

D208_Performance_Assessment_NBM2_Task_1

July 12, 2021

1 D208 Performance Assessment NBM2 Task 1

1.1 Multiple Regression for Predictive Modeling

Ryan L. Buchanan Student ID: 001826691 Masters Data Analytics (12/01/2020) Program Mentor: Dan Estes (385) 432-9281 (MST) rbuch49@wgu.edu

1.1.1 A1. Research Question:

How much many GBs of data will a customer use yearly? Can this be predicted accurately from a list of explanatory variables?

1.1.2 A2. Objectives & Goals:

Stakeholders in the company will benefit by knowing, with some measure of confidence, how much data a customer might predictably use. This will provide weight for decisions in whether or not to expand customer data limits, provide unlimited (or metered) media streaming & expand company cloud computing resources for increased bandwidth demands.

1.1.3 B1. Summary of Assumptions:

Assumptions of a multiple regression model include: * There is a linear relationship between the dependent variables & the independent variables. * The independent variables are not too highly correlated with each other. * y_i observations are selected independently & randomly from the population. * Residuals should normally distributed with a mean of zero.

1.1.4 B2. Tool Benefits:

Python & IPython Jupyter notebooks will be used to support this analysis. Python offers very intuitive, simple & versatile programming style & syntax, as well as a large system of mature packages for data science & machine learning. Since, Python is cross-platform, it will work well whether consumers of the analysis are using Windows PCs or a MacBook laptop. It is fast when compared with other possible programming languages like R or MATLAB (Massaron, p. 8). Also, there is strong support for Python as the most popular data science programming language in popular literature & media (CBTNuggets)

1.1.5 B3. Appropriate Technique:

Multiple regression is an appropriate technique to analyze the research question because our target variable, predicting a real number of GBs per year, is a continuous variable (how much data is used). Also, perhaps there are several (versus simply one) explanatory variables (area type, job, children, age, income, etc.) that will add to our understanding when trying to predict how much data a customer will use in a given year. When adding or removing independent variables from our regression equation, we will find out whether or not they have a positive or negative relationship to our target variable & how that might affect company decisions on marketing segmentation.

1.1.6 C1. Data Goals:

My approach will include: 1. Back up my data and the process I am following as a copy to my machine and, since this is a manageable dataset, to GitHub using command line and gitbash. 2. Read the data set into Python using Pandas' read_csv command. 3. Evaluate the data structure to better understand input data. 4. Naming the dataset as a the variable "churn_df" and subsequent useful slices of the dataframe as "df". 5. Examine potential misspellings, awkward variable naming & missing data. 6. Find outliers that may create or hide statistical significance using histograms. 7. Imputing records missing data with meaningful measures of central tendency (mean, median or mode) or simply remove outliers that are several standard deviations above the mean.

Most relevant to our decision making process is the dependent variable of "Bandwidth_GB_Year" (the average yearly amount of data used, in GB, per customer) which will be our continuous target variable. We need to train & then test our machine on our given dataset to develop a model that will give us an idea of how much data a customer may use given the amounts used by known customers given their respective data points for selected predictor variables.

In cleaning the data, we may discover relevance of the continuous predictor variables: * Children * Income * Outage_sec_perweek * Email * Contacts
* Yearly_equip_failure * Tenure (the number of months the customer has stayed with the provider)
* MonthlyCharge * Bandwidth_GB_Year

Likewise, we may discover relevance of the categorical predictor variables (all binary categorical with only two values, "Yes" or "No", except where noted): * Churn: Whether the customer discontinued service within the last month (yes, no) * Techie: Whether the customer considers themselves technically inclined (based on customer questionnaire when they signed up for services) (yes, no) * Contract: The contract term of the customer (month-to-month, one year, two year) * Port_modem: Whether the customer has a portable modem (yes, no) * Tablet: Whether the customer owns a tablet such as iPad, Surface, etc. (yes, no) * InternetService: Customer's internet service provider (DSL, fiber optic, None) * Phone: Whether the customer has a phone service (yes, no) * Multiple: Whether the customer has multiple lines (yes, no) * OnlineSecurity: Whether the customer has an online security add-on (yes, no) * OnlineBackup: Whether the customer has an online backup add-on (yes, no) * DeviceProtection: Whether the customer has device protection add-on (yes, no) * TechSupport: Whether the customer has a technical support add-on (yes, no) * StreamingTV: Whether the customer has streaming TV (yes, no) * StreamingMovies: Whether the customer has streaming movies (yes, no)

Finally, discrete ordinal predictor variables from the survey responses from customers regarding various customer service features may be relevant in the decision-making process. In the surveys, customers provided ordinal numerical data by rating 8 customer service factors on a scale

of 1 to 8 (1 = most important, 8 = least important):

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

1.1.7 C2. Summary Statistics:

Discuss the summary statistics, including the target variable and all predictor variables that you will need to gather from the data set to answer the research question.

1.1.8 C3. Steps to Prepare Data:

- Import dataset to Python dataframe.
- Rename columns/variables of survey to easily recognizable features (ex: "Item1" to "TimelyResponse").
- Get a description of dataframe, structure (columns & rows) & data types.
- View summary statistics.
- Drop less meaningful identifying (ex: "Customer_id") & demographic columns (ex: zip code) from dataframe.
- Check for records with missing data & impute missing data with meaningful measures of central tendency (mean, median or mode) or simply remove outliers that are several standard deviations above the mean.
- Create dummy variables in order to convert categorical, yes/no data points into numerical values.
- View univariate & bivariate visualizations.
- Finally, the prepared dataset will be extracted & provided as "churn_prepared.csv"

```
[ ]: # 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')

```

```

[:]: # 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

```

```

[:]: # Load data set into Pandas dataframe
churn_df = pd.read_csv('Data/churn_clean.csv')

# Rename last 8 survey columns for better description of variables
churn_df.rename(columns = {'Item1': 'TimelyResponse',
                           'Item2': 'Fixes',
                           'Item3': 'Replacements',
                           'Item4': 'Reliability',
                           'Item5': 'Options',
                           'Item6': 'Respectfulness',
                           'Item7': 'Courteous',
                           'Item8': 'Listening'},
                inplace=True)

