Evaluating a Logistic Regression Model for Churn Prediction to Identify Key Drivers of Customer Attrition

Joanne Senoren

Master of Science, Data Analytics

Table of Contents

Part I: Research Question

A1. Research Question

Customer churn rate, the rate at which consumers cancel subscriptions or services, is a significant factor in company profitability (Ahmad et al., 2019). Regardless of industry, any membership-based company benefits from minimizing its churn rate because obtaining new customers is more costly than maintaining them (Ahmad et al., 2019). Based on this information, the paper seeks to answer the following research question: *"What factors in this telecommunications dataset can predict the probability of customers who are more likely to churn?"*

By targeting these variables, the telecommunications company can plan to minimize churn by focusing on specific factors influencing customers. For example, suppose a high number of outages is related to a high probability of customer churn. In that case, the company can concentrate resources such as spending, staffing, and programming on relevant departments to lessen outages and, in turn, reduce churn.

A2. Analysis Goals

The primary goal of this analysis is to find the best combination of variables that can optimally predict the probability of customer churn. To accomplish this goal, I will select several relevant variables for the analysis, create a logistic regression model, refine it using several reduction methods, and evaluate its statistical and practical significance. Developing a model and identifying relevant factors can help determine what areas within the company could be improved and decrease churn.

Part II: Model and Programming Justifications

B1. Four Logistic Regression Assumptions

These are the four selected assumptions of logistic regression: the response variable is binary, there is independence of observations, non-multicollinearity, and a linear relationship between the continuous explanatory variables and the logit of the response variable.

The target variable for the analysis is ‘Churn,’ consisting of binary data (Yes and No), so the model is considered a binary logistic regression model. The dataset will be checked for duplicated values, but at this time, we can assume that each observation is an individual unique customer. Non-multicollinearity refers to the assumption that the explanatory variables do not indicate a strong correlation with each other since multicollinearity makes models unreliable. A linear relationship between the continuous explanatory variables and the logit of the response variable means that the model represents unbiased data and consistent coefficients that you can use with new data. This relationship between the logit of the response variable and continuous explanatory variable is observed in section F1 and Figure 29.

B2. Python Justification

I completed the data preparation and analysis using Jupyter Notebook in a Python environment. Python is my choice for programming because I am familiar with its syntactical rules, and I find Python more accessible to ‘read’ than R. Python helps me stay organized with straightforward annotation markers. At the same time, Jupyter Notebook allows me to write in Markdown. Markdown language helps make code more approachable by letting me separate the notebook into digestible sections. A second benefit of using Python is its wide array of packages, libraries, and modules available for computing formulas, customizing visualizations, and simplifying complicated computations. The following table shows the Python libraries and modules used in the programming code and how they were helpful for the logistic regression analysis.

Table 1

Python Packages and Uses

Package Name	Package Use in Analysis
NumPy	- used to work with arrays - mathematical functions for calculations
pandas	- create DataFrames - manipulate DataFrames - generate statistical summaries
matplotlib	- establish and customize subplots for visualizations - simple histograms and bar plots
seaborn	- customize boxplots, layered histograms, and heatmap for cross-tabulation
statsmodels	- create a fitted regression model with Logit() - check VIF scores - generate confusion matrix

B3. Logistic Regression Justification

Multiple logistic regression determines the maximum likelihood of an outcome against a combination of explanatory variables. One of logistic regression's assumptions is that the response variable should be dichotomous (binary) and could be re-expressed to 0 and 1 as required. Simultaneously, the model accepts explanatory variables that are either qualitative or quantitative. The selected response variable for this analysis is 'Churn' with Yes and No values that we can encode to 1 and 0, respectively. The explanatory variables to be examined have types ranging from categorical, binary, discrete, and continuous, making them appropriate for logistic regression. Thus, logistic regression modeling matches the data characteristics of the variables relevant to the predictive analysis the research question pursues.

Part III: Data Preparation

C1. Data Cleaning Goals

My primary data-cleaning goal is to prepare the dataset and make it structurally appropriate for the logistic regression model. I performed the following data cleaning steps in order.

1. Check for nulls and fix them if there are necessary changes (Figure 1).
2. Check for duplicates to ensure the dataset meets the unique observations assumption (Figure 2).
3. Compare the unique values of the dataset to the provided dictionary to ensure the correct categories and inputs are present (Figure 3).
4. Update any names that might be ambiguous and use the dictionary as a guide (Figure 4).
5. Address outliers for each variable individually and determine the treatment of outliers as needed (Figure 5). An instance of examining the outliers for 'Yearly_equip_failure' is shown in Figure 6.

Additional data preparation steps and the appropriate code can be found in section C4. Sample codes used in the data-cleaning process are found in the figures below. It is essential to prepare data since this step is where we can ensure that the data fitted to the model is accurate and truthful. Data cleaning also allows me to observe the data and note discrepant but truthful numbers, such as outliers with a wide range but also realistic values. These dataset characteristics are essential to keep in mind while developing the model. To see the data cleaning code and cleaned dataset, please visit the attached file named "D208_MultipleLogisticRegression.ipynb."

Figure 1

Code for Checking Nulls and Nulls Imputation

```
[25]: # Double check nulls for each row

      nulls = [print(column, df[column].isnull().sum()) for column in df if df[column].isna().sum() > 0]

      print(nulls)

      InternetService 2129
      [None]

[29]: # Replace nan values in 'InternetService' with 'None' inputs to match dictionary
      df['InternetService'].fillna('None', inplace=True)

      # Verify that None is in array and not nan
      df['InternetService'].unique()

[29]: array(['Fiber Optic', 'DSL', 'None'], dtype=object)

[31]: # Check nulls again
      (df.isnull().sum(axis=1) > 0).sum()

[31]: 0
```

Figure 2

Code for Checking Duplicates

```
[33]: df.duplicated().value_counts()

[33]: False    10000
      Name: count, dtype: int64
```

Figure 3

Code for Examining Then Comparing Unique Values with Dictionary

```
: # Check unique values against dictionary
  for col in df:
      print(col, ':', df[col].unique())
```

Figure 4

Code for Updating Ambiguous Names

Change from 'Item...' to respective dictionary names

```
# Establish dictionary to map survey names
name_dict = {'Item1': 'TimelyResponse', 'Item2':'TimelyFixes', 'Item3':'TimelyReplacements', 'Item4':'Reliability',
             'Item5':'Options', 'Item6':'RespectfulResponse', 'Item7':'CourteousExchange', 'Item8':'ActiveListening'}

# Rename the dataframe and change the variable names so it persists
df.rename(columns=name_dict, inplace=True)

print(df.info())
```

```
# Rename other ambiguous names
name_dict = {'Multiple': 'MultipleLines', 'Contract':'ContractType',
             'Tenure':'Tenure_months', 'InternetService':'InternetServiceType', 'Outage_sec_perweek': 'Outage_sec_perweek_avg',
            'MonthlyCharge': 'MonthlyCharge_avg', 'Bandwidth_GB_Year': 'Bandwidth_GB_Year_avg'}

# Rename the dataframe and change the variable names so it persists
df.rename(columns=name_dict, inplace=True)

print(df.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10000 entries, 0 to 9999
Data columns (total 50 columns):
```

Figure 5

Code for Boxplots to Observe Quantitative Variable Outliers

```
cont_var = ['Population', 'Children', 'Age', 'Income', 'Outage_sec_perweek', 'Email', 'Contacts',
            'Yearly_equip_failure', 'Tenure_months', 'MonthlyCharge', 'Bandwidth_GB_Year']

# Set figure options
fig, axes = plt.subplots(4, 3, figsize=(20, 20))
fig.subplots_adjust(hspace=.5, wspace=.25)

# For loop over list of variables to generate box plots
for var, ax in zip(cont_var, axes.flat):
    ax.set_label(var)
    sns.boxplot(data=df, x=var, ax=ax)

axes[3,2].remove() # Removes unused chart
# Show figure
plt.show()
```

Figure 6

Detailed Observation of ‘Yearly_equip_failure’ Outliers

Yearly_equip_failure Outliers & Approach

```
# Gets Q1 and Q3 values
q1, q3 = np.percentile(df['Yearly_equip_failure'], [25, 75])

# Calculate interquartile range
iqr = q3 - q1

# Calculate upper limit
upper = q3 + (1.5 * iqr)
print('Upper Limit:', upper)

# Set variable for count
upper_count = 0

# Count outliers in Equipment Failure
for x in df['Yearly_equip_failure']:
    if x > upper:
        upper_count += 1


print('Upper Outliers:', upper_count)

fail_range = df['Yearly_equip_failure'].max() - upper
print('Upper Outlier Range:', fail_range)
print(df['Yearly_equip_failure'].describe())
```

```
Upper Limit: 2.5
Upper Outliers: 94
Upper Outlier Range: 3.5
count    10000.000000
mean         0.398000
std          0.635953
min          0.000000
25%          0.000000
50%          0.000000
75%          1.000000
max          6.000000
Name: Yearly_equip_failure, dtype: float64
```

Yearly equipment failure outliers consists of only about 1% of the population in dataset. Additionally, the maximum value of 6 or once every two months, is a realistic number than can help in determining if this is an issue that could be related to churn. Let's keep yearly equipment failure unchanged.

