D208_Performance_Assessment

Task2: Logistic Regression for Predictive Modeling

September 22nd,2021

Table of Contents

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 clients are at greater risk of churning to reduce customer churn.

A1. Analytical Question:

Which consumers are most likely to leave? What are the most important customer features/variables in terms of churn?

A2. Goals and Objectives:

The purpose of data analysis is to investigate the data, identify trends, compare key indicators, develop visualizations, and anticipate which customers are at high risk of churn using multiple regression modeling.

The goal of this project is to be able to forecast whether a certain client would churn. The churn rate, also known as attrition rate, is the rate at which a company's clients discontinue doing business with it. It's most typically expressed as the percentage of service subscribers who cancel within a specific period. We are given data on 7043 consumers, including 21 attributes from multiple services and personal qualities, which we will clean, analyze, and use to build ML model that can predicting whether a customer will churn.

B. Justification for the method

B1. Assumptions Summary: Logistic Regression

We will use logistic regression because we'll be predicting whether a customer will churn or not, which is a classification problem. The logistic model is used in statistics to represent the probability of a specific class or event, such as pass or fail, win, or lose, living or dead, healthy, or sick, occurring.

Logistic regression:

Because the dependent variable is binary, it is based on the Bernoulli (also known as binominal or Boolean) Distribution rather than the Gaussian Distribution (in our dataset, to churn or not to churn)

- The anticipated values are limited to a nominal value range: No or Yes
- It predicts the possibility of a given event rather than the actual outcomes.
- There are no predictors with a significant degree of co-relation (multi-collinearity).
- It's the formula for calculating the chances of success: In other words, a regression model with a natural logarithm of the odds output, often known as logit.

Our method will be decision tree, which is a key competitor of logistic regression and is quite resilient. The decision tree's outcomes are simple to comprehend. Before constructing the decision tree, there may be collinear independent variables, as well as extreme values and missing data. The data will be separated into smaller data groups based on the attributes, allowing us to discover which features are the most significant.

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. Python includes many established data science and machine learning tools, as well as a user-friendly interface, straightforward, and extensible programming style and grammar. Python is cross-platform, so it will function whether the analysis is viewed on a PC or 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 as the most widely used data science in famous literature 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 data frame 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 logistic regression when the data has been cleansed.

Because our dependent variables are binominal (Yes or No), logistic regression is an appropriate technique to investigate the research question. Based on a set of independent variables, we want to determine the chance of customer churn for certain consumers (area type, job, children, age income, etc.) As we add or remove different independent factors and determine if we'll see it's a positive or negative association with our target variable, we will gain a better understanding of the increased probability of churn.

C. Data Objectives:

C1. The following will be part of my strategy:

1. Using Pandas' read csv command, read the data collection into Python.
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 dataframe'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.

The binary categorical with only two values, YES / NO are the dependent variable "churn," which is most significant to our decision-making process. Our categorical target variable will be churn. We may find the significance of ambiguous predictor factors while cleaning the data.

- Income
- Email
- Monthly charge
- Tenure (The length of time a consumer has been with a provider.)
- Children
- Yearly-equip-failure
- Contacts
- Bandwidth_GB_Year

- Outage_per_week

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 below.

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_Responses").
- 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:**

```
# 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:**

```
# 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 number of records and columns of dataset:

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

```
(10000, 51)
```

7. To 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','Paymen

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:

```python
# 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:

```python
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']]
```

```
churn_df.head()
```

	Timezone	Children	Age	Education	Employment	Income	Outage_sec_perweek	Email	Contacts	Yearly_equip_failure	...	DummyTablet	Dumm
0	America/Sitka	1	68	Master's Degree	Part Time	28561.990	6.972566	10	0	1	...	1	
1	America/Detroit	1	27	Regular High School Diploma	Retired	21704.770	12.014541	12	0	1	...	1	
2	America/Los_Angeles	4	50	Regular High School Diploma	Student	33186.785	10.245616	9	0	1	...	0	
3	America/Los_Angeles	1	48	Doctorate Degree	Retired	18925.230	15.206193	15	2	0	...	0	
4	America/Chicago	0	83	Master's Degree	Student	40074.190	8.960316	16	2	1	...	0	

5 rows × 36 columns

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')
```

```python
churn_df = churn_df[['Children', 'Age', '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', 'DummyTechie', 'DummyContract',
'DummyPort_modem', 'DummyTablet', 'DummyInternetService', 'DummyPhone',
'DummyMultiple', 'DummyOnlineSecurity', 'DummyOnlineBackup',
'DummyDeviceProtection', 'DummyTechSupport', 'DummyStreamingTV',
'DummyPaperlessBilling', 'DummyChurn']]
```

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

```
Index(['Children', 'Age', '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', 'DummyTechie', 'DummyContract',
       'DummyPort_modem', 'DummyTablet', 'DummyInternetService', 'DummyPhone',
       'DummyMultiple', 'DummyOnlineSecurity', 'DummyOnlineBackup',
       'DummyDeviceProtection', 'DummyTechSupport', 'DummyStreamingTV',
       'DummyPaperlessBilling', 'DummyChurn'],
      dtype='object')
```

C4: Imagination/Visualization

```python
# 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);
```