```

```
[ ]: # Display Churn dataframe
churn_df
```

```
[ ]: CaseOrder Customer_id Interaction \
0          1      K409198 aa90260b-4141-4a24-8e36-b04ce1f4f77b
1          2      S120509 fb76459f-c047-4a9d-8af9-e0f7d4ac2524
2          3      K191035 344d114c-3736-4be5-98f7-c72c281e2d35
3          4        D90850 abfa2b40-2d43-4994-b15a-989b8c79e311
4          5      K662701 68a861fd-0d20-4e51-a587-8a90407ee574
...      ...      ...      ...
9995      9996      M324793 45deb5a2-ae04-4518-bf0b-c82db8dbe4a4
9996      9997      D861732 6e96b921-0c09-4993-bbda-a1ac6411061a
9997      9998      I243405 e8307ddf-9a01-4fff-bc59-4742e03fd24f
9998      9999      I641617 3775ccfc-0052-4107-81ae-9657f81ecd3
9999      10000      T38070 9de5fb6e-bd33-4995-aec8-f01d0172a499
```

```
UID City State \
0 e885b299883d4f9fb18e39c75155d990 Point Baker AK
1 f2de8bef964785f41a2959829830fb8a West Branch MI
2 f1784cfa9f6d92ae816197eb175d3c71 Yamhill OR
3 dc8a365077241bb5cd5ccd305136b05e Del Mar CA
4 aabb64a116e83fdc4befc1fbab1663f9 Needville TX
...      ...      ...      ...
9995 9499fb4de537af195d16d046b79fd20a Mount Holly VT
9996 c09a841117fa81b5c8e19afec2760104 Clarksville TN
9997 9c41f212d1e04dca84445019bbc9b41c Mobeetie TX
9998 3e1f269b40c235a1038863ecf6b7a0df Carrollton GA
9999 0ea683a03a3cd544aefe8388aab16176 Clarkesville GA
```

```
County Zip Lat Lng ... MonthlyCharge \
0 Prince of Wales-Hyder 99927 56.25100 -133.37571 ... 172.455519
1 Ogemaw 48661 44.32893 -84.24080 ... 242.632554
2 Yamhill 97148 45.35589 -123.24657 ... 159.947583
3 San Diego 92014 32.96687 -117.24798 ... 119.956840
4 Fort Bend 77461 29.38012 -95.80673 ... 149.948316
...      ...      ...      ...      ...
9995 Rutland 5758 43.43391 -72.78734 ... 159.979400
9996 Montgomery 37042 36.56907 -87.41694 ... 207.481100
9997 Wheeler 79061 35.52039 -100.44180 ... 169.974100
9998 Carroll 30117 33.58016 -85.13241 ... 252.624000
9999 Habersham 30523 34.70783 -83.53648 ... 217.484000
```

```
Bandwidth_GB_Year TimelyResponse Fixes Replacements Reliability \
0 904.536110 5 5 5 3
1 800.982766 3 4 3 3
2 2054.706961 4 4 2 4
3 2164.579412 4 4 4 2
```

4	271.493436	4	4	4	3
...
9995	6511.252601	3	2	3	3
9996	5695.951810	4	5	5	4
9997	4159.305799	4	4	4	4
9998	6468.456752	4	4	6	4
9999	5857.586167	2	2	3	3

	Options	Respectfulness	Courteous	Listening
0	4	4	3	4
1	4	3	4	4
2	4	3	3	3
3	5	4	3	3
4	4	4	4	5
...
9995	4	3	2	3
9996	4	5	2	5
9997	4	4	4	5
9998	3	3	5	4
9999	3	3	4	1

[10000 rows x 50 columns]

```
[ ]: # List of Dataframe Columns
df = churn_df.columns
print(df)
```

```
Index(['CaseOrder', 'Customer_id', 'Interaction', 'UID', 'City', 'State',
      'County', 'Zip', 'Lat', 'Lng', 'Population', 'Area', 'TimeZone', 'Job',
      'Children', 'Age', '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', 'TimelyResponse', 'Fixes', 'Replacements',
      'Reliability', 'Options', 'Respectfulness', 'Courteous', 'Listening'],
      dtype='object')
```

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

```
[ ]: (10000, 50)
```

```
[ ]: # Describe Churn dataset statistics
churn_df.describe()
```

```
[ ]:
      CaseOrder      Zip      Lat      Lng      Population \
count  10000.00000  10000.00000  10000.00000  10000.00000  10000.00000
```

mean	5000.50000	49153.319600	38.757567	-90.782536	9756.562400
std	2886.89568	27532.196108	5.437389	15.156142	14432.698671
min	1.00000	601.000000	17.966120	-171.688150	0.000000
25%	2500.75000	26292.500000	35.341828	-97.082813	738.000000
50%	5000.50000	48869.500000	39.395800	-87.918800	2910.500000
75%	7500.25000	71866.500000	42.106908	-80.088745	13168.000000
max	10000.00000	99929.000000	70.640660	-65.667850	111850.000000

	Children	Age	Income	Outage_sec_perweek	\
count	10000.0000	10000.000000	10000.000000	10000.000000	
mean	2.0877	53.078400	39806.926771	10.001848	
std	2.1472	20.698882	28199.916702	2.976019	
min	0.0000	18.000000	348.670000	0.099747	
25%	0.0000	35.000000	19224.717500	8.018214	
50%	1.0000	53.000000	33170.605000	10.018560	
75%	3.0000	71.000000	53246.170000	11.969485	
max	10.0000	89.000000	258900.700000	21.207230	

	Email	...	MonthlyCharge	Bandwidth_GB_Year	TimelyResponse	\
count	10000.000000	...	10000.000000	10000.000000	10000.000000	
mean	12.016000	...	172.624816	3392.341550	3.490800	
std	3.025898	...	42.943094	2185.294852	1.037797	
min	1.000000	...	79.978860	155.506715	1.000000	
25%	10.000000	...	139.979239	1236.470827	3.000000	
50%	12.000000	...	167.484700	3279.536903	3.000000	
75%	14.000000	...	200.734725	5586.141369	4.000000	
max	23.000000	...	290.160419	7158.981530	7.000000	

	Fixes	Replacements	Reliability	Options	Respectfulness	\
count	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000	
mean	3.505100	3.487000	3.497500	3.492900	3.497300	
std	1.034641	1.027977	1.025816	1.024819	1.033586	
min	1.000000	1.000000	1.000000	1.000000	1.000000	
25%	3.000000	3.000000	3.000000	3.000000	3.000000	
50%	4.000000	3.000000	3.000000	3.000000	3.000000	
75%	4.000000	4.000000	4.000000	4.000000	4.000000	
max	7.000000	8.000000	7.000000	7.000000	8.000000	

	Courteous	Listening
count	10000.000000	10000.000000
mean	3.509500	3.495600
std	1.028502	1.028633
min	1.000000	1.000000
25%	3.000000	3.000000
50%	4.000000	3.000000
75%	4.000000	4.000000
max	7.000000	8.000000

[8 rows x 23 columns]

```
[ ]: # Remove less meaningful demographic variables from statistics description
churn_df = churn_df.drop(columns=['CaseOrder', 'Customer_id', 'Interaction',
    → 'UID', 'City',
    → 'Population',
    → 'Area', 'TimeZone', 'Job', 'Marital'])
churn_df.describe()
```

```
[ ]:
      Children      Age      Income  Outage_sec_perweek  \
count  10000.0000  10000.000000  10000.000000  10000.000000
mean     2.0877    53.078400   39806.926771    10.001848
std     2.1472    20.698882   28199.916702     2.976019
min      0.0000    18.000000    348.670000     0.099747
25%      0.0000    35.000000   19224.717500     8.018214
50%      1.0000    53.000000   33170.605000    10.018560
75%      3.0000    71.000000   53246.170000    11.969485
max     10.0000    89.000000  258900.700000    21.207230

      Email      Contacts  Yearly_equip_failure      Tenure  \
count  10000.000000  10000.000000  10000.000000  10000.000000
mean     12.016000     0.994200         0.398000    34.526188
std      3.025898     0.988466         0.635953    26.443063
min      1.000000     0.000000         0.000000     1.000259
25%     10.000000     0.000000         0.000000     7.917694
50%     12.000000     1.000000         0.000000    35.430507
75%     14.000000     2.000000         1.000000    61.479795
max     23.000000     7.000000         6.000000    71.999280

      MonthlyCharge  Bandwidth_GB_Year  TimelyResponse      Fixes  \
count  10000.000000  10000.000000  10000.000000  10000.000000
mean     172.624816    3392.341550     3.490800     3.505100
std      42.943094    2185.294852     1.037797     1.034641
min      79.978860    155.506715     1.000000     1.000000
25%     139.979239    1236.470827     3.000000     3.000000
50%     167.484700    3279.536903     3.000000     4.000000
75%     200.734725    5586.141369     4.000000     4.000000
max     290.160419    7158.981530     7.000000     7.000000

      Replacements  Reliability      Options  Respectfulness  Courteous  \
count  10000.000000  10000.000000  10000.000000  10000.000000  10000.000000
mean      3.487000     3.497500     3.492900     3.497300     3.509500
std      1.027977     1.025816     1.024819     1.033586     1.028502
min      1.000000     1.000000     1.000000     1.000000     1.000000
25%      3.000000     3.000000     3.000000     3.000000     3.000000
50%      3.000000     3.000000     3.000000     3.000000     4.000000
```