C2. Dependent and Independent Variables Summary Statistics

The following table (Table 2) presents the dependent variable 'Churn' and the selected independent variables for the logistic regression model. These variables focus on customer demographic information, customer profiles, and interactions with the company.

Table 2

Dependent and Independent Variables (Type & Description)

Name	Variable Type	Data Type	Description
Churn	Dependent	Qualitative	Yes or No on whether the customer ended service in the past month
Area	Independent	Qualitative	Customer's area density type at sign-up
Age	Independent	Quantitative	Customers age at sign-up
Gender	Independent	Qualitative	The customer's stated gender
Income	Independent	Quantitative	Customer income at sign-up
Children	Independent	Quantitative	How many children did the customer have at the sign-up
Outage_sec_perweek_avg	Independent	Quantitative	In an average of seconds per week, there was an outage at a customer's neighborhood
Contacts	Independent	Quantitative	Number of times customer contacted technical support

Yearly_equip_failure	Independent	Quantitative	Number of times customer's equipment was fixed or reset in the past year
ContractType	Independent	Qualitative	Customer contract type
Port_modem	Independent	Qualitative	'Yes' or 'No' on customer owning a portable modem
Tablet	Independent	Qualitative	'Yes' or 'No' on customer owning a tablet
InternetServiceType	Independent	Qualitative	Type of internet service provider customer signed up with
MultipleLines	Independent	Qualitative	'Yes' or 'No' customer signed up for multiple lines
OnlineSecurity	Independent	Qualitative	'Yes' or 'No' customer signed up for online security service
OnlineBackup	Independent	Qualitative	'Yes' or 'No' customer signed up for online backup add-on
DeviceProtection	Independent	Qualitative	'Yes' or 'No' customer signed up for device protection add-on
TechSupport	Independent	Qualitative	'Yes' or 'No' customer signed up for tech support service
StreamingMovies	Independent	Qualitative	'Yes' or 'No' customers signed up for streaming movies
Tenure_months	Independent	Quantitative	Number of months the customer has stayed with the company
MonthlyCharge_avg	Independent	Quantitative	Average monthly charge per customer
Bandwidth_GB_Year_avg	Independent	Quantitative	Average yearly bandwidth usage per customer

Figure 7 shows the summary statistics for the response variable 'Churn.' Figure 8 consists of the percentages and counts of the qualitative variables, and Figure 9 shows the summary statistics for the qualitative variables.

Figure 7

'Churn' Variable Summary Statistics

```
## Dependent Variable Summary Statistics
print(df[['Churn']].describe())
counts = df['Churn'].value_counts()

percs = df['Churn'].value_counts(normalize=True).mul(100).round(1).astype(str) + '%'

print(pd.concat([counts,percs], axis=1, keys=['count', 'percentage']))
```

```
        Churn
count   10000
unique      2
top        No
freq     7350
       count percentage
Churn
No      7350      73.5%
Yes     2650      26.5%
```

The generated summary statistics are the frequencies and percentages of the Boolean values. This applies to all categorical variables (Hazra & Gogtay, 2016). The ‘count’ row tells us that the dataset has 10,000 observations, also called rows. The ‘unique’ row shows the unique categories in that specific variable. The response variable ‘Churn’ is categorical with Boolean values of ‘Yes’ and ‘No’. The top frequented value, or mode, for ‘Churn’ is ‘No.’ 2,650 out of the 10,000 customers in the dataset have canceled their service with the company in the last month.

Figure 8 and Figure 9 below show the other categorical variables included in the mode. We can read the generated statistics like Figure 7, where the row labeled ‘top’ shows us each variable's mode or most frequented value. ‘Freq’ is the number of observations of the mode value. In Figure 9, we see the count of these frequencies as percentages out of the dataset population. These percentages and counts will show themselves again in section C3.

Figure 8

Qualitative Variables Summary Statistics

```
cat_var = ['Area', 'Gender', 'ContractType', 'Port_modem', 'Tablet', 'InternetServiceType',
           'MultipleLines', 'OnlineSecurity','OnlineBackup', 'DeviceProtection', 'TechSupport',
           'StreamingMovies']
df[cat_var].describe()
```

	Area	Gender	ContractType	Port_modem	Tablet	InternetServiceType	MultipleLines	OnlineSecurity	OnlineBackup	DeviceProtection	TechSupport	StreamingMovies
count	10000	10000	10000	10000	10000	10000	10000	10000	10000	10000	10000	10000
unique	3	3	3	2	2	3	2	2	2	2	2	2
top	Suburban	Female	Month-to-month	No	No	Fiber Optic	No	No	No	No	No	No
freq	3346	5025	5456	5166	7009	4408	5392	6424	5494	5614	6250	5110

Figure 9

Qualitative Variables Counts and Percentages

```
# Counts and Percentages
for col in df[cat_var]:
    counts = df[col].value_counts()

    percs = df[col].value_counts(normalize=True).mul(100).round(1).astype(str) + '%'

    print("\n")
    print(pd.concat([counts,percs], axis=1, keys=['count', 'percentage']))
```

```
          count percentage
Area
Suburban   3346      33.5%
Urban      3327      33.3%
Rural      3327      33.3%

        count percentage
Tablet
No       7009      70.1%
Yes      2991      29.9%

              count percentage
OnlineBackup
No             5494      54.9%
Yes            4506      45.1%

           count percentage
Gender
Female      5025      50.2%
Male        4744      47.4%
Nonbinary    231       2.3%

                     count percentage
InternetServiceType
Fiber Optic           4408      44.1%
DSL                   3463      34.6%
None                  2129      21.3%

                  count percentage
DeviceProtection
No                 5614      56.1%
Yes                4386      43.9%

                count percentage
ContractType
Month-to-month   5456      54.6%
Two Year         2442      24.4%
One year         2102      21.0%

               count percentage
MultipleLines
No              5392      53.9%
Yes             4608      46.1%

             count percentage
TechSupport
No            6250      62.5%
Yes           3750      37.5%

            count percentage
Port_modem
No           5166      51.7%
Yes          4834      48.3%

                count percentage
OnlineSecurity
No               6424      64.2%
Yes              3576      35.8%

                 count percentage
StreamingMovies
No                5110      51.1%
Yes               4890      48.9%
```

Figure 10

Quantitative Summary Statistics

```
numeric_vars = ['Age', 'Income', 'Children', 'Outage_sec_perweek_avg',
    'Contacts', 'Yearly_equip_failure', 'Tenure_months',
    'MonthlyCharge_avg', 'Bandwidth_GB_Year_avg']
df[numeric_vars].describe()
```

	Age	Income	Children	Outage_sec_perweek_avg	Contacts	Yearly_equip_failure	Tenure_months	MonthlyCharge_avg	Bandwidth_GB_Year_avg
count	10000.000000	10000.000000	10000.0000	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000
mean	53.078400	39806.926771	2.0877	10.001848	0.994200	0.398000	34.526188	172.624816	3392.341550
std	20.698882	28199.916702	2.1472	2.976019	0.988466	0.635953	26.443063	42.943094	2185.294852
min	18.000000	348.670000	0.0000	0.099747	0.000000	0.000000	1.000259	79.978860	155.506715
25%	35.000000	19224.717500	0.0000	8.018214	0.000000	0.000000	7.917694	139.979239	1236.470827
50%	53.000000	33170.605000	1.0000	10.018560	1.000000	0.000000	35.430507	167.484700	3279.536903
75%	71.000000	53246.170000	3.0000	11.969485	2.000000	1.000000	61.479795	200.734725	5586.141370
max	89.000000	258900.700000	10.0000	21.207230	7.000000	6.000000	71.999280	290.160419	7158.981530

Figure 10 above shows the quantitative statistics for the numeric variables in the dataset. Here, we see the count (total rows count) mean, std, min, 25%, 50%, 75%, and max. Min and max are the minimum and maximum values found in each variable. The mean is the average value, meaning all the numeric values in the rows were summed for that variable and divided by the total count (10,000). Next is the std or standard deviation, which tells us the spread of data concerning the mean. 50% is the median value of the variable. The 25% and 75% percentages are the first and third quartiles, respectively.