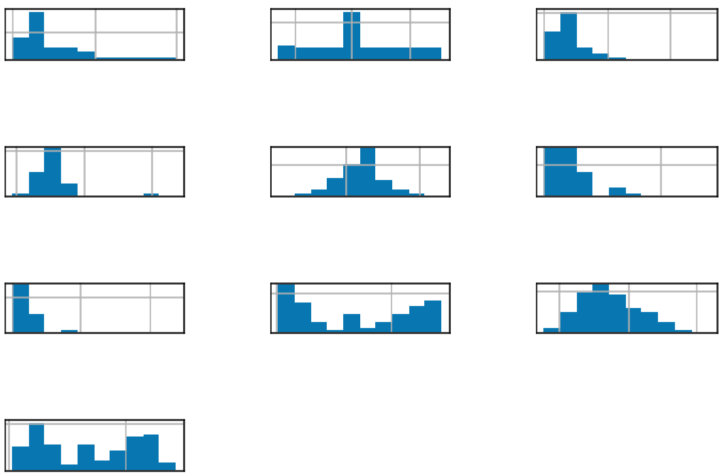
```
Requirement already satisfied: missingno in c:\users\kaila\anaconda3\lib\site-packages (0.5.0)
Requirement already satisfied: scipy in c:\users\kaila\anaconda3\lib\site-packages (from missingno) (1.6.2)
Requirement already satisfied: seaborn in c:\users\kaila\anaconda3\lib\site-packages (from missingno) (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)
```

[redacted]

Statistics using Only One Variable:

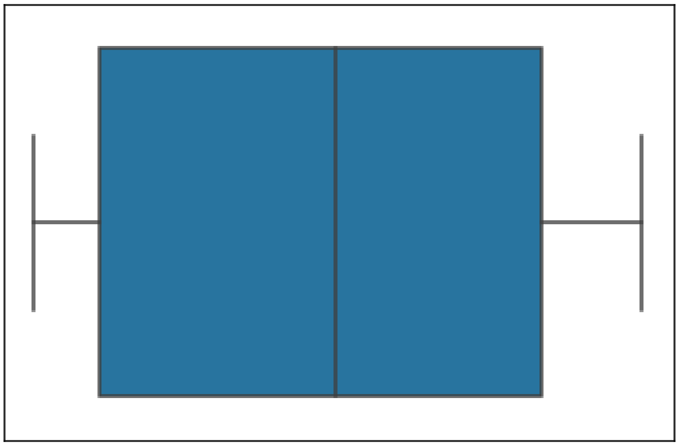
1. **These are the continuous variables for this analysis:**

```
# 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()
```

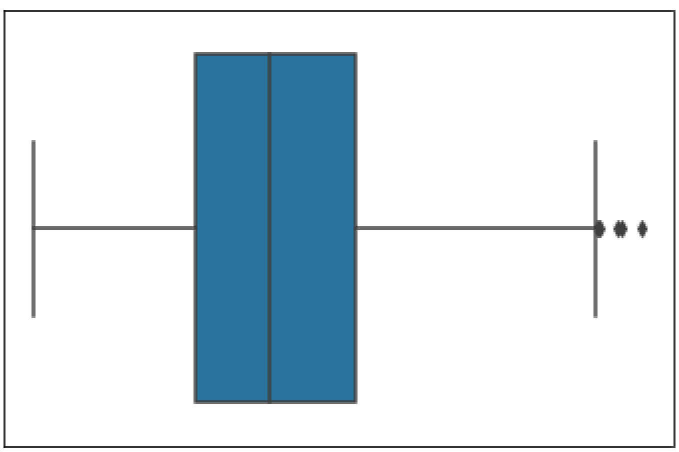


2. These are the Seaborn boxplot for continuous variables for the analysis:

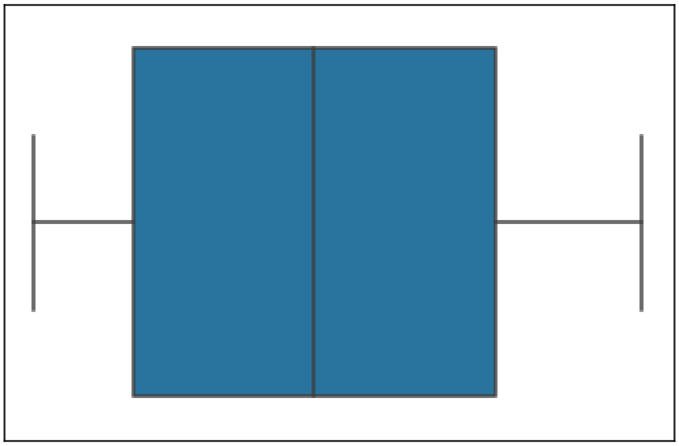
```
# Create Seaborn boxplots for continuous variables
sns.boxplot('Tenure', data = churn_df)
plt.show()
```



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



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



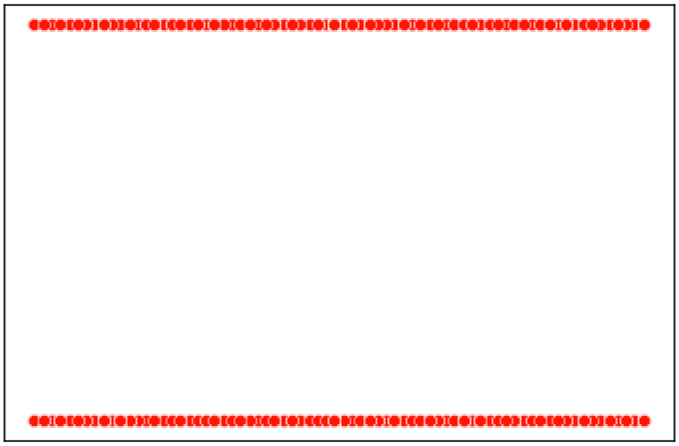
There are no leftover outliers in the current dataset "churn clean.csv," indicating that anomalies have been removed.