75%	4.000000	4.000000	4.000000	4.000000	4.000000
max	8.000000	7.000000	7.000000	8.000000	7.000000

```

Listening
count    10000.000000
mean       3.495600
std        1.028633
min        1.000000
25%        3.000000
50%        3.000000
75%        4.000000
max        8.000000

```

```

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

```

Children	0
Age	0
Income	0
Gender	0
Churn	0
Outage_sec_perweek	0
Email	0
Contacts	0
Yearly_equip_failure	0
Techie	0
Contract	0
Port_modem	0
Tablet	0
InternetService	0
Phone	0
Multiple	0
OnlineSecurity	0
OnlineBackup	0
DeviceProtection	0
TechSupport	0
StreamingTV	0
StreamingMovies	0
PaperlessBilling	0
PaymentMethod	0
Tenure	0
MonthlyCharge	0
Bandwidth_GB_Year	0
TimelyResponse	0
Fixes	0
Replacements	0
Reliability	0

```
Options          0
Respectfulness   0
Courteous        0
Listening        0
dtype: int64
```

1.1.9 Dummy variable data preparation

Turn all yes/no into dummy variables a la Performance Lab Python.

```
[ ]: 
[ ]: 
[ ]:
```

1.1.10 C4. Visualizations:

Generate univariate and bivariate visualizations of the distributions of variables in the cleaned data set. Include the target variable in your bivariate visualizations.

```
[ ]: # Visualize missing values in dataset

# Install appropriate library
!pip install missingno

# Importing the libraries
import missingno as msno

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

```
findfont: Font family ['sans-serif'] not found. Falling back to DejaVu Sans.
findfont: Font family ['sans-serif'] not found. Falling back to DejaVu Sans.
```

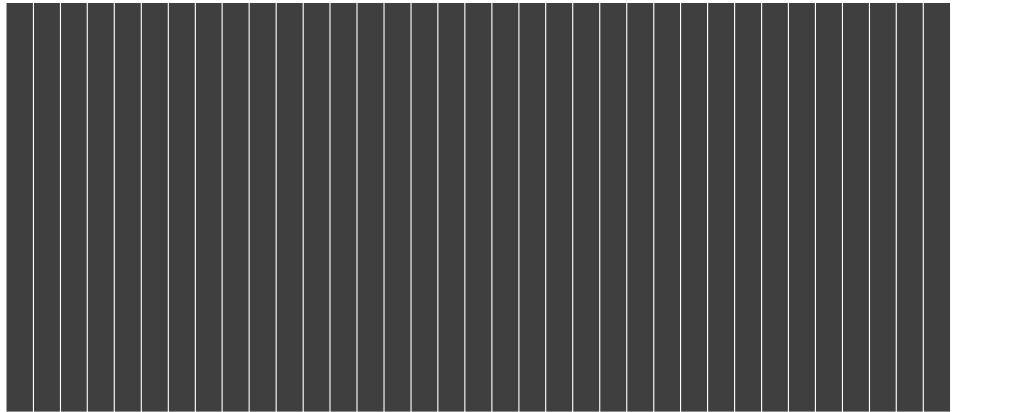
```
Requirement already satisfied: missingno in c:\users\vreed\anaconda3\lib\site-
packages (0.5.0)
Requirement already satisfied: numpy in c:\users\vreed\anaconda3\lib\site-
packages (from missingno) (1.18.1)
Requirement already satisfied: scipy in c:\users\vreed\anaconda3\lib\site-
packages (from missingno) (1.4.1)
Requirement already satisfied: matplotlib in c:\users\vreed\anaconda3\lib\site-
packages (from missingno) (3.1.3)
Requirement already satisfied: seaborn in c:\users\vreed\anaconda3\lib\site-
packages (from missingno) (0.10.0)
Requirement already satisfied: python-dateutil>=2.1 in
c:\users\vreed\anaconda3\lib\site-packages (from matplotlib->missingno) (2.8.1)
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.1 in
c:\users\vreed\anaconda3\lib\site-packages (from matplotlib->missingno) (2.4.6)
Requirement already satisfied: cycycler>=0.10 in c:\users\vreed\anaconda3\lib
```

```

\site-packages (from matplotlib->missingno) (0.10.0)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\users\vreed\anaconda3\lib
\site-packages (from matplotlib->missingno) (1.1.0)
Requirement already satisfied: pandas>=0.22.0 in c:\users\vreed\anaconda3\lib
\site-packages (from seaborn->missingno) (1.0.1)
Requirement already satisfied: six>=1.5 in c:\users\vreed\anaconda3\lib\site-
packages (from python-dateutil>=2.1->matplotlib->missingno) (1.14.0)
Requirement already satisfied: setuptools in c:\users\vreed\anaconda3\lib\site-
packages (from kiwisolver>=1.0.1->matplotlib->missingno) (45.2.0.post20200210)
Requirement already satisfied: pytz>=2017.2 in c:\users\vreed\anaconda3\lib
\site-packages (from pandas>=0.22.0->seaborn->missingno) (2019.3)

```

findfont: Font family ['sans-serif'] not found. Falling back to DejaVu Sans.