Let's look at 'MonthlyCharge_avg' as an example. The mean value is 172.62, which tells us that, on average, most customers are charged about $173 for their telecommunications subscription. The standard deviation (std) is ~43 or about $43, so the spread from the average of ~$172 is a charge of ~$43 more or a charge of ~$43 less. Since the lower quartile is at ~$140, we can assume that 25% of the customers in the dataset are getting charged less than ~$140. Alternatively, the upper quartile (75%) tells us that 25% of the population is charged more than $200. The median (50%) at ~$167 is a very close value to the mean, so from here, we can assume that the spread of data has a normal distribution. These definitions can also be used for the rest of the numerical variables. More insight on spread and normality can be found in section C3 with the univariate and bivariate visualizations.

C3. Univariate and Bivariate Visualizations

Figure 11

Response Variable Univariate Visualization

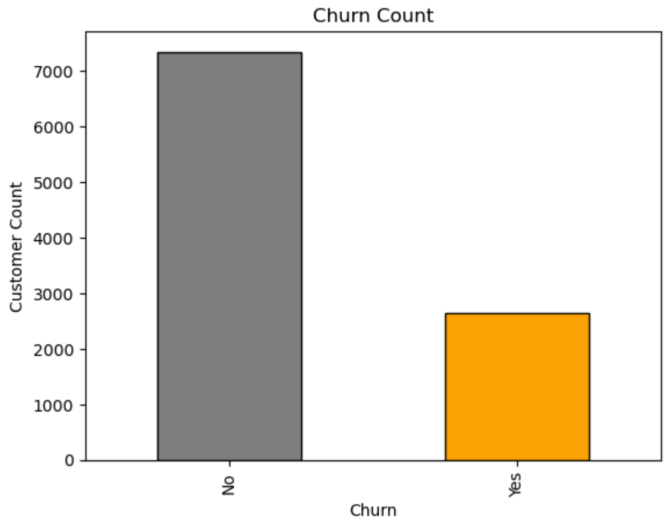
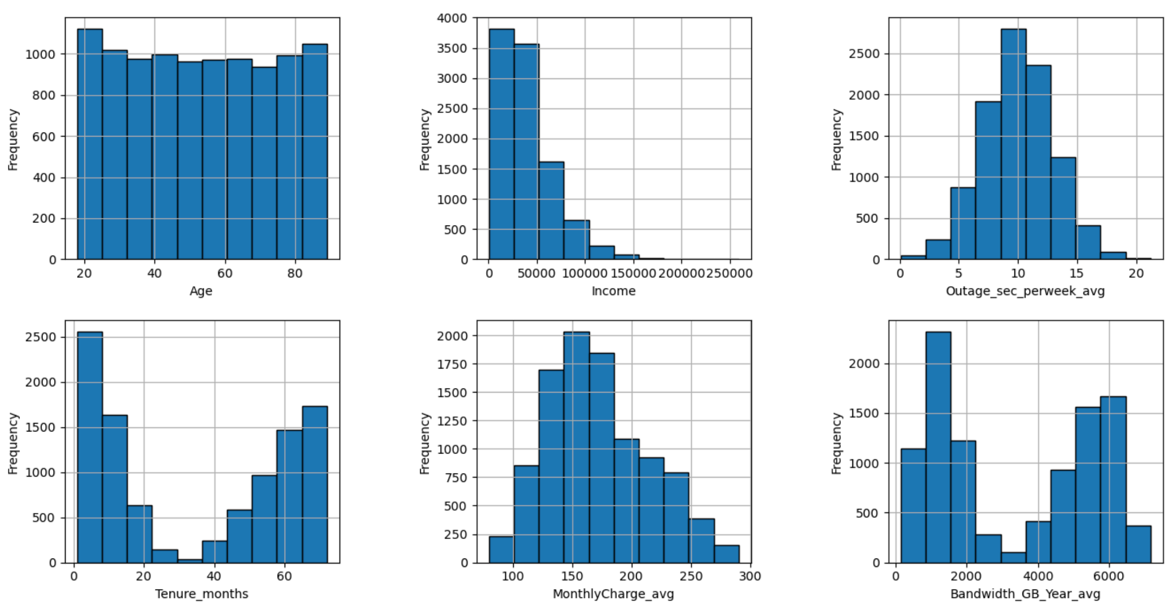


Figure 11 shows the customer count by the 'Churn' categories. Fewer customers in the dataset have churned in the past month than those who stayed with the company. This may become an issue with logistic regression modeling since we now see that the data is imbalanced. Further discussion on data imbalance can be found in sections F1 and F2.

Figure 12

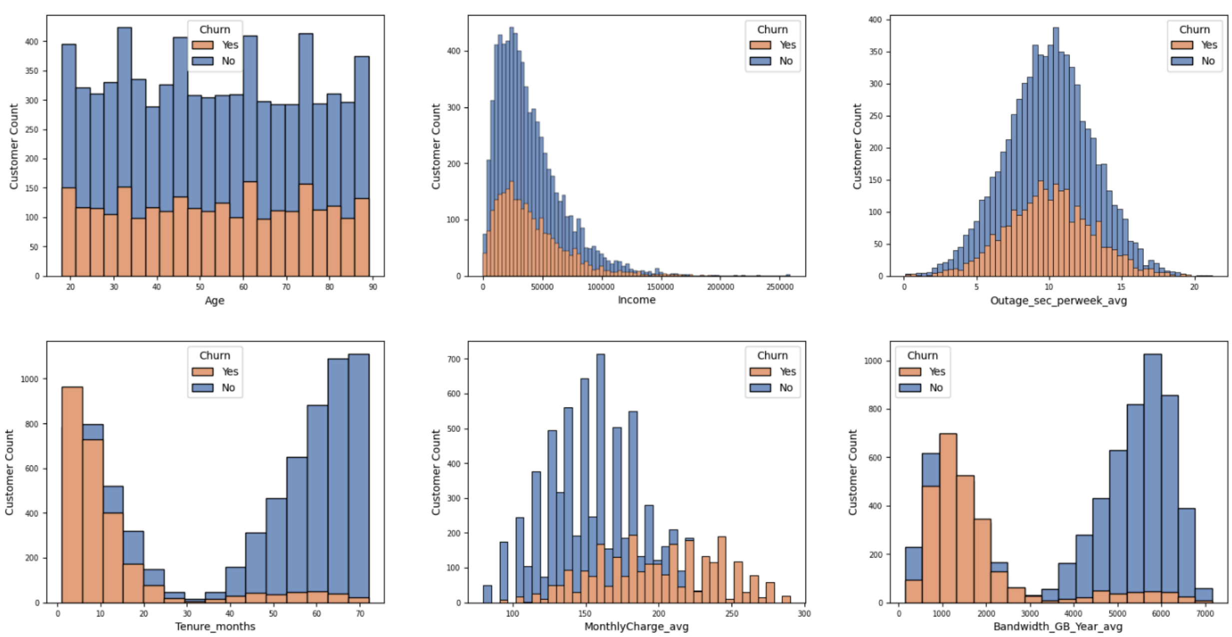
Univariate Visualizations for Continuous Variables



The univariate visualizations (Figure 12) show the distribution of each explanatory numeric variable. ‘Age’ demonstrates a uniform distribution, telling us that there is an approximately even distribution of customers aged 18 to 89 years old. ‘Income’ shows a positive or right-skewed distribution where most customers earn about $40,000. ‘MonthlyCharge_avg’ and ‘Outage_sec_perweek_avg’ roughly show a normal distribution with their bell-shaped curve. ‘Bandwidth_GB_Year_avg’ and ‘Tenure_months’ both indicate bimodal distribution. Bi-modal distribution tells us that two sub-groups might influence two means, which is discussed after viewing the bivariate statistics in Figure 13.

