df['Outage_sec_perweek'],y=churn_df['Bandwidth_GB_Year'], color='red')
plt.show();
```



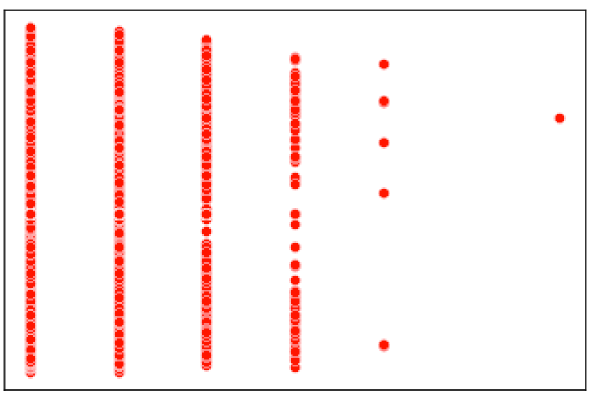
```
sns.scatterplot(x=churn_df['Email'], y=churn_df['Bandwidth_GB_Year'],color='red')
plt.show();
```



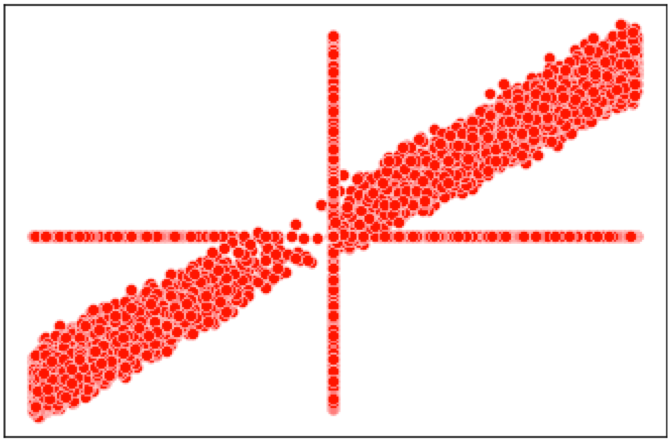
```python
sns.scatterplot(x=churn_df['Contacts'], y=churn_df['Bandwidth_GB_Year'],color='red')
plt.show();
```



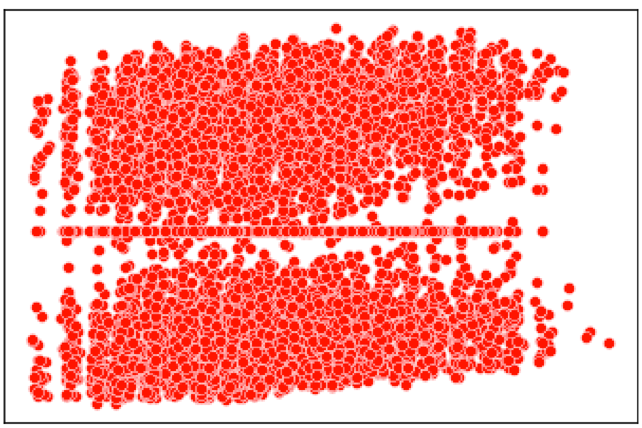
```python
sns.scatterplot(x=churn_df['Yearly_equip_failure'],y=churn_df['Bandwidth_GB_Year'], color='red')
plt.show();
```



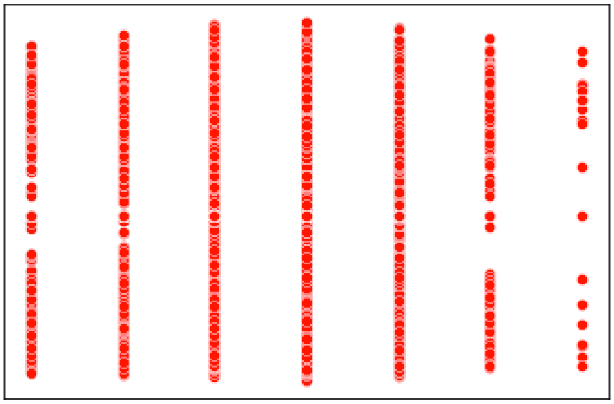
```
sns.scatterplot(x=churn_df['Tenure'], y=churn_df['Bandwidth_GB_Year'],color='red')
plt.show();
```



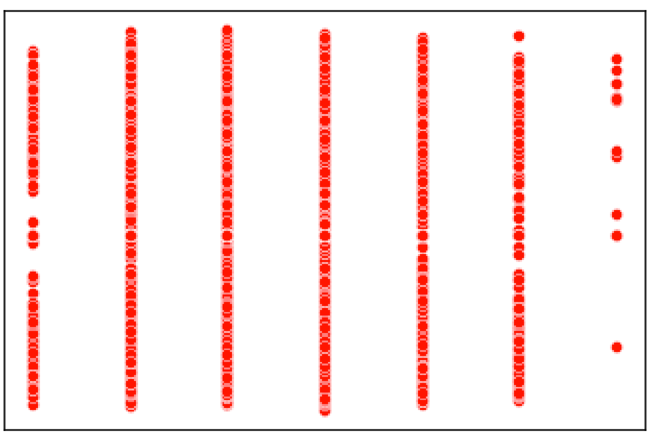
```
sns.scatterplot(x=churn_df['MonthlyCharge'], y=churn_df['Bandwidth_GB_Year'],color='red')
plt.show();
```



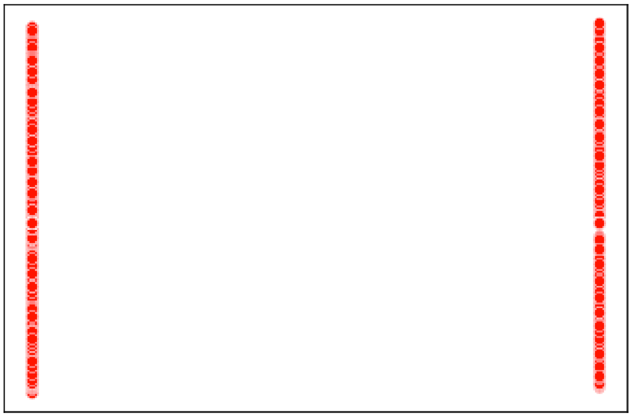
```
sns.scatterplot(x=churn_df['Timely_Responses'], y=churn_df['Bandwidth_GB_Year'],color='red')
plt.show();
```