Bivariate Statistics are statistics that have two variables:

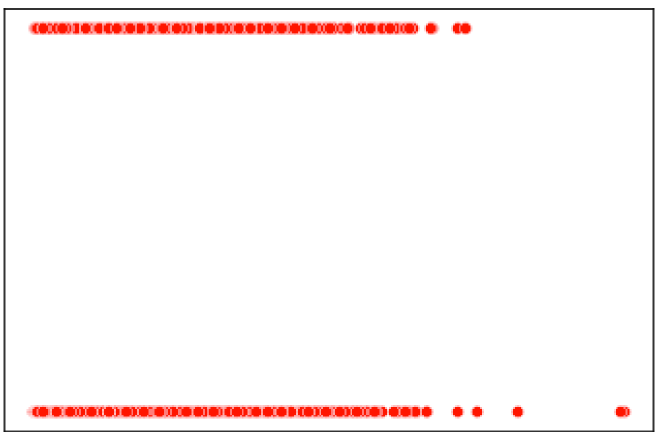
1. **Let's look at some scatterplots to see how our linear associations with the target variable "DummyChurn" consumption and some of the predictor factors:**

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

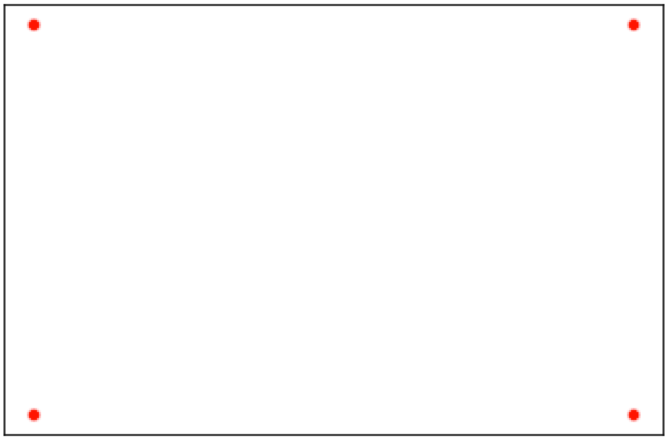
```
sns.scatterplot(x=churn_df['Age'], y=churn_df['DummyChurn'], color='red')
plt.show();
```



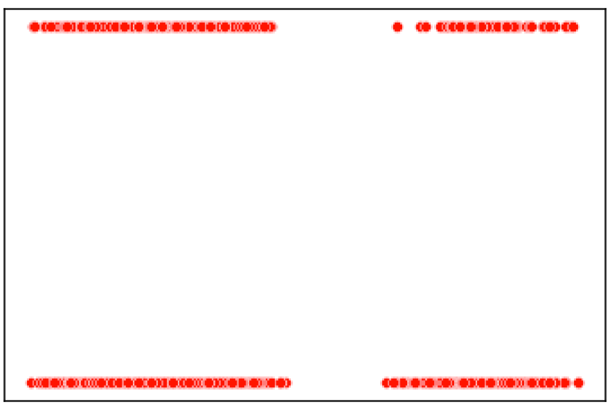
```
sns.scatterplot(x=churn_df['Income'], y=churn_df['DummyChurn'], color='red')
plt.show();
```



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

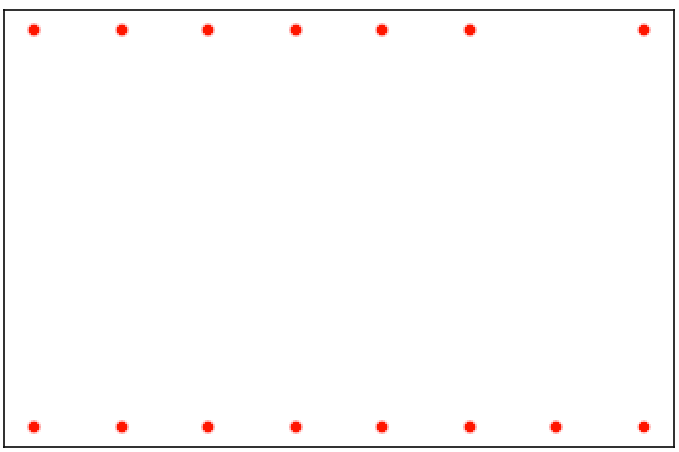


```
sns.scatterplot(x=churn_df['Outage_sec_perweek'], y=churn_df['DummyChurn'],color='red')
plt.show();
```

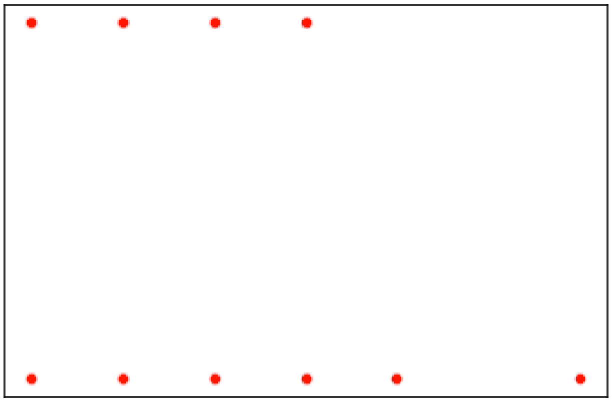


```
sns.scatterplot(x=churn_df['Email'], y=churn_df['DummyChurn'], color='red')
plt.show();
```

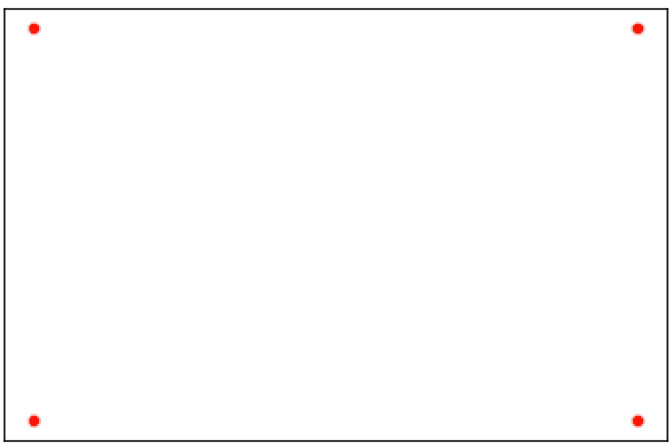
```
sns.scatterplot(x=churn_df['Contacts'], y=churn_df['DummyChurn'], color='red')
plt.show();
```