Figure 13

Bivariate Visualizations for Continuous Variables

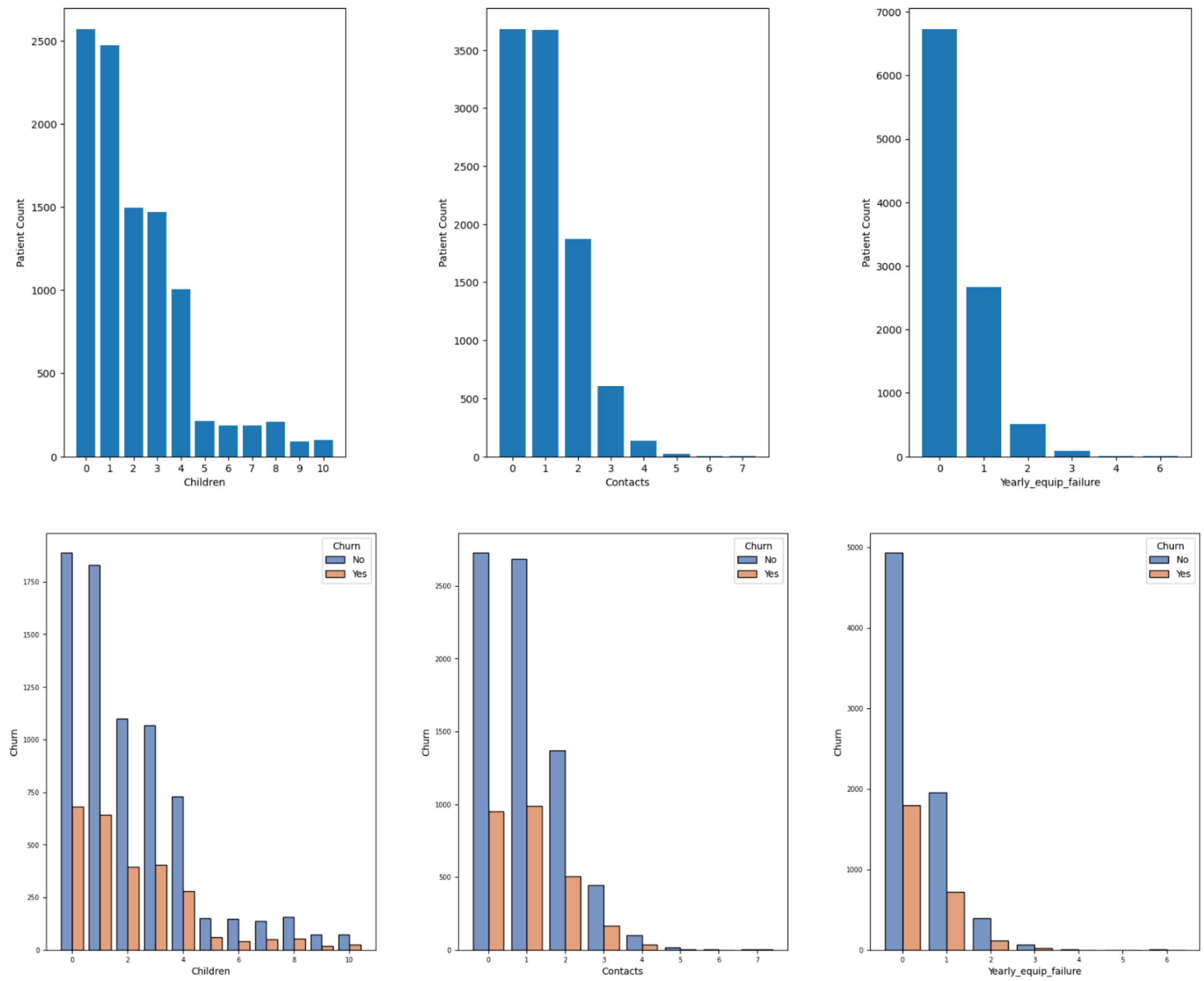


The bivariate histograms above split the dataset between the two categories in the ‘Churn’ variable. The two categories of ‘Yes’ and ‘No’ generally follow the same distribution as the univariate visualizations of ‘Age,’ ‘Income,’ and ‘Outage_sec_perweek_avg.’ We can also say that the ‘Yes’ or ‘No’ distribution in ‘MonthlyCharge_avg’ is approximately normal. However, we see that the mean and median of each group are different. More customers who did not churn had a monthly average charge of less than $225. Among those who churned, the average monthly charges range to $300. Based on the observation that there are two means, the bivariate distribution for 'MonthlyCharge_avg' and 'Churn' is bi-modal.

The 'Churn' split reveals the sub-groups within the bi-modal distribution of tenure and average yearly bandwidth. We see that most churned customers had a tenure of less than ten months compared to those that stayed loyal to the company, with a mean of around sixty-five months. The same observations apply to the 'Bandwidth_GB_Year_avg,' where less usage had a higher average of churned customers, and more usage had a higher average of loyal customers. This pattern is something to remember since we could assume that less time means less usage, and therefore, 'Bandwidth_GB_Year_avg' and 'Tenure_months' could demonstrate multicollinearity.

Figure 14

Univariate and Bivariate Visualizations for Discrete Variables



The figure above (Figure 14) shows univariate and bivariate visualizations for the selected discrete variables. Along the first row, we see all three discrete variables presenting a right-skewed distribution where most customers have three or fewer children, customers corresponded with the company once or twice, and most did not experience an equipment failure. The bivariate visualizations on the second row show a similar pattern where ‘Churn’ is broken down by its categories that follow a similar pattern.

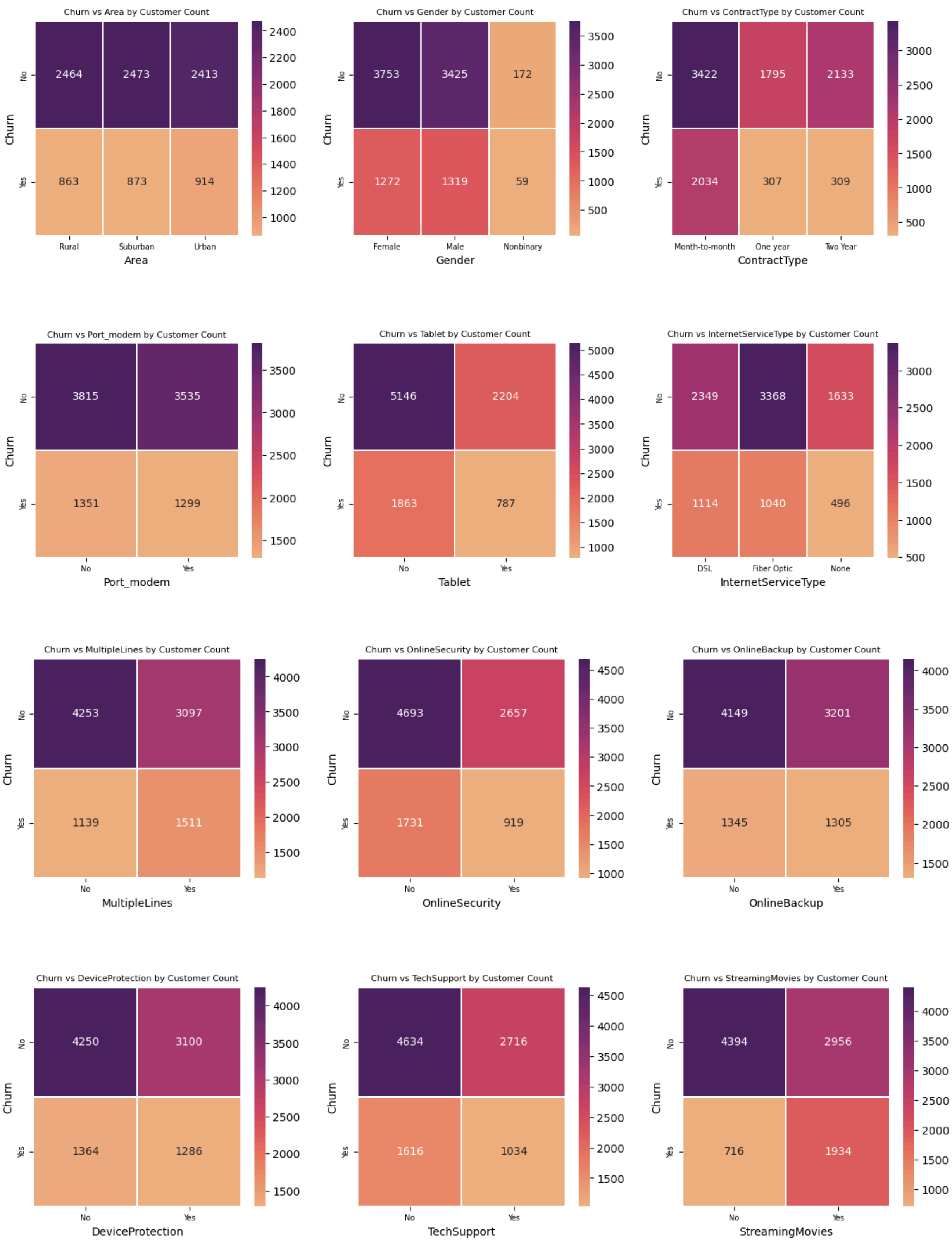
Figure 15

Univariate Visualizations for Categorical Variables

Figure 15 shows the breakout percentages of each category within each qualitative variable. We see that most of the categorical variables have Boolean values. 'Area,' 'Gender,' 'ContractType,' and 'InternetServiceType' have nominal categories.

Figure 16

Bivariate Visualizations for Categorical Variables



The bivariate visualizations above (Figure 16) show the cross-tabulation of each categorical predictor variable against the response variable. We can use the heat maps above to look at ratios between churned customers and the categories of each variable. Among those who churned, many had month-to-month contracts, did not own a tablet, signed up to stream movies, and did not sign up for online security or technical support.