```

[:]: '''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())

```

```

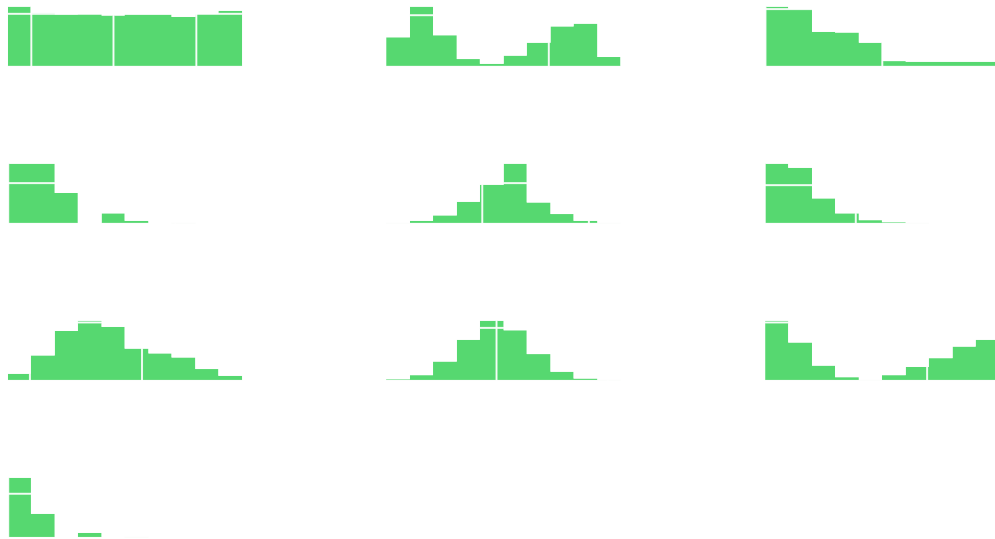
[:]: 'No need to impute an missing values as the dataset appears complete/cleaned'

```

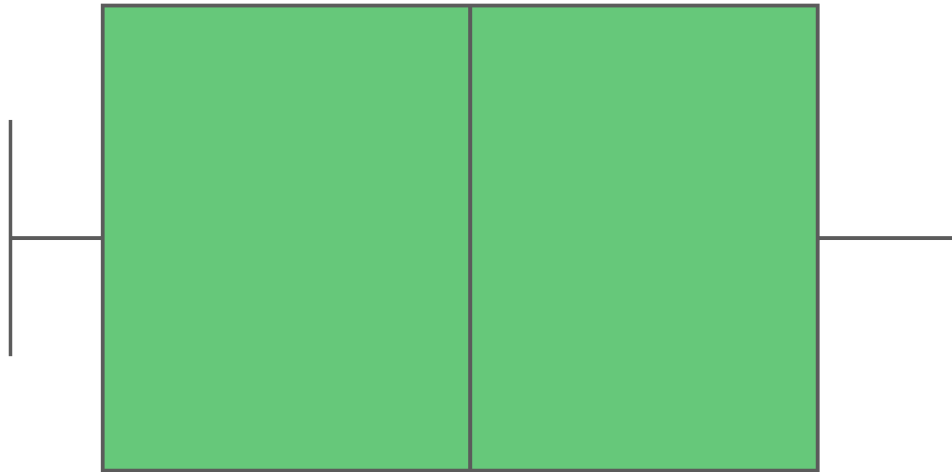
1.2 Univariate Statistics

```
[ ]: # Create histograms of continuous 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()
```

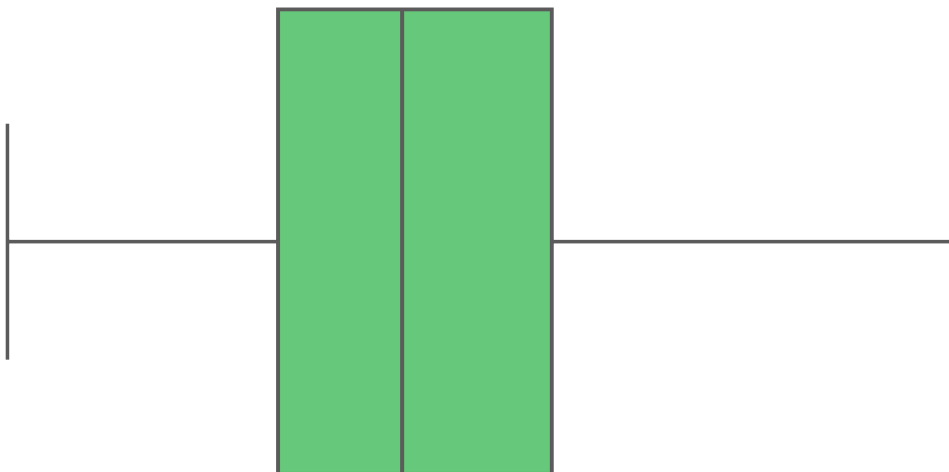
findfont: Font family ['sans-serif'] not found. Falling back to DejaVu Sans.
findfont: Font family ['sans-serif'] not found. Falling back to DejaVu Sans.



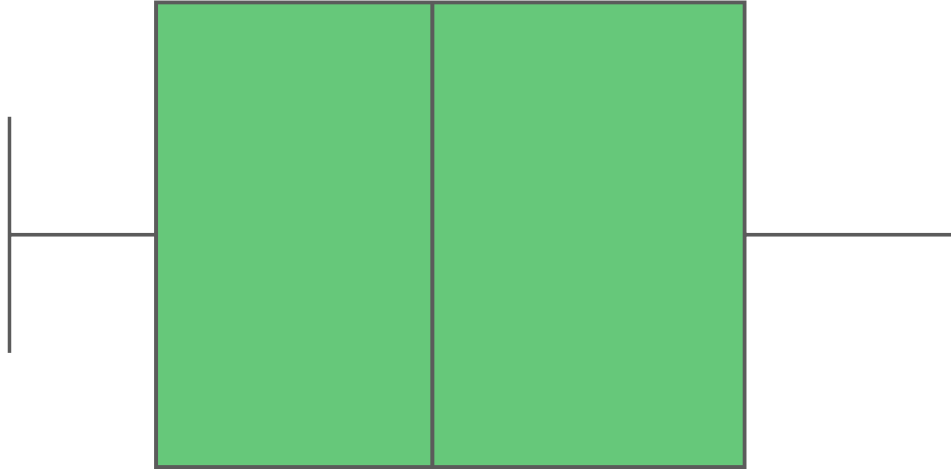
```
[ ]: # Create Seaborn boxplots for continuous variables
 sns.boxplot('Tenure', data = churn_df)
 plt.show()
```



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



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

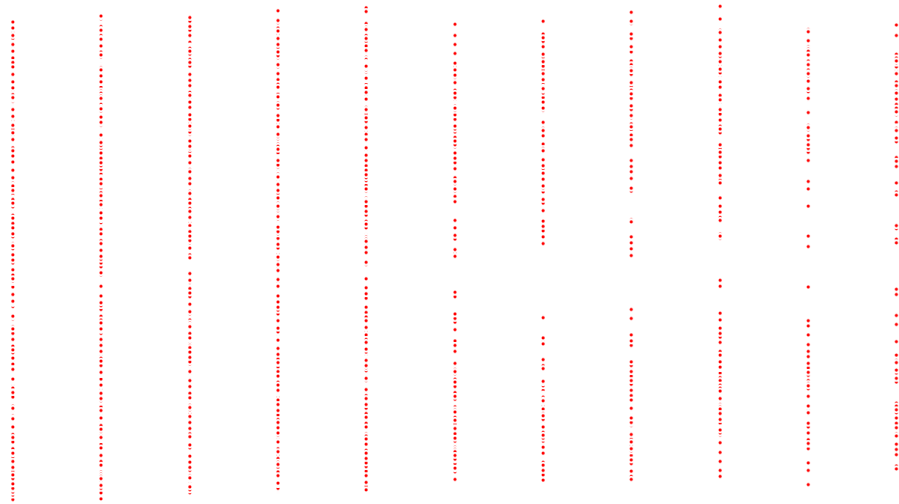


1.2.1 It appears that anomalies have been removed from the dataset present "churn_clean.csv" as there are no remaining outliers.

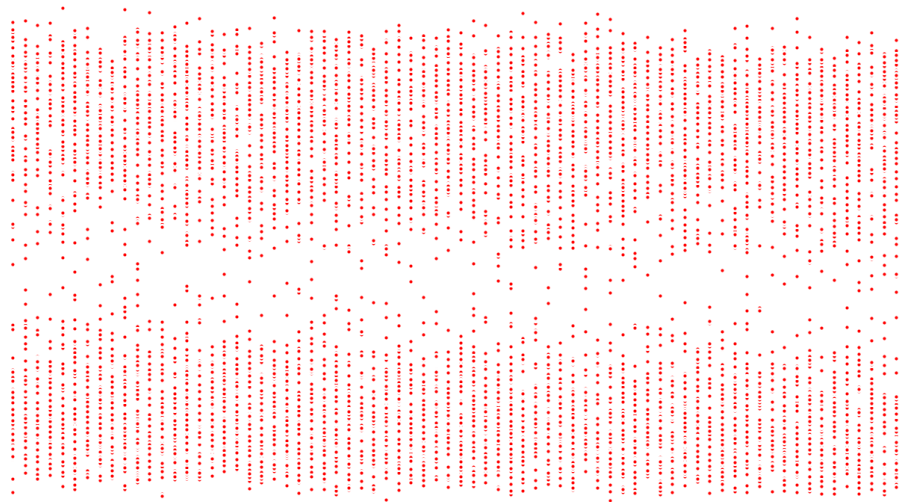
1.3 Bivariate Statistics

1.3.1 Let's run some scatterplots to get an idea of our linear relationships with our target variable of "Bandwidth_GB_Year" usage & some of the respective predictor variables.