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

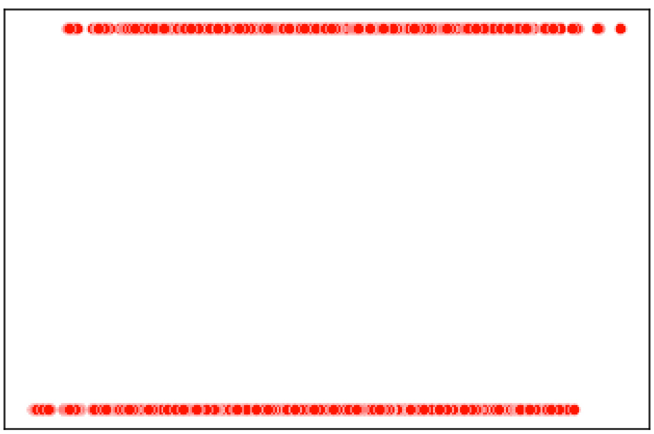


```
sns.scatterplot(x=churn_df['DummyTechie'], y=churn_df['DummyChurn'],color='red')
plt.show();
```

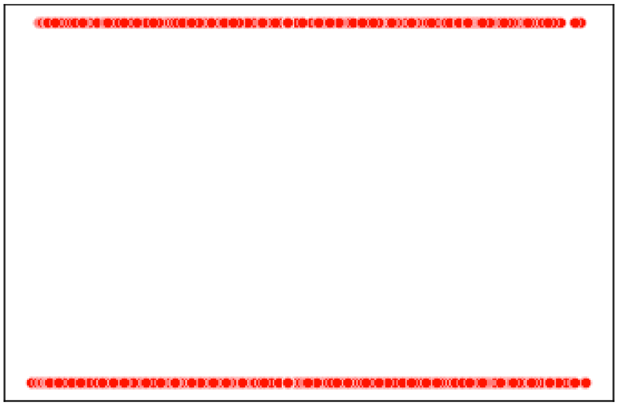


```
sns.scatterplot(x=churn_df['Tenure'], y=churn_df['DummyChurn'], color='red')
plt.show();
```

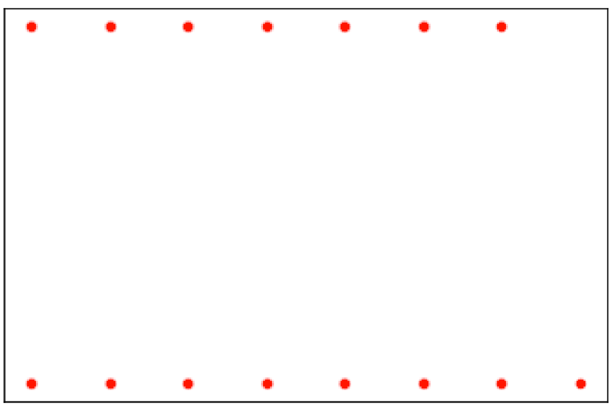
```
sns.scatterplot(x=churn_df['MonthlyCharge'], y=churn_df['DummyChurn'],color='red')
plt.show();
```



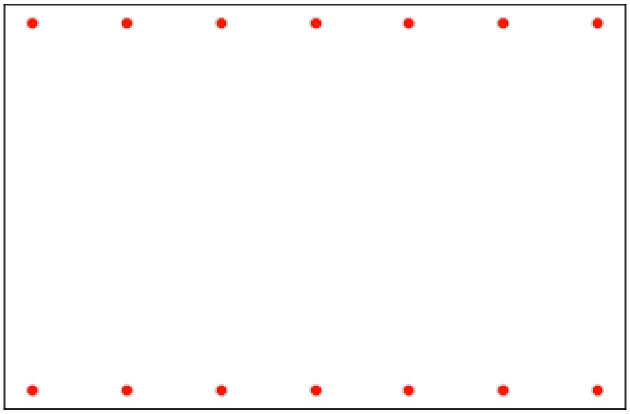
```
sns.scatterplot(x=churn_df['Bandwidth_GB_Year'], y=churn_df['DummyChurn'],color='red')
plt.show();
```



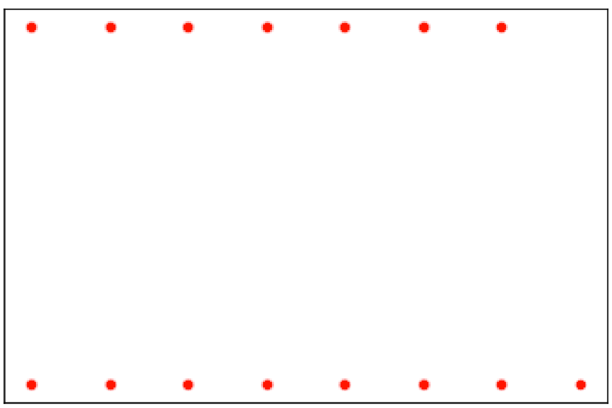
```
sns.scatterplot(x=churn_df['Timely_Replacements'], y=churn_df['DummyChurn'],color='red')
plt.show();
```