C4. Data Transformation (Data Wrangling)

Performing data transformation across categorical variables is required for logistic regression modeling (Middleton, 2022a). The 'Churn' and explanatory variables with Boolean values of 'No' and 'Yes' should be re-expressed to 0 and 1, respectively. Other variables with nominal data should be re-encoded and regenerated as indicator variables or 'dummy' variables. Redundancy in the model is eliminated when dropping one column for each indicator variable. Figure 17 shows the code and outputs for the re-expressed variables. Figure 18 demonstrates one-hot encoding. Refer to the section with the heading "Data Wrangling" in the Jupyter Notebook file for more details. Figure 19 shows an overview of variables and data types of the wrangled data frame with only numeric data.

Figure 17

Re-expression of Categorical Variables

```
# Establish list of variables for categorical re-expression
reexp_cols = ['Churn', 'Port_modem', 'Tablet', 'MultipleLines', 'OnlineSecurity',
              'OnlineBackup', 'DeviceProtection', 'TechSupport', 'StreamingMovies']
```

```
# New column names variable
new_col_list = []

# Function for looping through target columns to re-express 'Yes/No' to 1/0
for col in reexp_cols:
    # Make new name and duplicate original column
    new_col = col +'_numeric'
    df[new_col] = df[col]

    # Make dictionary to change boolean to numeric
    new_dict = {new_col: {'No': 0, 'Yes': 1}}

    # Change all values in duplicated columns to ordinal encoding
    df.replace(new_dict, inplace=True)

    # Add to new_col_list
    new_col_list.append(new_col)

# Check that numeric columns were created by filtering and printing them

numeric_cols = [var for var in df.columns if 'numeric' in var]
print(df[numeric_cols].info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10000 entries, 0 to 9999
Data columns (total 9 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   Churn_numeric             10000 non-null  int64
 1   Port_modem_numeric        10000 non-null  int64
 2   Tablet_numeric            10000 non-null  int64
 3   MultipleLines_numeric     10000 non-null  int64
 4   OnlineSecurity_numeric    10000 non-null  int64
 5   OnlineBackup_numeric      10000 non-null  int64
 6   DeviceProtection_numeric  10000 non-null  int64
 7   TechSupport_numeric       10000 non-null  int64
 8   StreamingMovies_numeric   10000 non-null  int64
```

```
# Show unique values in new_col_list
for col in new_col_list:
    print(col, df[col].unique())
```

```
Churn_numeric [0 1]
Port_modem_numeric [1 0]
Tablet_numeric [1 0]
MultipleLines_numeric [0 1]
OnlineSecurity_numeric [1 0]
OnlineBackup_numeric [1 0]
DeviceProtection_numeric [0 1]
TechSupport_numeric [0 1]
StreamingMovies_numeric [1 0]
```

Figure 18

Generating Dummy Variables

```python
# Establish selection for dummy variable creation
target_dummy_vars = ['Area', 'Gender', 'ContractType', 'InternetServiceType']
# Create dummy variables and drop one level to remove redundancy
dummies = pd.get_dummies(df[target_dummy_vars], prefix=target_dummy_vars, drop_first=True, dtype=int)

dummies.head()
```

	Area_Suburban	Area_Urban	Gender_Male	Gender_Nonbinary	ContractType_One year	ContractType_Two Year	InternetServiceType_Fiber Optic	InternetServiceType_None
0	0	1	1	0	1	0	1	0
1	0	1	0	0	0	0	1	0
2	0	1	0	0	0	1	0	0
3	1	0	1	0	0	1	0	0
4	1	0	1	0	0	0	1	0

```python
# Add dummies columns to df and make change permanent by reassigning df
df = pd.concat([df, dummies], axis=1)
```

```python
# Create array for generated dummy variables and check that they made it to the dataframe
dummy_vars = ['Area_Suburban', 'Area_Urban', 'Gender_Male', 'Gender_Nonbinary', 'ContractType_One year',
             'ContractType_Two Year', 'InternetServiceType_Fiber Optic', 'InternetServiceType_None']

print(df[dummy_vars].info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10000 entries, 0 to 9999
Data columns (total 8 columns):
 #   Column                           Non-Null Count  Dtype
---  ------                           --------------  -----
 0   Area_Suburban                    10000 non-null  int64
 1   Area_Urban                       10000 non-null  int64
 2   Gender_Male                      10000 non-null  int64
 3   Gender_Nonbinary                 10000 non-null  int64
 4   ContractType_One year            10000 non-null  int64
 5   ContractType_Two Year            10000 non-null  int64
 6   InternetServiceType_Fiber Optic  10000 non-null  int64
 7   InternetServiceType_None         10000 non-null  int64
```

Figure 19

Wrangle Data Frame Overview Types and Variables

```python
# Drop original variables so all we have are the variables we will use for the logistic model

vars_todrop = ['Churn', 'Port_modem', 'Tablet', 'MultipleLines', 'OnlineSecurity',
              'OnlineBackup', 'DeviceProtection', 'TechSupport', 'StreamingMovies',
               'Area', 'Gender', 'ContractType', 'InternetServiceType']
df.drop(vars_todrop, axis=1, inplace=True)
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10000 entries, 0 to 9999
Data columns (total 26 columns):
 #   Column                           Non-Null Count  Dtype
---  ------                           --------------  -----
 0   Age                              10000 non-null  int64
 1   Income                           10000 non-null  float64
 2   Children                         10000 non-null  int64
 3   Outage_sec_perweek_avg           10000 non-null  float64
 4   Contacts                         10000 non-null  int64
 5   Yearly_equip_failure             10000 non-null  int64
 6   Tenure_months                    10000 non-null  float64
 7   MonthlyCharge_avg                10000 non-null  float64
 8   Bandwidth_GB_Year_avg            10000 non-null  float64
 9   Churn_numeric                    10000 non-null  int64
 10  Port_modem_numeric               10000 non-null  int64
 11  Tablet_numeric                   10000 non-null  int64
 12  MultipleLines_numeric            10000 non-null  int64
 13  OnlineSecurity_numeric           10000 non-null  int64
 14  OnlineBackup_numeric             10000 non-null  int64
 15  DeviceProtection_numeric         10000 non-null  int64
 16  TechSupport_numeric              10000 non-null  int64
 17  StreamingMovies_numeric          10000 non-null  int64
 18  Area_Suburban                    10000 non-null  int64
 19  Area_Urban                       10000 non-null  int64
 20  Gender_Male                      10000 non-null  int64
 21  Gender_Nonbinary                 10000 non-null  int64
 22  ContractType_One year            10000 non-null  int64
 23  ContractType_Two Year            10000 non-null  int64
 24  InternetServiceType_Fiber Optic  10000 non-null  int64
 25  InternetServiceType_None         10000 non-null  int64
dtypes: float64(5), int64(21)
```

Part IV: Model Comparison and Analysis

D1. Initial Model

Figure 20

Initial Logistic Regression Model & AIC/BIC

```
y = df[['Churn_numeric']]
X = df[expl_list].assign(const=1)

model_A = sm.Logit(y, X).fit()

print(model_A.summary())
```

```
Optimization terminated successfully.
         Current function value: 0.226081
         Iterations 9
                           Logit Regression Results
==============================================================================
Dep. Variable:          Churn_numeric   No. Observations:                10000
Model:                          Logit   Df Residuals:                     9974
Method:                           MLE   Df Model:                           25
Date:                Mon, 26 Aug 2024   Pseudo R-squ.:                  0.6090
Time:                        14:37:04   Log-Likelihood:                -2260.8
converged:                       True   LL-Null:                       -5782.2
Covariance Type:            nonrobust   LLR p-value:                     0.000
==================================================================================================
                                     coef    std err          z      P>|z|      [0.025      0.975]
--------------------------------------------------------------------------------------------------
Age                                0.0280      0.006      4.358      0.000       0.015       0.041
Income                          7.021e-07   1.34e-06      0.523      0.601   -1.93e-06    3.33e-06
Children                          -0.2481      0.060     -4.137      0.000      -0.366      -0.131
Outage_sec_perweek_avg            -0.0020      0.013     -0.160      0.873      -0.027       0.023
Contacts                           0.0555      0.038      1.453      0.146      -0.019       0.130
Yearly_equip_failure              -0.0291      0.060     -0.487      0.626      -0.146       0.088
Tenure_months                     -0.7966      0.153     -5.208      0.000      -1.096      -0.497
MonthlyCharge_avg                  0.0202      0.009      2.137      0.033       0.002       0.039
Bandwidth_GB_Year_avg              0.0084      0.002      4.474      0.000       0.005       0.012
Port_modem_numeric                 0.1109      0.076      1.467      0.142      -0.037       0.259
Tablet_numeric                    -0.0278      0.083     -0.336      0.737      -0.190       0.134
MultipleLines_numeric              0.3372      0.193      1.743      0.081      -0.042       0.716
OnlineSecurity_numeric            -0.8394      0.142     -5.904      0.000      -1.118      -0.561
OnlineBackup_numeric              -0.4562      0.091     -5.034      0.000      -0.634      -0.279
DeviceProtection_numeric          -0.5171      0.089     -5.811      0.000      -0.691      -0.343
TechSupport_numeric               -0.0222      0.135     -0.164      0.869      -0.288       0.243
StreamingMovies_numeric            0.4846      0.151      3.199      0.001       0.188       0.782
Area_Suburban                     -0.0650      0.093     -0.696      0.487      -0.248       0.118
Area_Urban                         0.0175      0.092      0.190      0.850      -0.164       0.199
Gender_Male                       -0.2876      0.139     -2.066      0.039      -0.560      -0.015
Gender_Nonbinary                   0.0429      0.261      0.164      0.870      -0.469       0.555
ContractType_One year             -3.2714      0.125    -26.172      0.000      -3.516      -3.026
ContractType_Two Year             -3.3611      0.122    -27.460      0.000      -3.601      -3.121
InternetServiceType_Fiber Optic    1.7436      0.958      1.820      0.069      -0.134       3.621
InternetServiceType_None           2.3171      0.658      3.521      0.000       1.027       3.607
const                             -7.7032      0.425    -18.109      0.000      -8.537      -6.869
==================================================================================================
```

```
print('Model A AIC: {}'.format(model_A.aic))
print('Model A BIC: {}'.format(model_A.bic))
```

```
Model A AIC: 4573.611588594536
Model A BIC: 4761.080438265917
```