```
[ ]: # 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();
```

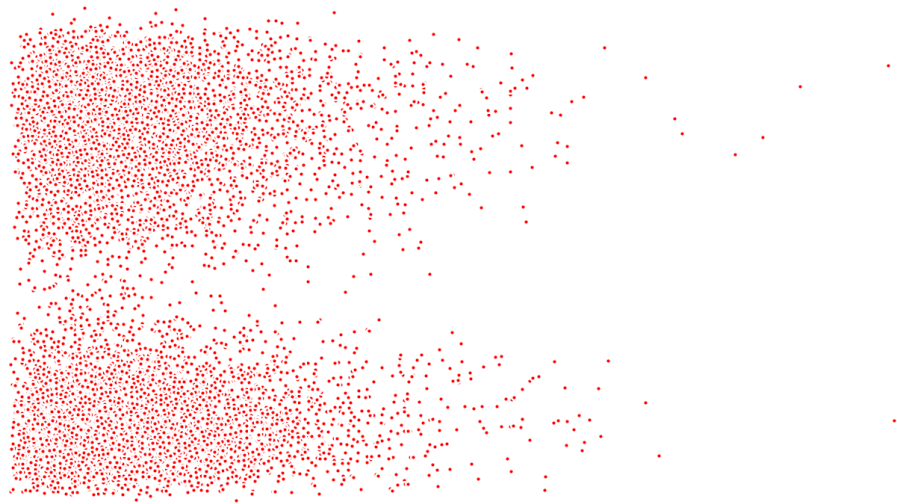


```
[ ]: sns.scatterplot(x=churn_df['Age'], y=churn_df['Bandwidth_GB_Year'], color='red')
plt.show();
```

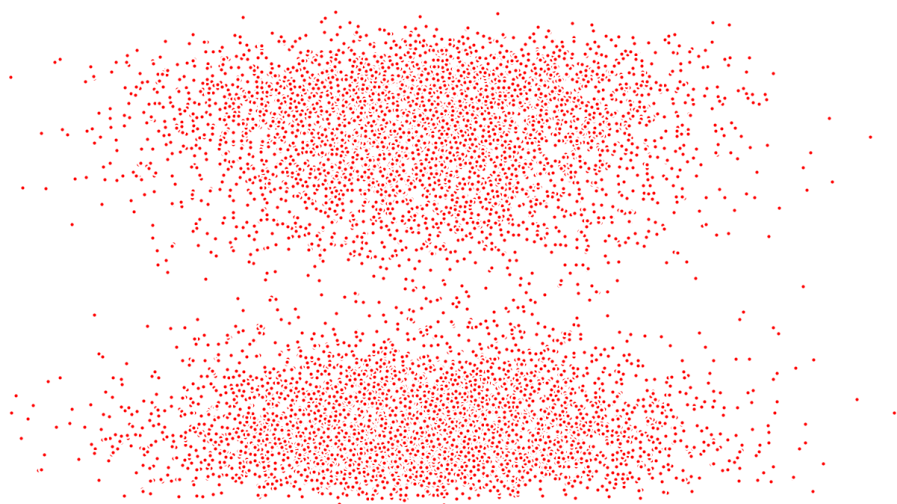


```
[ ]: sns.scatterplot(x=churn_df['Income'], y=churn_df['Bandwidth_GB_Year'],
                    color='red')
```

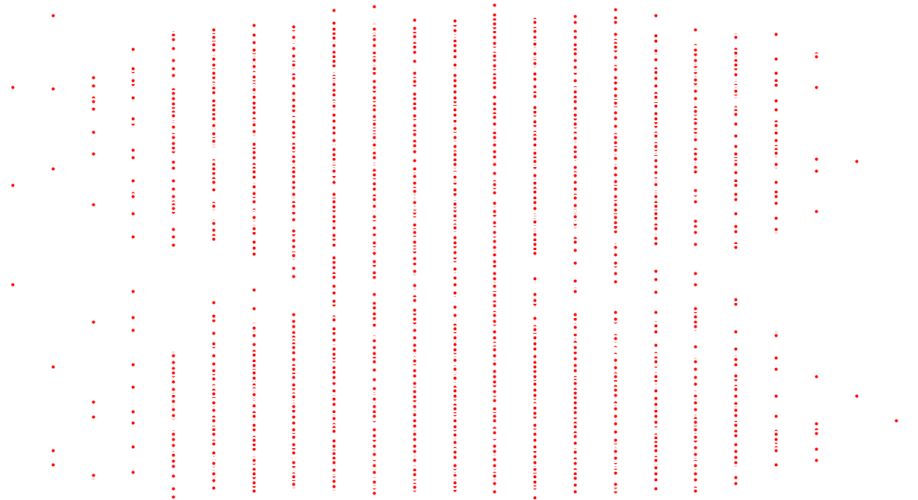
```
plt.show();
```



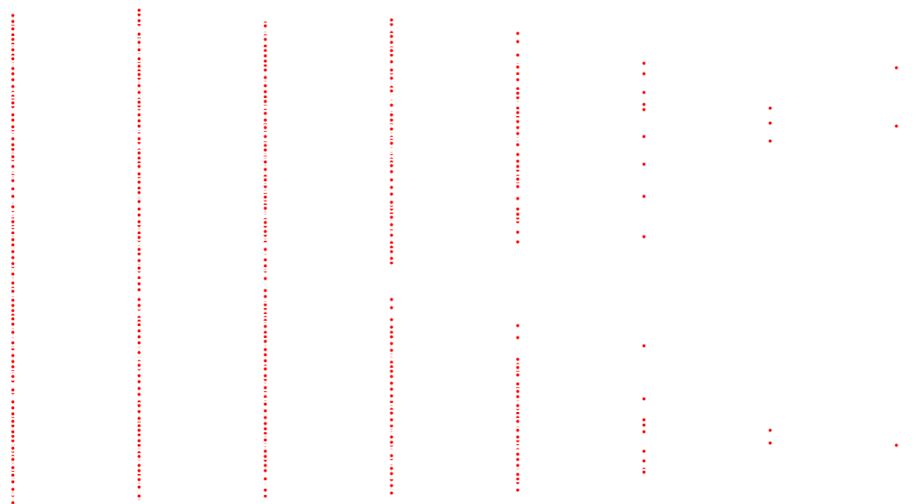
```
[ ]: sns.scatterplot(x=churn_df['Outage_sec_perweek'],  
    →y=churn_df['Bandwidth_GB_Year'], color='red')  
plt.show();
```



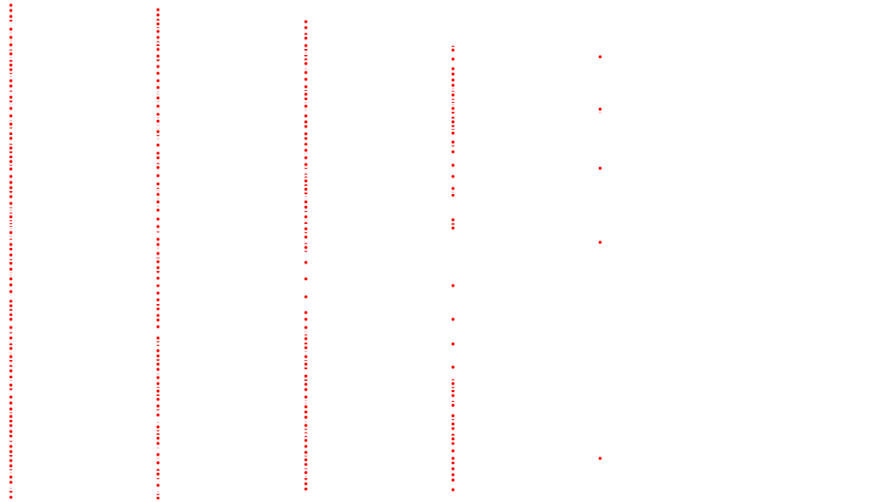