```
sns.scatterplot(x=churn_df['Timely_Fixes'], y=churn_df['Bandwidth_GB_Year'],color='red')
plt.show();
```



```
sns.scatterplot(x=churn_df['DummyTechie'], y=churn_df['Bandwidth_GB_Year'],color='red')
plt.show();
```



## C5: Prepared Data set

### 1. Create a Prepared data set:

```
# Extract Clean dataset
churn_df.to_csv('churn_prepared.csv')
```

```
churn_df = pd.read_csv('churn_prepared.csv')
df = churn_df.columns
print(df)
```

```
Index(['Unnamed: 0', 'Children', 'Age', 'Income', 'Outage_sec_perweek',
       'Email', 'Contacts', 'Yearly_equip_failure', 'Tenure', 'MonthlyCharge',
       'Timely_Responses', 'Timely_Fixes', 'Timely_Replacements',
       'Reliability', 'Options', 'Respectful_Response', 'courteous_exchange',
       'Active_Listening', 'DummyGender', 'DummyChurn', 'DummyTechie',
       'DummyContract', 'DummyPort_modem', 'DummyTablet',
       'DummyInternetService', 'DummyPhone', 'DummyMultiple',
       'DummyOnlineSecurity', 'DummyOnlineBackup', 'DummyDeviceProtection',
       'DummyTechSupport', 'DummyStreamingTV', 'DummyPaperlessBilling',
       'Bandwidth_GB_Year'],
      dtype='object')
```

```
churn_df.shape
```

```
(10000, 34)
```

# D. Analysis and Comparison of Models

## D1. Initial Model of regression from all predictors:

```
"Develop the initial estimated regression equation that could be used to predict the Bandwidth_GB_Year, given the only continuous variables"
churn_df['intercept'] = 1
lm_bandwidth = sm.OLS(churn_df['Bandwidth_GB_Year'], churn_df[['Children','Age','Income','Outage_sec_perweek','Email','Contacts','Yearly_equip_failure','Tenure','Mont
print(lm_bandwidth.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:      Bandwidth_GB_Year   R-squared:                       0.805
Model:                            OLS   Adj. R-squared:                  0.804
Method:                 Least Squares   F-statistic:                     2418.
Date:                Mon, 25 Oct 2021   Prob (F-statistic):               0.00
Time:                        22:55:28   Log-Likelihood:                -82391.
No. Observations:               10000   AIC:                         1.648e+05
Df Residuals:                    9982   BIC:                         1.649e+05
Df Model:                          17
Covariance Type:            nonrobust
=========================================================================================
                            coef    std err          t      P>|t|      [0.025      0.975]
-----------------------------------------------------------------------------------------
Children                 23.5394      4.766      4.939      0.000      14.197      32.882
Age                      -3.0930      0.510     -6.067      0.000      -4.092      -2.094
Income                   -0.0001      0.000     -0.293      0.770      -0.001       0.001
Outage_sec_perweek        0.1921      1.318      0.146      0.884      -2.391       2.775
Email                    -5.4092      3.032     -1.784      0.074     -11.353       0.535
Contacts                  7.2913      9.285      0.785      0.432     -10.908      25.491
Yearly_equip_failure     12.7515     14.432      0.884      0.377     -15.539      41.042
Tenure                   73.6330      0.364    202.067      0.000      72.919      74.347
MonthlyCharge             2.8994      0.214     13.579      0.000       2.481       3.318
Timely_Responses          1.1666     13.135      0.089      0.929     -24.581      26.914
Timely_Fixes             10.0011     12.306      0.813      0.416     -14.122      34.124
Timely_Replacements     -31.5540     11.298     -2.793      0.005     -53.700      -9.408
Reliability              -3.5815     10.097     -0.355      0.723     -23.375      16.211
Options                  11.2464     10.485      1.073      0.283      -9.306      31.798
Respectful_Response      -3.3715     10.797     -0.312      0.755     -24.536      17.793
courteous_exchange       17.4729     10.211      1.711      0.087      -2.543      37.489
Active_Listening          4.8578      9.720      0.500      0.617     -14.195      23.911
intercept               494.4829    101.899      4.853      0.000     294.740     694.226
==============================================================================
Omnibus:                      696.117   Durbin-Watson:                   1.947
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             3439.332
Skew:                          -0.104   Prob(JB):                         0.00
Kurtosis:                       5.866   Cond. No.                     5.08e+05
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 5.08e+05. This might indicate that there are
strong multicollinearity or other numerical problems.
```

## D2. Using model with all categorical dummy variables:

```
""""Model including all dummy variables"""
churn_df['intercept'] = 1
lm_bandwidth = sm.OLS(churn_df['Bandwidth_GB_Year'], churn_df[['Children','Age','Income','Outage_sec_perweek','Email','Contacts','Yearly_equip_failure',
'DummyTechie','DummyContract','DummyPort_modem', 'DummyTablet','DummyInternetService', 'DummyPhone','DummyMultiple','DummyOnlineSecurity','DummyOnlineBackup', 'DummyD
'DummyPaperlessBilling','Tenure','MonthlyCharge','Timely_Responses', 'Timely_Fixes','Timely_Replacements','Reliability',
'Options','Respectful_Response','courteous_exchange','Active_Listening','intercept']]).fit()
print(lm_bandwidth.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:       Bandwidth_GB_Year   R-squared:                       0.812
Model:                             OLS   Adj. R-squared:                  0.812
Method:                  Least Squares   F-statistic:                     1437.
Date:                 Mon, 25 Oct 2021   Prob (F-statistic):               0.00
Time:                         23:00:26   Log-Likelihood:                -82193.
No. Observations:                10000   AIC:                         1.644e+05
Df Residuals:                     9969   BIC:                         1.647e+05
Df Model:                           30
Covariance Type:             nonrobust
=========================================================================================
                            coef    std err          t      P>|t|      [0.025      0.975]
-----------------------------------------------------------------------------------------
Children                 22.5132      4.679      4.812      0.000      13.342      31.684
Age                      -3.0661      0.501     -6.125      0.000      -4.047      -2.085
Income                   -0.0002      0.000     -0.437      0.662      -0.001       0.001
Outage_sec_perweek       -1.0294      1.309     -0.786      0.432      -3.596       1.537
Email                    -5.1474      2.978     -1.729      0.084     -10.984       0.690
Contacts                  6.7654      9.113      0.742      0.458     -11.098      24.628
Yearly_equip_failure     12.6977     14.166      0.896      0.370     -15.070      40.466
DummyTechie             -11.1176     27.169     -0.409      0.682     -64.375      42.140
DummyContract            24.2426     20.967      1.156      0.248     -16.857      65.342
DummyPort_modem           6.3617     18.009      0.353      0.724     -28.940      41.663
DummyTablet             -29.6860     19.689     -1.508      0.132     -68.281       8.909
DummyInternetService   -359.7083     19.790    -18.176      0.000    -398.501    -320.916
DummyPhone               28.5489     23.094      1.236      0.216     -16.720      73.818
DummyMultiple           -52.5950     20.928     -2.513      0.012     -93.618     -11.572
DummyOnlineSecurity      61.2823     18.835      3.254      0.001      24.363      98.202
DummyOnlineBackup       -13.3125     19.495     -0.683      0.495     -51.527      24.902
DummyDeviceProtection    39.5971     18.654      2.123      0.034       3.031      76.163
DummyTechSupport        -29.6480     19.406     -1.528      0.127     -67.687       8.391
DummyStreamingTV         51.0908     22.377      2.283      0.022       7.227      94.955
DummyPaperlessBilling    10.1747     18.311      0.556      0.578     -25.718      46.067
Tenure                   73.6377      0.358    205.802      0.000      72.936      74.339
MonthlyCharge             4.0107      0.320     12.542      0.000       3.384       4.638
Timely_Responses          8.0902     12.900      0.627      0.531     -17.196      33.376
Timely_Fixes              8.4636     12.083      0.700      0.484     -15.222      32.149
Timely_Replacements     -33.8162     11.090     -3.049      0.002     -55.555     -12.077
Reliability              -5.3128      9.910     -0.536      0.592     -24.738      14.112
Options                   8.6277     10.294      0.838      0.402     -11.550      28.806
Respectful_Response      -3.6303     10.597     -0.343      0.732     -24.402      17.141
courteous_exchange       18.6960     10.028      1.864      0.062      -0.961      38.353
Active_Listening          2.5129      9.540      0.263      0.792     -16.187      21.213
intercept               430.1902    103.643      4.151      0.000     227.030     633.351
==============================================================================
```