The initial model (Figure 20) includes all selected, prepared, and transformed variables. The pseudo R^2 is ~ 0.6090. The LLR p-value, the overall statistical significance of the model, is at 0.00, below the alpha of 0.05, which tells us that this can be considered a meaningful model (Middleton, 2022b). The calculated AIC is 4,573.61, and the BIC is 4,761.08. We see that there are several variables with p-values that are above the preset alpha of 0.05. We also know that pseudo R^2 can increase when there are a lot of predictor variables (Middleton, 2022b). Thus, we approach the pseudo R^2 metric with some caution since we have a lot of variables for this initial model.

D2. Logistic Model Reduction Justification

At this point, the initial model has not been checked for non-multicollinearity, one of the assumptions for logistic regression. To be confident in the model's practicality, addressing multicollinearity is crucial because meeting this logistic regression assumption will make us confident in utilizing the model's coefficients and ensure stability (Larose & Larose, 2019). Figure 21 shows the code and output that recursively identifies high variance inflation factors (VIF) across the variables in the model and removes them. VIFs 10 and above show severe multicollinearity and should be removed (Choueiry, 2020). The remaining variables show moderate to little multicollinearity since we removed variables demonstrating severe multicollinearity with VIF above 10. See the sections "Addressing Multicollinearity" and "Reiterative High VIF Removal" in the Jupyter Notebook file to review the code in detail.

Figure 21

Addressing Multicollinearity

```python
for var in expl_list:
    max_val = vif_df['VIF'].max()
    max_name = vif_df['feature'][vif_df['VIF'] == max_val].values[0]
    max_index = vif_df['feature'][vif_df['VIF'] == max_val].index[0]

    if max_val > vif_threshold:
        print('Max Variable to Remove {}'.format(max_name))

        # Remove large VIF var in explanatory list
        expl_list.remove(max_name)
        X = df[expl_list]

        # Create dataframe to load variable name and VIF scores
        vif_df = pd.DataFrame()

        # Create column called feature to list all variables
        vif_df["feature"] = X.columns

        # Creates column called 'VIF' to list all corresponding VIF scores
        vif_df["VIF"] = [variance_inflation_factor(X.values, i) for i in range(len(X.columns))]
        print(vif_df)

print('New List: {}'.format(expl_list))
print('New List Length {}'.format(len(expl_list)))
```

```
Max Variable to Remove Bandwidth_GB_Year_avg
                              feature        VIF
0                                 Age   6.744327
1                              Income   2.882687
2                            Children   1.913136
3               Outage_sec_perweek_avg  10.009395
4                            Contacts   1.983514
5                 Yearly_equip_failure   1.385622
6                       Tenure_months   2.633943
7                    MonthlyCharge_avg  38.565635
8                  Port_modem_numeric   1.907348
9                      Tablet_numeric   1.421261
10             MultipleLines_numeric   2.634451
11            OnlineSecurity_numeric   1.559828
12              OnlineBackup_numeric   2.194206
13          DeviceProtection_numeric   1.913100
14               TechSupport_numeric   1.711834
15           StreamingMovies_numeric   3.839917
16                     Area_Suburban   1.954893
17                        Area_Urban   1.948956
18                       Gender_Male   1.919695
19                  Gender_Nonbinary   1.046673
20             ContractType_One year   1.379794
21             ContractType_Two Year   1.440926
22  InternetServiceType_Fiber Optic   2.596444
23          InternetServiceType_None   1.569729
Max Variable to Remove MonthlyCharge_avg
                              feature       VIF
0                                 Age  6.238181
1                              Income  2.814570
2                            Children  1.894265
3               Outage_sec_perweek_avg  8.615981
4                            Contacts  1.965223
5                 Yearly_equip_failure  1.381028
6                       Tenure_months  2.587289
7                  Port_modem_numeric  1.892072
8                      Tablet_numeric  1.413280
9              MultipleLines_numeric  1.812031
10            OnlineSecurity_numeric  1.538250
11              OnlineBackup_numeric  1.786453
12          DeviceProtection_numeric  1.761230
13               TechSupport_numeric  1.572985
14           StreamingMovies_numeric  1.922011
15                     Area_Suburban  1.924827
16                        Area_Urban  1.922607
17                       Gender_Male  1.898113
18                  Gender_Nonbinary  1.045387
19             ContractType_One year  1.368116
20             ContractType_Two Year  1.432591
21  InternetServiceType_Fiber Optic  2.168106
22          InternetServiceType_None  1.564996
```

Even more importantly, some variables with p-values surpass the alpha of 0.05, so they are not statistically significant for the model. Performing backward stepwise elimination as the reduction method will help to avoid overfitting due to too many variables that cause regression models to be unreliable and may not perform well in generalizing new data (*Forward Stepwise Variable Selection for Logistic Regression - Deciding on the Number of Variables*, n.d.). Overfitting affects models' consistency because too many factors can make a model fit too specifically to its training dataset, and it cannot accommodate new test data (Larose & Larose, 2019). Backward stepwise elimination works by generating a model and checking the explanatory variables' p-values. The next step would be to select the highest p-value, remove it, and then rerun the model to see the changes in p-values. This second step would run recursively to remove variables above the alpha threshold 0.05. P-values below 0.05 are statistically significant and should be kept in the model. Figure 22 presents the model reduction code. To view the recursive outputs, go to the "Model Reduction" section in the Jupyter Notebook file.

Figure 22

Model Reduction Code

```
# Define function for repetitive code
def pval_df(model):
    pvalues = pd.DataFrame(model.pvalues).reset_index()
    pvalues.columns = ['feature', 'pvalue']
    pvalues.drop(pvalues[pvalues['feature'] == 'const'].index[0], inplace=True)
    return pvalues
```

```
pvalues = pval_df(model_B)

print(pvalues)

# Set alpha
alpha = 0.05

# Establish list
updated_list = []

# Set counter for interation
count = 0

print('Model B before reduction:')
print(model_B.summary())

for x in pvalues['pvalue']:

    # Find maximum pvalue and its index in the dataframe
    max_pval = pvalues['pvalue'].max()
    max_name = pvalues['feature'][pvalues['pvalue'] == max_pval].values[0]
    max_index = pvalues['feature'][pvalues['pvalue'] == max_pval].index[0]

    # If pvalue is larger that 0.05...
    if max_pval > alpha:
        # Print name of variable to remove and add 1 to iteration count
        count += 1
        print('\n Iteration: {}'.format(count))
        print('Variable to Remove: {}'.format(max_name))

        # ... we remove it from dataframe
        pvalues.drop(index=max_index, inplace=True)

        # change the list to the reduced variables
        updated_list = pvalues['feature'].values

        # confirm it's been reduced
        print('\n Length of new list: {} \n'.format(len(updated_list)))

        print('Updated Model (without {}):'.format(max_name))

        # Re-run the linear regression with new list
        X = df[updated_list].assign(const=1)
        model = sm.Logit(y, X).fit()

        #Redo p-values
        pvalues = pval_df(model)

        print(model.summary())
```