```
[ ]: sns.scatterplot(x=churn_df['Email'], y=churn_df['Bandwidth_GB_Year'],  
                    ↪color='red')  
plt.show();
```



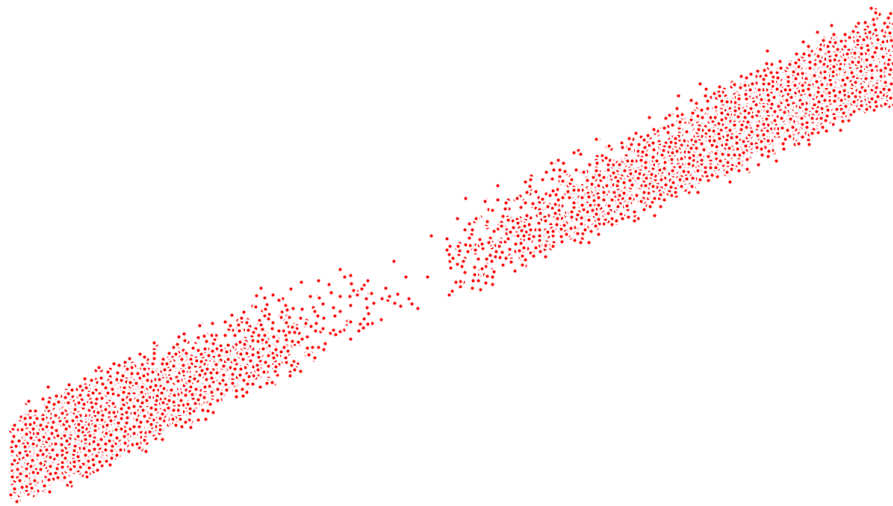
```
[ ]: sns.scatterplot(x=churn_df['Contacts'], y=churn_df['Bandwidth_GB_Year'],  
                    ↪color='red')  
plt.show();
```



```
[ ]: 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();
```

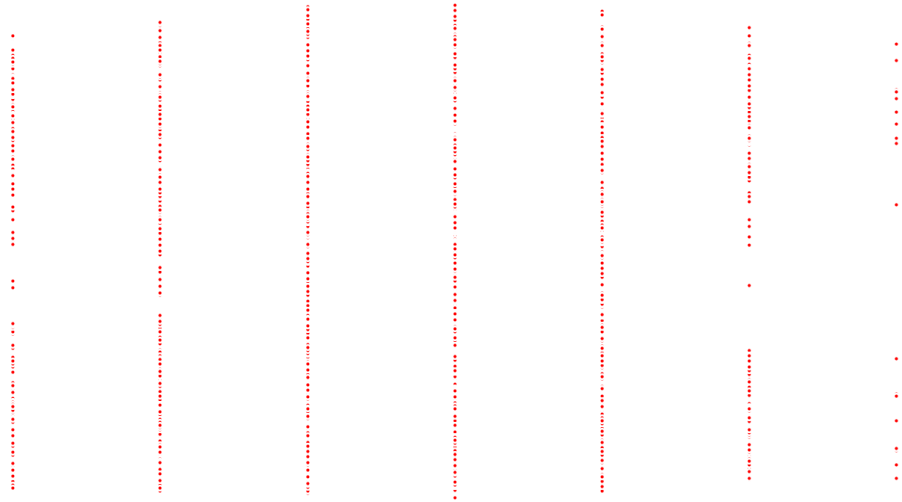


```
[ ]: sns.scatterplot(x=churn_df['MonthlyCharge'], y=churn_df['Bandwidth_GB_Year'],
    ↪color='red')
plt.show();
```

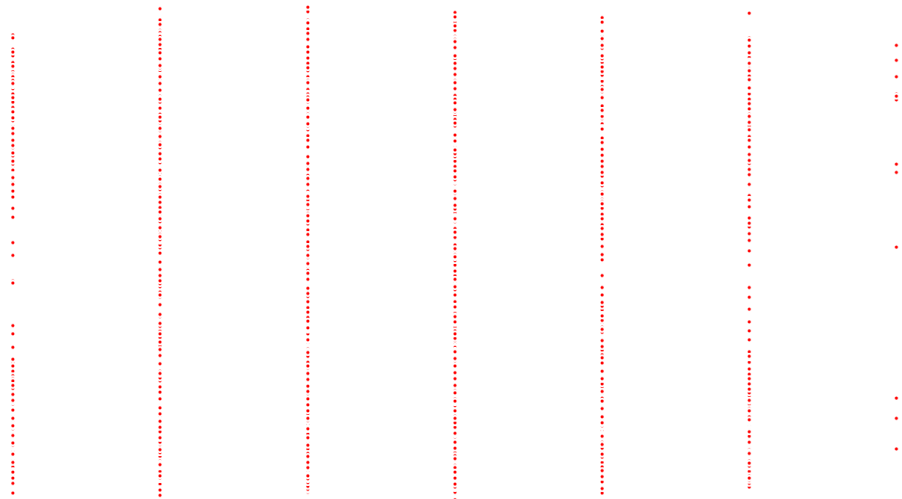


```
[ ]: sns.scatterplot(x=churn_df['TimelyResponse'], y=churn_df['Bandwidth_GB_Year'],
    ↪color='red')
```

```
plt.show();
```



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



1.3.2 C5. Prepared Dataset:

Provide a copy of the prepared data set.

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

1.3.3 Part IV: Model Comparison and Analysis

D. Compare an initial and a reduced multiple regression model by doing the following:

1. Construct an initial multiple regression model from all predictors that were identified in Part C2.
2. Justify a statistically based variable selection procedure and a model evaluation metric to reduce the initial model in a way that aligns with the research question.
3. Provide a reduced multiple regression model that includes both categorical and continuous variables.

Note: The output should include a screenshot of each model.

1.3.4 D1. Initial Model

Construct an initial multiple regression model from all predictors that were identified in Part C2.