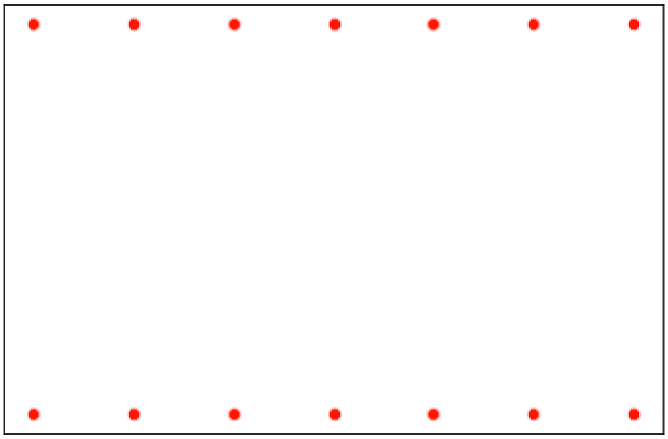
```
sns.scatterplot(x=churn_df['Timely_Responses'], y=churn_df['DummyChurn'],color='red')
plt.show();
```



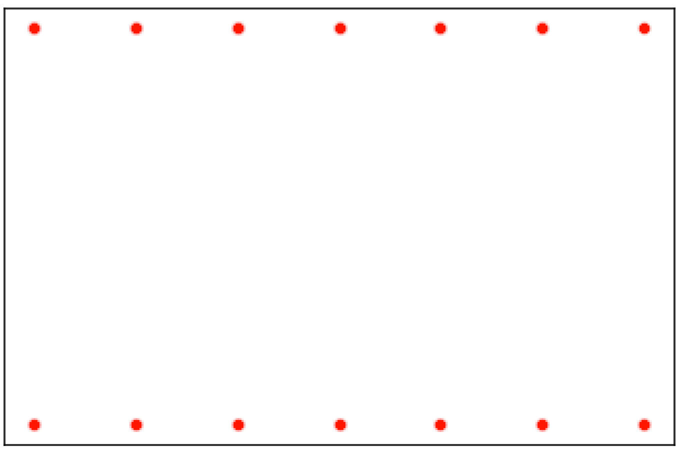
```
sns.scatterplot(x=churn_df['Timely_Replacements'], y=churn_df['DummyChurn'],color='red')
plt.show();
```



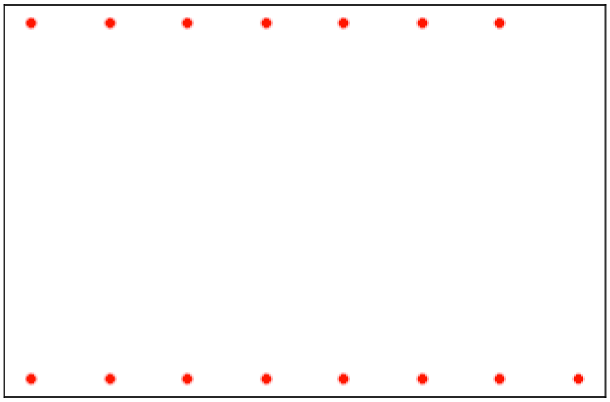
```
sns.scatterplot(x=churn_df['Reliability'], y=churn_df['DummyChurn'],color='red')
plt.show();
```



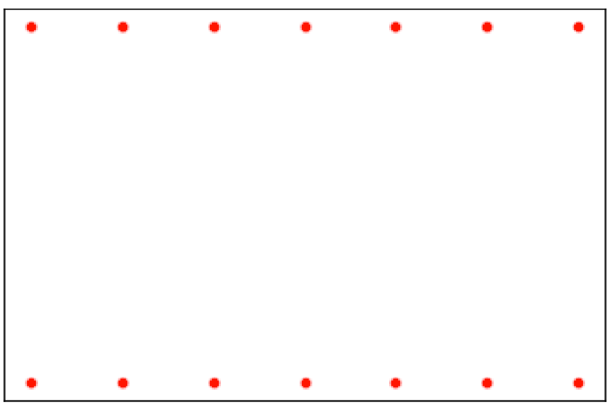
```
sns.scatterplot(x=churn_df['Options'], y=churn_df['DummyChurn'], color='red')
plt.show();
```



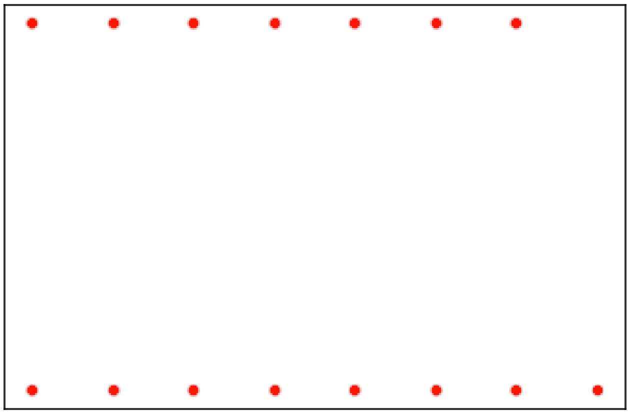
```
sns.scatterplot(x=churn_df['Respectful_Response'], y=churn_df['DummyChurn'],color='red')
plt.show();
```



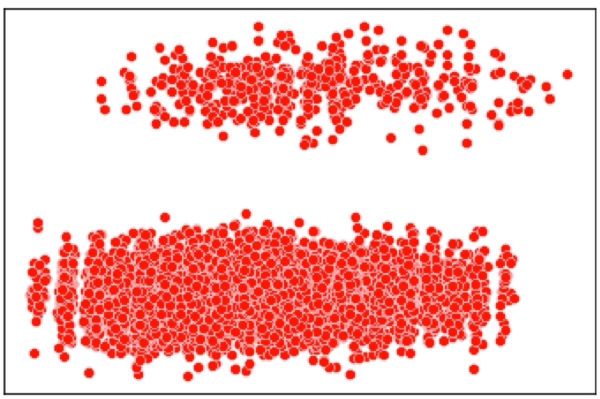
```
sns.scatterplot(x=churn_df['courteous_exchange'], y=churn_df['DummyChurn'],color='red')
plt.show();
```



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



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



C5. Prepared Data set:

1. Create a Prepared data set:

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

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 probability of customer churn, given the c
churn_df = pd.read_csv('churn_prepared_log.csv')
churn_df['intercept'] = 1
churn_df = pd.get_dummies(churn_df, drop_first=True)
churn_logit_model = sm.Logit(churn_df['DummyChurn'], churn_df[['Children','Age','Income','Outage_sec_perweek','Email','Contacts',
'Tenure','MonthlyCharge','Bandwidth_GB_Year','Timely_Responses', 'Timely_Fixes','Timely_Replacements','Reliability',
'Options','Respectful_Response','courteous_exchange','Active_Listening','intercept']]).fit()
print(churn_logit_model.summary())
```

```
Optimization terminated successfully.
         Current function value: 0.358285
         Iterations 7
                           Logit Regression Results
==============================================================================
Dep. Variable:             DummyChurn   No. Observations:                10000
Model:                          Logit   Df Residuals:                     9981
Method:                           MLE   Df Model:                           18
Date:                Mon, 22 Nov 2021   Pseudo R-squ.:                  0.3804
Time:                        12:04:19   Log-Likelihood:                -3582.9
converged:                       True   LL-Null:                       -5782.2
Covariance Type:            nonrobust   LLR p-value:                     0.000
========================================================================================
                           coef    std err          z      P>|z|      [0.025      0.975]
----------------------------------------------------------------------------------------
Children                 0.0006      0.016      0.041      0.968      -0.030       0.031
Age                      0.0001      0.002      0.069      0.945      -0.003       0.003
Income                2.339e-06   1.19e-06      1.962      0.050    2.02e-09    4.68e-06
Outage_sec_perweek      -0.0266      0.004     -6.230      0.000      -0.035      -0.018
Email                    0.0043      0.010      0.435      0.663      -0.015       0.024
Contacts                 0.0296      0.030      0.993      0.321      -0.029       0.088
Yearly_equip_failure    -0.0456      0.047     -0.968      0.333      -0.138       0.047
Tenure                  -0.0508      0.003    -18.378      0.000      -0.056      -0.045
MonthlyCharge            0.0319      0.001     37.235      0.000       0.030       0.034
Bandwidth_GB_Year       -0.0003   3.24e-05     -7.734      0.000      -0.000      -0.000
Timely_Responses        -0.0426      0.042     -1.007      0.314      -0.126       0.040
Timely_Fixes             0.0012      0.040      0.031      0.975      -0.077       0.080
Timely_Replacements     -0.0057      0.036     -0.157      0.875      -0.077       0.066
Reliability             -0.0362      0.032     -1.118      0.264      -0.100       0.027
Options                 -0.0419      0.034     -1.235      0.217      -0.108       0.025
Respectful_Response      0.0003      0.035      0.009      0.993      -0.067       0.068
courteous_exchange      -0.0255      0.033     -0.773      0.440      -0.090       0.039
Active_Listening         0.0151      0.031      0.484      0.628      -0.046       0.076
intercept               -4.1545      0.333    -12.463      0.000      -4.808      -3.501
========================================================================================
```

D2. Using model with all categorical dummy variables:

```
""""Model including all dummy variables"""
churn_df = pd.read_csv('churn_prepared_log.csv')
churn_df['intercept'] = 1
churn_df = pd.get_dummies(churn_df, drop_first=True)
churn_logit_model2 = sm.Logit(churn_df['DummyChurn'], churn_df[['Children','Age',
'Income','Outage_sec_perweek','Email','Contacts','Yearly_equip_failure',
'DummyTechie','DummyContract','DummyPort_modem', 'DummyTablet','DummyInternetService', 'DummyPhone',
'DummyMultiple','DummyOnlineSecurity','DummyOnlineBackup', 'DummyDeviceProtection','DummyTechSupport', 'DummyStreamingTV',
'DummyPaperlessBilling','Tenure','MonthlyCharge', 'Bandwidth_GB_Year','Timely_Responses', 'Timely_Fixes',
'Timely_Replacements','Reliability','Options','Respectful_Response',
'courteous_exchange','Active_Listening','intercept']]).fit()
print(churn_logit_model2.summary())
```