D3. Reduced Logistic Regression Model

Figure 23

Reduced Logistic Regression Model & AIC/BIC

```
                           Logit Regression Results
==============================================================================
Dep. Variable:          Churn_numeric   No. Observations:                10000
Model:                          Logit   Df Residuals:                     9988
Method:                           MLE   Df Model:                           11
Date:                Mon, 26 Aug 2024   Pseudo R-squ.:                  0.4881
Time:                        16:10:15   Log-Likelihood:                -2959.9
converged:                       True   LL-Null:                       -5782.2
Covariance Type:            nonrobust   LLR p-value:                     0.000
====================================================================================================
                                       coef    std err          z      P>|z|      [0.025      0.975]
----------------------------------------------------------------------------------------------------
Tenure_months                       -0.0825      0.002    -42.190      0.000      -0.086      -0.079
MultipleLines_numeric                1.2328      0.069     17.969      0.000       1.098       1.367
OnlineBackup_numeric                 0.5885      0.067      8.834      0.000       0.458       0.719
DeviceProtection_numeric             0.3802      0.066      5.749      0.000       0.251       0.510
TechSupport_numeric                  0.1726      0.067      2.558      0.011       0.040       0.305
StreamingMovies_numeric              2.5879      0.079     32.852      0.000       2.433       2.742
Gender_Male                          0.2041      0.066      3.105      0.002       0.075       0.333
ContractType_One year               -2.4225      0.098    -24.597      0.000      -2.616      -2.229
ContractType_Two Year               -2.5448      0.096    -26.437      0.000      -2.733      -2.356
InternetServiceType_Fiber Optic     -1.0167      0.076    -13.416      0.000      -1.165      -0.868
InternetServiceType_None            -1.0387      0.092    -11.263      0.000      -1.219      -0.858
const                               -0.0869      0.100     -0.872      0.383      -0.282       0.108
====================================================================================================
```

```
print('Model B AIC: {}'.format(model_B.aic))
print('Model B BIC: {}'.format(model_B.bic))
```

```
Model B AIC: 5943.87930521158
Model B BIC: 6030.403389675294
```

The reduced model (Figure 23) consists of eleven explanatory variables demonstrating statistical significance based on p-values below 0.05. The pseudo R^2 is 0.4881. The LLR p-value is at 0.00, which indicates that this model exhibits an overall statistical significance. The calculated AIC is 5,943.88, and the BIC is 6,030.40.

E1. Initial Model and Reduced Model Comparison

Figure 24

Model A versus Model B

Logit Regression Results

Dep. Variable:	Churn_numeric	No. Observations:	10000
Model:	Logit	Df Residuals:	9974
Method:	MLE	Df Model:	25
Date:	Mon, 26 Aug 2024	Pseudo R-squ.:	0.6090
Time:	19:22:44	Log-Likelihood:	-2260.8
converged:	True	LL-Null:	-5782.2
Covariance Type:	nonrobust	LLR p-value:	0.000

	coef	std err	z	P>|z|	[0.025	0.975]
Age	0.0280	0.006	4.358	0.000	0.015	0.041
Income	7.021e-07	1.34e-06	0.523	0.601	-1.93e-06	3.33e-06
Children	-0.2481	0.060	-4.137	0.000	-0.366	-0.131
Outage_sec_perweek_avg	-0.0020	0.013	-0.160	0.873	-0.027	0.023
Contacts	0.0555	0.038	1.453	0.146	-0.019	0.130
Yearly_equip_failure	-0.0291	0.060	-0.487	0.626	-0.146	0.088
Tenure_months	-0.7966	0.153	-5.208	0.000	-1.096	-0.497
MonthlyCharge_avg	0.0202	0.009	2.137	0.033	0.002	0.039
Bandwidth_GB_Year_avg	0.0084	0.002	4.474	0.000	0.005	0.012
Port_modem_numeric	0.1109	0.076	1.467	0.142	-0.037	0.259
Tablet_numeric	-0.0278	0.083	-0.336	0.737	-0.190	0.134
MultipleLines_numeric	0.3372	0.193	1.743	0.081	-0.042	0.716
OnlineSecurity_numeric	-0.8394	0.142	-5.904	0.000	-1.118	-0.561
OnlineBackup_numeric	-0.4562	0.091	-5.034	0.000	-0.634	-0.279
DeviceProtection_numeric	-0.5171	0.089	-5.811	0.000	-0.691	-0.343
TechSupport_numeric	-0.0222	0.135	-0.164	0.869	-0.288	0.243
StreamingMovies_numeric	0.4846	0.151	3.199	0.001	0.188	0.782
Area_Suburban	-0.0650	0.093	-0.696	0.487	-0.248	0.118
Area_Urban	0.0175	0.092	0.190	0.850	-0.164	0.199
Gender_Male	-0.2876	0.139	-2.066	0.039	-0.560	-0.015
Gender_Nonbinary	0.0429	0.261	0.164	0.870	-0.469	0.555
ContractType_One year	-3.2714	0.125	-26.172	0.000	-3.516	-3.026
ContractType_Two Year	-3.3611	0.122	-27.460	0.000	-3.601	-3.121
InternetServiceType_Fiber Optic	1.7436	0.958	1.820	0.069	-0.134	3.621
InternetServiceType_None	2.3171	0.658	3.521	0.000	1.027	3.607
const	-7.7032	0.425	-18.109	0.000	-8.537	-6.869

Logit Regression Results

Dep. Variable:	Churn_numeric	No. Observations:	10000
Model:	Logit	Df Residuals:	9988
Method:	MLE	Df Model:	11
Date:	Mon, 26 Aug 2024	Pseudo R-squ.:	0.4881
Time:	16:10:15	Log-Likelihood:	-2959.9
converged:	True	LL-Null:	-5782.2
Covariance Type:	nonrobust	LLR p-value:	0.000

	coef	std err	z	P>|z|	[0.025	0.975]
Tenure_months	-0.0825	0.002	-42.190	0.000	-0.086	-0.079
MultipleLines_numeric	1.2328	0.069	17.969	0.000	1.098	1.367
OnlineBackup_numeric	0.5885	0.067	8.834	0.000	0.458	0.719
DeviceProtection_numeric	0.3802	0.066	5.749	0.000	0.251	0.510
TechSupport_numeric	0.1726	0.067	2.558	0.011	0.040	0.305
StreamingMovies_numeric	2.5879	0.079	32.852	0.000	2.433	2.742
Gender_Male	0.2041	0.066	3.105	0.002	0.075	0.333
ContractType_One year	-2.4225	0.098	-24.597	0.000	-2.616	-2.229
ContractType_Two Year	-2.5448	0.096	-26.437	0.000	-2.733	-2.356
InternetServiceType_Fiber Optic	-1.0167	0.076	-13.416	0.000	-1.165	-0.868
InternetServiceType_None	-1.0387	0.092	-11.263	0.000	-1.219	-0.858
const	-0.0869	0.100	-0.872	0.383	-0.282	0.108

The pseudo R^2 value in Model A was 0.6090, and in Model B, it was 0.4881 as shown in Figure 24. Based on the statsmodels Logit module, this metric is calculated using McFadden's pseudo R^2. The higher value indicates a better fit when comparing McFadden's pseudo R^2 between two models (*FAQ: What Are Pseudo R-squareds?*, n.d.). Model A demonstrates a higher pseudo R^2 than Model B. Next. We can consider the LLR p-value that indicates the overall statistical significance of a logistic model. A lower AIC and BIC demonstrate a better fit (Middleton, 22b). The calculated AIC and BIC for Model A are lower than Model B, so Model A indicates a better fit than Model B. For both models, the statistical significance is 0.00, below the alpha of 0.05, so we can determine that both models exhibit statistical importance. Another notable difference between the models lies in the individual variables' p-values. Model A includes variables with p-values over 0.05, demonstrating a weak statistical significance. Model B only consists of statistically significant values.

Overall, model A shows a better fit than Model B but is prone to overfitting because of its model's high number of variables, and some variables are not statistically significant. Model A also consisted of multicollinear variables that were statistically significant, with the possibility that the variables were affecting the LLR p-value. Alternatively, Model B offers a stronger statistically significant model. Still, the metric values across pseudo R^2, AIC, and BIC tell us it does not provide the best fit between the two models. However, it is arguable that Model B is better because it is less complex than Model A and could accommodate new data better.

The analysis above is further demonstrated with the model's confusion matrices and accuracy scores. Model A in Figure 25 shows a higher accuracy score of about 90%, while Model B's accuracy score is 86.5% (Figure 26). I hesitate to use the accuracy score as a valuable metric for this analysis because the data is skewed. There are more non-churned customers than churned customers, so a data imbalance is exhibited in the accuracy scores. For example, there are 7,350 true negatives in the dataset, which is a large volume of data that could generate a high number of chance predictions for TN instead of calculated predictions.