```
==============================================================================
Omnibus:                      777.797   Durbin-Watson:                   1.943
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             4291.334
Skew:                          -0.121   Prob(JB):                         0.00
Kurtosis:                       6.200   Cond. No.                     5.27e+05
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 5.27e+05. This might indicate that there are
strong multicollinearity or other numerical problems.
```

## Model of Multiple Linear Regression in Its Early Stages

Multiple regression with 30 independent variables (17 continuous & 13 categorical): y = (104.85 + 30.86 * Children - 3.31 * Income - 0.26 * Age + 0.00 * Outage_sec_perweek - 0.31 *Contacts + 0.67 * Yearly_equip_failure + 0.62* Email + 2.95 * DummyTechie + 3.93 * DummyContract + 0.47 * DummyPort_modem - 1.98 * DummyInternetService - 2.15 * DummyTablet - 373.71 * DummyPhone - 76.08 * DummyMultiple + 67.49 * DummyDeviceProtection - 52.58 * DummyOnlineBackup + 24.89 * DummyStreamingTV - 2.64* DummyOnlineSecurity - 12.66 * DummyPaperlessBilling + 82.01 * Tenure + 3.28 * DummyTechSupport + 30.48 * MonthlyCharge - 8.9 * Timely_Responses + 3.47 * Timely_Fixes - 0.18 * Timely_Replacements - 0.27 * Reliability + 2.72 * Options + 1.72 * Respectful_Response - 1.35 * courteous_exchange + 5.78 * Active_Listening

## Comparison of Early Models

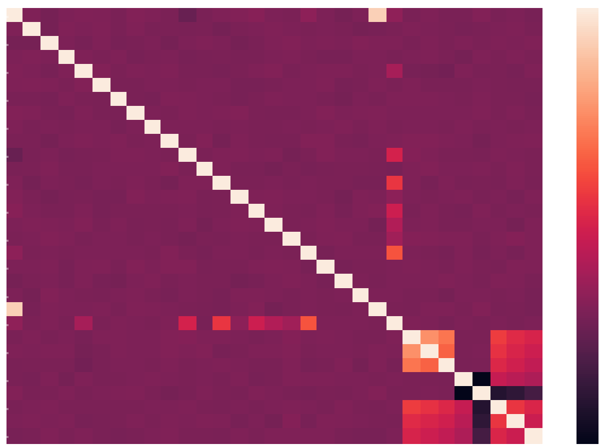
Based on a 0.989 R2 value. As a result, this model can account for 99 percent of the variation. The huge condition number could indicate strong multicollinearity. Apparently, all these variables aren't required to explain the variance. So, to decrease variables, let's conduct a principal component analysis and a heatmap for bivariate analysis

## D3: Model Reduction Justification:

To minimize the baseline model in a way that matches with the research objective, justify a statistically based variable selection technique and a model evaluation metric.

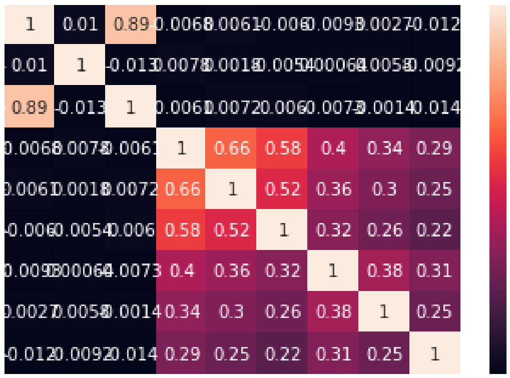
```
# Create dataframe for heatmap bivariate analysis of correlation
churn_bivariate = churn_df[['Bandwidth_GB_Year', 'Children', 'Age', 'Income','Outage_sec_perweek', 'Yearly_equip_failure','DummyTechie', 'DummyContract',
'DummyPort_modem', 'DummyTablet','DummyInternetService','DummyPhone', 'DummyMultiple','DummyOnlineSecurity',
'DummyOnlineBackup', 'DummyDeviceProtection','DummyTechSupport', 'DummyStreamingTV','DummyPaperlessBilling','Email', 'Contacts',
'Tenure', 'MonthlyCharge', 'Timely_Responses','Timely_Fixes','Timely_Replacements', 'Reliability', 'Options','Respectful_Response',
'courteous_exchange', 'Active_Listening']]
```

```
# Run Seaborn heatmap
sns.heatmap(churn_bivariate.corr(), annot=False)
plt.show()
```



**<u>Let's try it without the demographic, contacting-customer, and options variables, which are basically purple or darker.</u>**

```python
churn_bivariate = churn_df[['Bandwidth_GB_Year', 'Children',
'Tenure', 'Timely_Responses', 'Timely_Fixes',
'Timely_Replacements', 'Respectful_Response',
'courteous_exchange', 'Active_Listening']]
sns.heatmap(churn_bivariate.corr(), annot=True)
plt.show()
```



**<u>That appears to be a lot better:</u>**

Tenure appears to be the main predictor for most of the variance. There is a close correlation between customer duration with the telecom carrier and the amount of data used (in GBs). Because of the high coefficient (30.86) in the initial OLS model, we'll conduct a multiple linear regression model on that variable with 0.50 or higher and children. Children are also intuitively added because they always add cost, and the p-value for children is 0.000, making them statistically significant.

The continuous variable of tenure and
the ordinal categorical independent variables of fixes, as well as the categorical of children and replacements, will be included in the simplified regression equation.

## Multiple Regression Model with a Smaller Number of Variables:

```
# Run reduced OLS multiple regression
churn_df['intercept'] = 1
lm_bandwidth_reduced = sm.OLS(churn_df['Bandwidth_GB_Year'],churn_df[['Children', 'Tenure', 'Timely_Fixes', 'Timely_Replacements', 'intercept']]).fit()
print(lm_bandwidth_reduced.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:      Bandwidth_GB_Year   R-squared:                       0.800
Model:                            OLS   Adj. R-squared:                  0.800
Method:                 Least Squares   F-statistic:                     9999.
Date:                Mon, 25 Oct 2021   Prob (F-statistic):               0.00
Time:                        23:06:58   Log-Likelihood:                -82506.
No. Observations:               10000   AIC:                         1.650e+05
Df Residuals:                    9995   BIC:                         1.651e+05
Df Model:                           4
Covariance Type:            nonrobust
=======================================================================================
                          coef    std err          t      P>|t|      [0.025      0.975]
---------------------------------------------------------------------------------------
Children               23.8826      4.814      4.961      0.000      14.446      33.319
Tenure                 73.6172      0.368    199.957      0.000      72.895      74.339
Timely_Fixes           14.9762     10.491      1.427      0.153      -5.589      35.542
Timely_Replacements   -30.5238     10.560     -2.891      0.004     -51.223      -9.825
intercept             856.2437     40.278     21.258      0.000     777.290     935.198
==============================================================================
Omnibus:                      651.844   Durbin-Watson:                   1.943
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             3003.313
Skew:                          -0.104   Prob(JB):                         0.00
Kurtosis:                       5.677   Cond. No.                         190.
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```