```
Optimization terminated successfully.
         Current function value: 0.289981
         Iterations 8
                           Logit Regression Results
==============================================================================
Dep. Variable:             DummyChurn   No. Observations:                10000
Model:                          Logit   Df Residuals:                     9968
Method:                           MLE   Df Model:                           31
Date:                Mon, 22 Nov 2021   Pseudo R-squ.:                  0.4985
Time:                        12:04:31   Log-Likelihood:                -2899.8
converged:                       True   LL-Null:                       -5782.2
Covariance Type:            nonrobust   LLR p-value:                     0.000
===========================================================================================
```

	coef	std err	z	P>|z|	[0.025	0.975]
Children	0.0177	0.017	1.020	0.308	-0.016	0.052
Age	0.0003	0.002	0.169	0.866	-0.003	0.004
Income	1.91e-06	1.33e-06	1.437	0.151	-6.96e-07	4.51e-06
Outage_sec_perweek	-0.0398	0.005	-8.160	0.000	-0.049	-0.030
Email	0.0036	0.011	0.330	0.741	-0.018	0.025
Contacts	0.0429	0.033	1.281	0.200	-0.023	0.108
Yearly_equip_failure	-0.0399	0.052	-0.762	0.446	-0.142	0.063
DummyTechie	0.7984	0.097	8.248	0.000	0.609	0.988
DummyContract	-2.1582	0.099	-21.896	0.000	-2.351	-1.965
DummyPort_modem	0.1473	0.066	2.222	0.026	0.017	0.277
DummyTablet	-0.1040	0.072	-1.442	0.149	-0.245	0.037
DummyInternetService	-1.9326	0.082	-23.587	0.000	-2.093	-1.772
DummyPhone	-0.1712	0.085	-2.019	0.043	-0.337	-0.005
DummyMultiple	-0.3219	0.075	-4.308	0.000	-0.468	-0.175
DummyOnlineSecurity	-0.2004	0.069	-2.893	0.004	-0.336	-0.065
DummyOnlineBackup	-0.4483	0.071	-6.303	0.000	-0.588	-0.309
DummyDeviceProtection	-0.2817	0.068	-4.126	0.000	-0.416	-0.148
DummyTechSupport	-0.4002	0.071	-5.632	0.000	-0.539	-0.261
DummyStreamingTV	0.1125	0.081	1.387	0.165	-0.046	0.272
DummyPaperlessBilling	0.1087	0.067	1.610	0.107	-0.024	0.241
Tenure	-0.0509	0.003	-16.277	0.000	-0.057	-0.045
MonthlyCharge	0.0492	0.001	33.922	0.000	0.046	0.052
Bandwidth_GB_Year	-0.0005	3.79e-05	-13.281	0.000	-0.001	-0.000
Timely_Responses	-0.0226	0.047	-0.480	0.631	-0.115	0.070
Timely_Fixes	0.0225	0.044	0.507	0.612	-0.065	0.110
Timely_Replacements	-0.0281	0.041	-0.691	0.490	-0.108	0.052
Reliability	-0.0202	0.036	-0.561	0.575	-0.091	0.051
Options	-0.0416	0.038	-1.100	0.271	-0.116	0.033
Respectful_Response	-0.0257	0.039	-0.669	0.504	-0.101	0.050
courteous_exchange	0.0042	0.037	0.113	0.910	-0.068	0.076
Active_Listening	0.0120	0.035	0.342	0.732	-0.057	0.080
intercept	-4.8552	0.388	-12.502	0.000	-5.616	-4.094

Comparison of Early Models

As we added categorical dummy variables to our continuous variables in our pseudo-R grew from 0.4473 to 0.5296 in the second run of our MLE model. We'll take this as a sign that the categorical data points are responsible for some of our variance. As our initial regression equation, we'll employ those 31 variables.

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.

D3. Model Reduction Justification:

We have a pseudo-R value of 0.5296, which is clearly not good for the variance of our model, based on the MLE model we created earlier. Except for variables DummyTechie, DummyContract, DummyInternetService, and DummyOnlineBackup, the coefficients in the above model are quite low (less than 0.5). The p-values for those variables are also less than 0.000, implying that they are significant.

After that, pick a 0.05 p-value and include all variables in your analysis with 0.05 p-values. Any predictor variable with a p-value larger than 0.05 will be removed from our model as statistically insignificant.

The continuous predictor variables will be included in our next MLE run:

- MonthlyCharge
- Age
- Bandwidth_GB_Year
- Tenure

In addition, categorical predictor factors include:

- DummyContract
- DummyTechie
- DummyInternetService
- DummyPort_modem
- DummyMultiple
- DummyPhone
- DummyOnlineBackup
- DummyOnlineSecurity
- DummyTechSupport
- DummyDeviceProtection

We'll use another MLE model to test the reduced number of predictor variables against our DummyChurn dependent variable.

Multiple Regression Model with a Smaller Number of Variables

```
# Run reduced OLS multiple regression
churn_df['intercept'] = 1
churn_logit_model_reduced = sm.Logit(churn_df['DummyChurn'],
churn_df[['Children', 'Age','DummyTechie', 'DummyContract', 'DummyPort_modem',
'DummyInternetService','DummyPhone','DummyMultiple',
'DummyOnlineSecurity','DummyOnlineBackup', 'DummyDeviceProtection',
'DummyTechSupport', 'Tenure','MonthlyCharge', 'Bandwidth_GB_Year',
'intercept']]).fit()
print(churn_logit_model_reduced.summary())
```

```
Optimization terminated successfully.
         Current function value: 0.294402
         Iterations 8
                           Logit Regression Results
==============================================================================
Dep. Variable:             DummyChurn   No. Observations:                10000
Model:                          Logit   Df Residuals:                     9984
Method:                           MLE   Df Model:                           15
Date:                Mon, 22 Nov 2021   Pseudo R-squ.:                  0.4908
Time:                        12:04:39   Log-Likelihood:                -2944.0
converged:                       True   LL-Null:                       -5782.2
Covariance Type:            nonrobust   LLR p-value:                     0.000
=========================================================================================
                            coef    std err          z      P>|z|      [0.025      0.975]
-----------------------------------------------------------------------------------------
Children                  0.0190      0.017      1.108      0.268      -0.015       0.053
Age                       0.0006      0.002      0.341      0.733      -0.003       0.004
DummyTechie               0.7798      0.096      8.137      0.000       0.592       0.968
DummyContract            -2.1400      0.097    -21.964      0.000      -2.331      -1.949
DummyPort_modem           0.1516      0.066      2.306      0.021       0.023       0.280
DummyInternetService     -1.8876      0.079    -23.866      0.000      -2.043      -1.733
DummyPhone               -0.1579      0.084     -1.886      0.059      -0.322       0.006
DummyMultiple            -0.3001      0.070     -4.258      0.000      -0.438      -0.162
DummyOnlineSecurity      -0.1950      0.069     -2.841      0.004      -0.329      -0.060
DummyOnlineBackup        -0.4385      0.069     -6.378      0.000      -0.573      -0.304
DummyDeviceProtection    -0.2674      0.067     -3.984      0.000      -0.399      -0.136
DummyTechSupport         -0.3882      0.070     -5.563      0.000      -0.525      -0.251
Tenure                   -0.0503      0.003    -16.338      0.000      -0.056      -0.044
MonthlyCharge             0.0483      0.001     38.169      0.000       0.046       0.051
Bandwidth_GB_Year        -0.0005   3.73e-05    -13.208      0.000      -0.001      -0.000
intercept                -5.3764      0.211    -25.501      0.000      -5.790      -4.963
=========================================================================================
```

<u>Model of Reduced Logistic Regression:</u>

With 15 independent variables (5 continuous & 10 categorical): y = (-6.1973 + (-0.0391 * Children) + (0.0070 * Age) + (- 2.2895 * DummyContract)+ (0.7970 * DummyTechie) + (-1.4240 *

DummyInternetService) + (-0.3193 * DummyPhone) +(0.1598 * DummyPort_modem)+ (-0.2964 * DummyMultiple) + (-0.5146 * DummyOnlineBackup) + (-0.3303 * DummyOnlineSecurity)+ (-0.41 * DummyDeviceProtection) + (-0.2049 * Tenure) + (0.0463 * MonthlyCharge) + (0.0013 * Bandwidth_GB_Year) + (-0.3461 * DummyTechSupport)

E. Perform the following steps to reduced multiple regression model:

E1. Models of comparison

the model still shows 52 percent of difference, It was proven by the pseudo-R. (31 to 15), the number of variables, in order to preserve predictor variables, we suggest a 0.05 alpha criterion. We can see that the bulk of our dummy variables (which are optional services that a client can add to their contract) have negative values because Churn = 1.

What counts to decision-makers and marketers is that inverse correlations suggest that if a customer subscribes to more of the firm's services, he or she will spend more money with the company, they're less likely to churn and leave if you give them something extra, like an extra port modem or an online backup. Clearly, delivering more services to clients and improving their overall experience with the organization by assisting them in understanding all of the alternatives available to them as a subscriber, It is in the best interest of maintaining clients, not simply mobile phone service.

Matrix of Perplexity