```
[ ]: # Develop the initial estimated regression equation that could be used to
    →predict the Bandwidth_GB_Year,
# given the 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',
    →'MonthlyCharge',
    →'TimelyResponse', 'Fixes',
    →'Replacements',
    →'Reliability',
    →'Options',
    →'Respectfulness',
    →'Courteous',
    →'Listening',
```

```

fit()
print(lm_bandwidth.summary())

```

OLS Regression Results

```

Dep. Variable:      Bandwidth_GB_Year      R-squared:      0.989
Model:              OLS                    Adj. R-squared:  0.989
Method:             Least Squares          F-statistic:     5.329e+04
Date:               Mon, 12 Jul 2021        Prob (F-statistic): 0.00
Time:               10:42:36                Log-Likelihood:  -68489.
No. Observations:   10000                  AIC:             1.370e+05
Df Residuals:       9982                   BIC:             1.371e+05
Df Model:           17
Covariance Type:    nonrobust

```

	coef	std err	t	P> t	[0.025
--	------	---------	---	------	--------

0.975]

Children	30.9275	1.065	29.050	0.000	28.841
33.014					
Age	-3.3206	0.110	-30.065	0.000	-3.537
-3.104					
Income	9.976e-05	8.1e-05	1.231	0.218	-5.91e-05
0.000					
Outage_sec_perweek	-0.3501	0.768	-0.456	0.649	-1.856
1.156					
Email	-0.2792	0.755	-0.370	0.712	-1.759
1.201					
Contacts	2.9707	2.312	1.285	0.199	-1.562
7.503					
Yearly equip_failure	0.9080	3.593	0.253	0.801	-6.136
7.952					
Tenure	82.0113	0.086	948.882	0.000	81.842
82.181					
MonthlyCharge	3.2768	0.053	61.585	0.000	3.173
3.381					
TimelyResponse	-8.8961	3.271	-2.720	0.007	-15.308
-2.484					
Fixes	3.4660	3.064	1.131	0.258	-2.541
9.473					
Replacements	-0.1771	2.812	-0.063	0.950	-5.690
5.335					
Reliability	-0.2697	2.515	-0.107	0.915	-5.199
4.659					

Options	2.7199	2.611	1.042	0.298	-2.398
7.838					
Respectfulness	1.7157	2.689	0.638	0.523	-3.554
6.986					
Courteous	-1.3482	2.543	-0.530	0.596	-6.333
3.637					
Listening	5.7844	2.420	2.390	0.017	1.040
10.529					
intercept	95.8754	26.146	3.667	0.000	44.624
147.127					

Omnibus:	12280.983	Durbin-Watson:	1.979
Prob(Omnibus):	0.000	Jarque-Bera (JB):	968.853
Skew:	0.449	Prob(JB):	4.13e-211
Kurtosis:	1.768	Cond. No.	5.60e+05

Warnings:

- [1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
- [2] The condition number is large, 5.6e+05. This might indicate that there are strong multicollinearity or other numerical problems.

1.3.5 Initial Multiple Linear Regression Model

With 17 independent variables: $y = 104.85 + 30.86 * \text{Children} - 3.31 * \text{Age} + 0.00 * \text{Income} - 0.26 * \text{Outage_sec_perweek} - 0.31 * \text{Email} + 2.95 * \text{Contacts} + 0.67 * \text{Yearly_equip_failure} + 82.01 * \text{Tenure} + 3.28 * \text{MonthlyCharge} - 8.9 * \text{TimelyResponse} + 3.47 * \text{Fixes} - 0.18 * \text{Replacements} - 0.27 * \text{Reliability} + 2.72 * \text{Options} + 1.72 * \text{Respectfulness} - 1.35 * \text{Courteous} + 5.78 * \text{Listening}$

1.3.6 Based on an R2 value = 0.989. So, 99% of the variation is explained by this model. The condition number is large which might suggest strong multicollinearity. Apparently, we do not need all of these variables to explain the variance. So, let's run a heatmap for bivariate analysis & a principal component analysis in order to reduce variables.

1.3.7 D2. Justification of Model Reduction

Justify a statistically based variable selection procedure and a model evaluation metric to reduce the initial model in a way that aligns with the research question.

```
[ ]: # Create dataframe for heatmap bivariate analysis of correlation
churn_bivariate = churn_df[['Bandwidth_GB_Year', 'Children', 'Age', 'Income',
                             'Outage_sec_perweek', 'Email', 'Contacts',
                             'Yearly_equip_failure',
                             'Tenure', 'MonthlyCharge', 'TimelyResponse',
                             'Fixes',
                             'Replacements', 'Reliability', 'Options',
                             'Respectfulness',
                             'Courteous', 'Listening']]
```

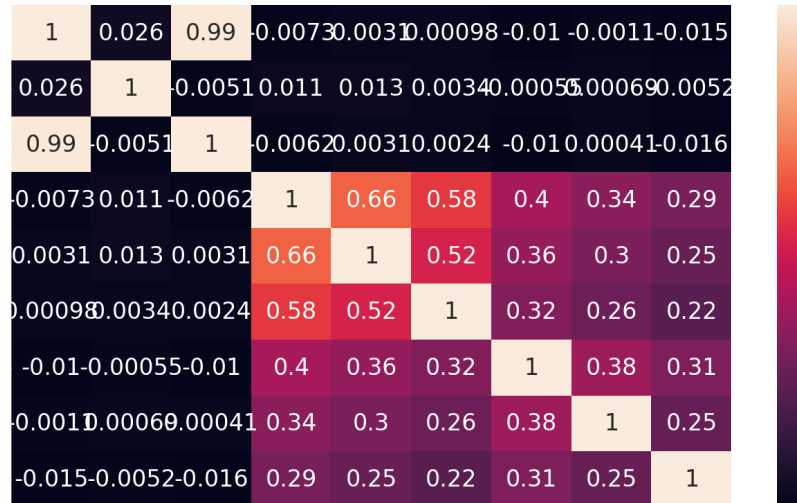
```
[ ]: # Run Seaborn heatmap
sns.heatmap(churn_bivariate.corr(), annot=True)
plt.show()
```



1.3.8 Alrighty, let's try that without some demographic, contacting-customer & options variables, basically purple or darker.

```
[ ]: churn_bivariate = churn_df[['Bandwidth_GB_Year', 'Children',
                                'Tenure', 'TimelyResponse', 'Fixes',
                                'Replacements', 'Respectfulness',
                                'Courteous', 'Listening']]

sns.heatmap(churn_bivariate.corr(), annot=True)
plt.show()
```

1.3.9 That looks a lot better.

Again, it appears that Tenure is the predictor for most of the variance. There is clearly a direct linear relationship between customer tenure with the telecom company & the amount of data (in GBs) that is being used. Let's run a multiple linear regression model on those variables with 0.50 or above & children because of its high coefficient (30.86) on the original OLS model. So, Y = bandwidth & then children, tenure, fixes, replacements.

1.3.10 D3. Reduced Multiple Regression Model

Provide a reduced multiple regression model that includes both categorical and continuous variables.