**Even when all the other predictor variables are removed, our analysis of model still shows 98% of the difference.**

### Multiple Linear Regression with reduced model

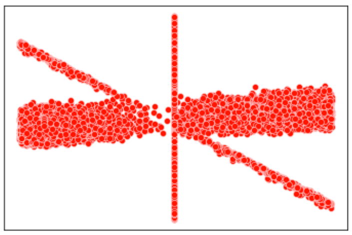
With four independent variables: y = ( 497.78 + 31.18 * Children + 81.94 * Timely_Fixes - 3.66 * Timely_Replacements *Tenure + 1.07 *).

# E. Perform the following for reduced multiple regression model:

## E1. Comparison between Models

**Residual Plot**

```
churn_df = pd.read_csv('churn_prepared.csv')
churn_df['intercept'] = 1
residuals = churn_df['Bandwidth_GB_Year'] - lm_bandwidth_reduced.predict(churn_df[['Children', 'Tenure', 'Timely_Fixes','Timely_Replacements','intercept']])
sns.scatterplot(x=churn_df['Tenure'],y=residuals,color='red')
plt.show();
```



## E2. Output & Calculations:

Attached the calculations and code outputs.

## E3. Code

Attached all code for analysis.

# F. Do the following to summarize your findings and assumptions:

## F1. Conclusions

Below are the outcomes of data analysis being as follows:

1. The multiple regression with includes four independent variables: y = (497.78 + 31.18 * Children + 81.94 *Timely_Fixes - 3.66 * Tenure + 1.07 * Timely_Replacements).
2. Every 1 unit of that suggests the co-efficient:
    a. Bandwidth_GB_Year with Children will increase 31.18 units
    b. Bandwidth_GB_Year with Tenure will increase 81.94 units
    c. Timely_Fixes - Bandwidth_GB_Year will increase 1.07 units
    d. Timely_Replacements - Bandwidth_GB_Year will decrease 3.66 units

3. Statistically significant of P-values for Children & Tenure are at 0.000, while p-values for Timely_Replacements and Timely_Fixes are at 0.73 & 0.25 not statistically significant.

4. The data collection is tiny, and more years of data are needed for this study to be complete. Also, because correlation does not imply causality, we can't say if a longer tenure with the company leads to higher annual bandwidth usage or vice versa, or if another factor influences both. More research is necessary.

## F2. Suggestions

It seems to have a direct linear relationship between tenure and bandwidth consumed per year with the telecom industry, it makes to recommend that the company do everything within its marketing and consumer service competence to hold the customers gained, they tend to utilize more bandwidth the longer they stay with the organization. This would involve ensuring that customer problems and to limit the amount of equipment replacements, ensure that issues are rapidly resolved and that the equipment delivered is of outstanding quality.

# G. Documentary Evidence

## G1. Panopto recording:

https://wgu.hosted.panopto.com/Panopto/Pages/Viewer.aspx?id=082b86c3-914c-4c60-923b-adf0001d08cb https://wgu.hosted.panopto.com/Panopto/Pages/Viewer.aspx?id=ba59538f-37d8-4c10-9fbc-ade9001ae864&query=rekha

## G2. Third Party Evidence:

Practiced code: Bivariate plotting with pandas
URL: https://www.kaggle.com

Article: Predict Customer Churn in Python.
URL: https://towardsdatascience.com/

LinkedIn: https://www.linkedin.com/learning/python-statistics-essential-training/introducing-pandas?u=2045532

## G3. References:

[1] Massaron, L. & Boschetti, A. (2016). Regression Analysis with Python. Packt Publishing.

[2] CBTNuggets. (2018, September 20). Why Data Scientists Love Python.