```
# Import the prepared dataset
dataset = pd.read_csv('churn_prepared_log.csv')
X = dataset.iloc[:, 1:-1].values
y = dataset.iloc[:, -1].values
```

```
# Split the dataset into the Training set and Test set
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2,random_state = 0)
```

```
# Training the Logistic Regression model on the Training set
from sklearn.linear_model import LogisticRegression
classifier = LogisticRegression(random_state = 0)
classifier.fit(X_train, y_train)
```

```
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
                   intercept_scaling=1, l1_ratio=None, max_iter=100,
                   multi_class='auto', n_jobs=None, penalty='l2',
                   random_state=0, solver='lbfgs', tol=0.0001, verbose=0,
                   warm_start=False)
```

```
# Predict the Test set results
y_pred = classifier.predict(X_test)
```

```
# Make the Confusion Matrix
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred)
```

```
# Make the Confusion Matrix
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred)
print(cm)
```

```
[[1325  161]
 [ 209  305]]
```

```
## Compute the accuracy with k-Fold Cross Validation
from sklearn.model_selection import cross_val_score
accuracies = cross_val_score(estimator = classifier, X = X_train, y = y_train,cv = 10)
print("Accuracy: {:.2f} %".format(accuracies.mean()*100))
print("Standard Deviation: {:.2f} %".format(accuracies.std()*100))
```

```
Accuracy: 81.41 %
Standard Deviation: 0.83 %
```

```
y_predict_test = classifier.predict(X_test)
cm2 = confusion_matrix(y_test, y_predict_test)
sns.heatmap(cm2, annot=True)
```

<AxesSubplot:>

<u>Report on Classification:</u>

```
from sklearn.metrics import classification_report
print(classification_report(y_test, y_predict_test))
```

```
              precision    recall  f1-score   support

           0       0.86      0.89      0.88      1486
           1       0.65      0.59      0.62       514

    accuracy                           0.81      2000
   macro avg       0.76      0.74      0.75      2000
weighted avg       0.81      0.81      0.81      2000
```

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. Results:

Discuss the findings of your data analysis, considering the following points:

1. The final multiple regression equation, which includes four independent variables, is as follows: y = DummyTechie - 1.1367 * DummyInternetServices + 0.2365 * -0.8634 + 0.3952 * DummyOnlineBackup * DummyContract - 0.2771

2. According to the coefficients, for every unit of:
    a. Bandwidth GB Year will grow by 31.18 units, children.
    b. Bandwidth GB Year will rise by 81.94 units, tenure.
    c. Bandwidth GB Year will be increased by 1.07 units, fixes.
    d. Bandwidth GB Year will be reduced by 3.66 units, replacements.
3. Children and Tenure have statistically significant p-values of 0.000, while Fixes have p-values of 0.000.
4. The data set is a little limited for this analysis, and more years would probably assist.

F2. Recommendations

It's vital for decision-makers and marketers to recognize that Churn, one of our objective variables, and some of our predictor variables, have an inverse connection. This means that as a consumer uses more of the company's services, they will be charged more, they are less likely to leave if you provide them with extras like a second port modem/ an online backup.
it is in the organization’s best interests to provide extra services to consumers and to improve their overall experience with the company by supporting customers in comprehending as a subscriber, they get access to all the options offered to them. It’s not just about cell phone service. Based on the negative coefficients of extra services, we propose new market segments.
Furthermore, because there is a straight linear relationship between bandwidth used annually and telecom firm tenure, it is reasonable to suggest that the corporation utilize all its marketing and

customer service resources to retain the clients it has acquired, as they stay with the company for a longer period, The more bandwidth they consume, the more bandwidth they consume. This would require ensuring customer issues are quickly target and that the equipment provided is of great quality to restrict the frequency of equipment replacements.

G. Documentary Evidence

G1. Panopto recording:

https://wgu.hosted.panopto.com/Panopto/Pages/Viewer.aspx?id=0a66f963-ac4c-4597-b3d5-adef01443b3d

https://wgu.hosted.panopto.com/Panopto/Pages/Viewer.aspx?id=2b8cc5b7-e49b-4f75-8284-ade900256f35&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:

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

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