```
[ ]: # Run reduced OLS multiple regression
churn_df['intercept'] = 1
lm_bandwidth = sm.OLS(churn_df['Bandwidth_GB_Year'], churn_df[['Children', 'Tenure', 'Fixes', 'Replacements', 'intercept']]).fit()
print(lm_bandwidth.summary())
```

OLS Regression Results

```
=====
Dep. Variable:      Bandwidth_GB_Year      R-squared:      0.984
Model:              OLS      Adj. R-squared:      0.984
```

```

Method:                Least Squares    F-statistic:                1.537e+05
Date:                  Mon, 12 Jul 2021  Prob (F-statistic):          0.00
Time:                  10:45:41          Log-Likelihood:            -70407.
No. Observations:      10000          AIC:                        1.408e+05
Df Residuals:          9995          BIC:                        1.409e+05
Df Model:               4
Covariance Type:       nonrobust
=====
               coef      std err          t      P>|t|      [0.025      0.975]
-----
Children         31.1763      1.288      24.211      0.000      28.652      33.700
Tenure           81.9518      0.105     783.845      0.000      81.747      82.157
Fixes            1.0728      3.129       0.343      0.732      -5.061       7.206
Replacements     -3.6585      3.149      -1.162      0.245      -9.831       2.514
intercept       506.7695     11.949      42.413      0.000     483.348     530.191
=====
Omnibus:                 380.733    Durbin-Watson:                1.978
Prob(Omnibus):            0.000    Jarque-Bera (JB):             295.369
Skew:                     0.334    Prob(JB):                     7.27e-65
Kurtosis:                 2.488    Cond. No.                     191.
=====

```

Warnings:

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

1.3.11 Reduced Multiple Linear Regression Model

With 4 independent variables: $y = 497.78 + 31.18 * \text{Children} + 81.94 * \text{Tenure} + 1.07 * \text{Fixes} - 3.66 * \text{Replacements}$

1.3.12 Well, there it is. Removing all those other predictor variables & our model still explains 98% of the variance.

1.3.13 Part IV: E

E. Analyze the data set using your reduced multiple regression model by doing the following:

1. Explain your data analysis process by comparing the initial and reduced multiple regression models, including the following elements:
the logic of the variable selection technique
the model evaluation metric
a residual plot
2. Provide the output and any calculations of the analysis you performed, including the model's residual error.

Note: The output should include the predictions from the refined model you used to perform the analysis.

3. Provide the code used to support the implementation of the multiple regression models.

1.3.14 E1. Model Comparison

Explain your data analysis process by comparing the initial and reduced multiple regression models, including the following elements:

-
the logic of the variable selection technique

-
the model evaluation metric

-
a residual plot

Note: Verbatim from fasttrack description of analysis of Titanic dataset, "Since male is the dummy variable, being male reduces the log odds by 2.75 while a unit increase in age reduces log odds by 0.037."

1.3.15 E2. Output & Calculations

Provide the output and any calculations of the analysis you performed, including the model's residual error.

Note: The output should include the predictions from the refined model you used to perform the analysis.

1.3.16 E3. Code

All code for analysis include above.

1.3.17 Part V: Data Summary and Implications

F. Summarize your findings and assumptions by doing the following:

1. Discuss the results of your data analysis, including the following elements:
 - a regression equation for the reduced model
 - an interpretation of coefficients of the statistically significant variables of the model
 - the statistical and practical significance of the model
 - the limitations of the data analysis
2. Recommend a course of action based on your results.

1.3.18 F1. Results

Discuss the results of your data analysis, including the following elements:

-
a regression equation for the reduced model

-
an interpretation of coefficients of the statistically significant variables of the model

```

</li>
<li>
the statistical and practical significance of the model
</li>
<li>
the limitations of the data analysis
</li>

```

1.3.19 F2. Recommendations

Clearly with such a direct linear relationship between bandwidth used yearly & tenure with the telecom company it makes sense to suggest the company do everything within marketing & customer service capability to retain the customers gained as the longer they stay with the company the more bandwidth they tend to use.

1.3.20 Part VI: Demonstration

G. Provide a Panopto video recording that includes all of the following elements:

- a demonstration of the functionality of the code used for the analysis
- an identification of the version of the programming environment
- a comparison of the two multiple regression models you used in your analysis
- an interpretation of the coefficients.

1.3.21 G. Video

link

1.3.22 H. Sources for Third-Party Code

Kaggle. (2018, May 01). Bivariate plotting with pandas. Kaggle. <https://www.kaggle.com/residentmario/bivariate-plotting-with-pandas#>

Sree. (2020, October 26). Predict Customer Churn in Python. Towards Data Science. <https://towardsdatascience.com/predict-customer-churn-in-python-e8cd6d3aaa7>

Wikipedia. (2021, May 31). Bivariate Analysis. [https://en.wikipedia.org/wiki/Bivariate_analysis#:~:text=Bivariate](https://en.wikipedia.org/wiki/Bivariate_analysis#:~:text=Bivariate%20analysis,which%20allows%20you%20to%20visualize%20the%20relationship%20between%20two%20variables.)

1.3.23 I. Sources

Ahmad, A. K., Jafar, A & Aljoumaa, K. (2019, March 20). Customer churn prediction in telecom using machine learning in big data platform. Journal of Big Data. <https://journalofbigdata.springeropen.com/articles/10.1186/s40537-019-0191-6>

Altexsoft. (2019, March 27). Customer Churn Prediction Using Machine Learning: Main Approaches and Models. Altexsoft. <https://www.altexsoft.com/blog/business/customer-churn-prediction-for-subscription-businesses-using-machine-learning-main-approaches-and-models/>

Bruce, P., Bruce A. & Gedeck P. (2020). Practical Statistics for Data Scientists. O'Reilly. CBTNuggets. (2018, September 20). Why Data Scientists Love Python. <https://www.cbtnuggets.com/blog/technology/data/why-data-scientists-love-python>

Freedman, D. Pisani, R. & Purves, R. (2018). Statistics. W. W. Norton & Company, Inc.

Frohbose, F. (2020, November 24). Machine Learning Case Study: Telco Customer Churn Prediction. Towards Data Science. <https://towardsdatascience.com/machine-learning-case-study-telco-customer-churn-prediction-bc4be03c9e1d>

Griffiths, D. (2009). A Brain-Friendly Guide: Head First Statistics. O'Reilly.

Grus, J. (2015). Data Science from Scratch. O'Reilly.

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

McKinney, W. (2018). Python for Data Analysis. O'Reilly.

Rossant, C. (2018). IPython Interactive Computing & Visualization Cookbook, 2nd Edition. Packt Publishing.

Rossant, C. (2015). Learning IPython Interactive Computing & Visualization, 2nd Edition. Packt Publishing.

VanderPlas, J. (2017). Python Data Science Handbook. O'Reilly.

```
[ ]: !wget -nc https://raw.githubusercontent.com/brpy/colab-pdf/master/colab_pdf.py
from colab_pdf import colab_pdf
colab_pdf('D208_Performance_Assessment_NBM2_Task_1.ipynb')
```

```
--2021-07-12 16:53:14-- https://raw.githubusercontent.com/brpy/colab-
pdf/master/colab_pdf.py
```

```
Resolving raw.githubusercontent.com (raw.githubusercontent.com)...
185.199.108.133, 185.199.109.133, 185.199.110.133, ...
```

```
Connecting to raw.githubusercontent.com
```

```
(raw.githubusercontent.com)|185.199.108.133|:443... connected.
```

```
HTTP request sent, awaiting response... 200 OK
```

```
Length: 1864 (1.8K) [text/plain]
```

```
Saving to: colab_pdf.py
```

```
colab_pdf.py          100%[=====>]    1.82K  --.-KB/s    in 0s
```

```
2021-07-12 16:53:14 (30.5 MB/s) - colab_pdf.py saved [1864/1864]
```

```
Mounted at /content/drive/
```

```
WARNING: apt does not have a stable CLI interface. Use with caution in scripts.
```

```
WARNING: apt does not have a stable CLI interface. Use with caution in scripts.
```

```
Extracting templates from packages: 100%
```