E2. Calculation Analysis Outputs

Figure 25

Model A Confusion Matrix and Accuracy Score

```
confusion_matrix_A = model_A.pred_table()
print(confusion_matrix_A)

# Extract TN, TP, FN and FP from confusion matrix
TN = confusion_matrix_A[0,0]
TP = confusion_matrix_A[1,1]
FN = confusion_matrix_A[1,0]
FP = confusion_matrix_A[0,1]


# Calculate accuracy
accuracy = (TN + TP) / (TN + TP + FN + FP)
print("Model A Accuracy: ", accuracy)
```

```
[[6904.  446.]
 [ 552. 2098.]]
Model A Accuracy:  0.9002
```

Figure 26

Model B Confusion Matrix and Accuracy Score

```
print(model_B.pred_table())
confusion_matrix_B = model_B.pred_table()

# Extract TN, TP, FN and FP from conf_matrix
TN = confusion_matrix_B[0,0]
TP = confusion_matrix_B[1,1]
FN = confusion_matrix_B[1,0]
FP = confusion_matrix_B[0,1]


# Calculate accuracy
accuracy = (TP + TN) / (TP + FP + FN + TN)
print("Model B Accuracy", accuracy)
```

```
[[6826.  524.]
 [ 823. 1827.]]
Model B Accuracy 0.8653
```

E3. Executable Python Code

To see all the code used for this analysis paper, please visit the Jupyter Notebook file titled "Multiple_Logistic_Regression_Notebook."

Part V: Data Summary and Implications

F1. Results of Data Analysis

Figure 27

Regression Equation

logit(p) = log(p/(1-p)) = -0.0869 – 0.0825 (Tenure by months) + 1.2328 (Multiple Lines) + 0.5885 (Online Backup Service) + 0.3802 (Device Protection Service) + 0.1726 (Tech Support Service) + 2.5879 (Movie Streaming Service) + 0.2041 (Male) – 2.4225 (One Year Contract) – 2.5448 (Two-Year Contract) – 1.0167 (Fiber Optic Internet) – 1.0387 (No Internet Service)

The logit(p) is where p is the probability of 'Churn' being 1. The coefficients on the right side of the equation in Figure 27 represent the variables with an influence on the log odds or logit of the successful probability of the response variable (Jankovic, 2023). The coefficients of the reduced model, as shown in Figure 27, are interpreted as follows:

- The coefficient of -0.0825 for tenure (by months) means that, holding all other variables constant, a one unit change in tenure (by months) causes the odds of churn to *change* by -8%.
- The coefficient of 1.2328 for having multiple lines means that, holding all other variables constant, having multiple lines compared to not having multiple lines, causes the odds of churning to *increase* by 243%.
- The coefficient of 0.5885 for having the online backup service means that, holding all other variables constant, having the online backup service compared to not having online backup service, causes the odds of churning to *increase* by 80%.
- The coefficient of 0.3802 for having the device protection service means that, holding all other variables constant, having the device protection service, compared to not having device protection, causes the odds of churning to *increase* by 46%.
- The coefficient of 0.1726 for having the tech support service means that, holding all other variables constant, having the tech support service, compared to not having tech support, causes the odds of churning to *increase* by 19%.
- The coefficient of 2.5879 for having the movie streaming service means that, holding all other variables constant, having the movie streaming service, compared to not having streaming service, causes the odds of churning to *increase* by 1230%.

- The coefficient of 0.2041 for being male means that, holding all other variables constant, being male, compared to not being male, causes the odds of *churning* to *increase* by 23%.
- The coefficient of -2.4225 for having a one-year contract means that, holding all other variables constant, having a one-year contract, compared to not having a one-year contract, causes the odds of churning to *decrease* by 91%.
- The coefficient of -2.5448 for having a two-year contract means that, holding all other variables constant, having a two-year contract, compared to not having a two-year contract, causes the odds of churning to *decrease* by 92%.
- The coefficient of -1.0167 for having fiber optic internet means that, holding all other variables constant, having fiber optic internet, compared to not having fiber optic internet, causes the odds of churning to *decrease* by 64%.
- The coefficient of -1.0387 for not having internet service means that, holding all other variables constant, not having an internet service (‘InternetService_None’ is 1), compared to having internet service (‘‘InternetService_None’ is 0), causes the odds of churning to *decrease* by 65%.
- When the explanatory variables above are 0, the log-odds of churning is -0.0869 and the odds of churning is -8%.

See Figure 28 to view the calculations of percentage changes.

Figure 28

Odds Percentage Calculations

```
# Calculate percentage odds

coeffs = model_B.params

for x, b in coeffs.items():
    perc = round((np.exp(b) - 1) * 100)
    print('{} : {}%'.format(x, perc))
```

```
Tenure_months : -8%
MultipleLines_numeric : 243%
OnlineBackup_numeric : 80%
DeviceProtection_numeric : 46%
TechSupport_numeric : 19%
StreamingMovies_numeric : 1230%
Gender_Male : 23%
ContractType_One year : -91%
ContractType_Two Year : -92%
InternetServiceType_Fiber Optic : -64%
InternetServiceType_None : -65%
const : -8%
```

As discussed in section E1, both the initial and reduced models had LLR p-values of 0.00, below the alpha threshold of 0.05, which tells us that the model overall is statistically significant. The main difference between the initial and the reduced models is that all the variables retained in Model B are statistically significant and less complex than in Model A. Regarding statistical significance, Model B edges out Model A because not only is the model statistically significant, but each variable in Model B has p-values below 0.05.

However, I cannot say that Model B is practically significant. Although the variables show vital significance in Model B, Model B has demonstrated a poorer fit than Model A by having lower values across AIC, BIC, and pseudo R^2. Model A's accuracy score was also higher than Model B's, and I say that with a caveat, knowing the dataset is not balanced. Model B's accuracy score is about 86.5%, indicating great model performance (Bobbitt, 2022). Due to the imbalance in the 'Churn' variable (Figure 30), I am not confident that this accuracy score is useful since accuracy does not consider how data is distributed (Bobbitt, 2022).

Finally, we could check all the assumptions listed in section B1 to help guide us in whether this reduced model is practical. We have confirmed that the observations are unique and independent during the cleaning stage and that the 'Churn' variable is dichotomous with 'Yes' and ‘No’ values that were re-expressed to 1 and 0, respectively. Multicollinear variables were treated and removed after the first model was generated. Figure 29 demonstrates whether the log odds of predictions for ‘Churn’ have a linear relationship with the continuous explanatory variable ‘Tenure_months.’ The scatterplot and correlation coefficient demonstrate a strong negative correlation with each other. After this observation, we can say that Model B meets all the assumptions we listed in section B1.

Figure 29

Linear Assumption Confirmation

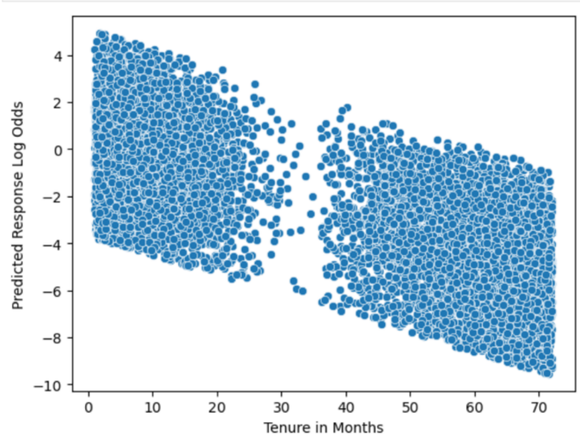
```
predictions = model_B.predict(X)
log_odds = np.log(predictions / (1 - predictions))

d = {'Predicted Response Log Odds' : log_odds,
     'Tenure in Months' : df['Tenure_months'],
     'Predictions': predictions}

# Create dataframe to graph
pred_df = pd.DataFrame(data=d)

sns.scatterplot(x = 'Tenure in Months' ,
                y = "Predicted Response Log Odds",
                data=d)
print(plt.show())

corr_coef = np.corrcoef(d['Tenure in Months'], d['Predicted Response Log Odds'])
print('Correlation Coefficient: {}'.format(corr_coef[0][1]))
```




```
None
Correlation Coefficient: -0.7407098313567145
```

Some inherent dataset characteristics severely limit our analysis. Both models can be more accurate and less biased if we address the data imbalance within the ‘Churn’ variable. The dataset has more 0s than 1s in for ‘Churn.’ The churn variable is imbalanced between 0 and 1, where 1 makes up only 25% of the population, with a high imbalanced ratio of ~36% (Figure 30).

Figure 30

Imbalance Ratio Calculation

```
# Check imbalance ratio
churn_count = df['Churn_numeric'].value_counts()
print(churn_count)

imbalance_ratio = churn_count[1] / churn_count[0]
print()
```

```
Churn_numeric
0    7350
1    2650
Name: count, dtype: int64
0.36054421768707484
```

Recognizing that this variable is time-based (churned in the last month), we could expand the dataset to previous months to balance the 'Churn' variable. Discussing this solution with someone with domain knowledge of the dataset is essential so we can confirm that increasing the population will influence refining the models. Other options to address data imbalance include other sampling techniques, such as random under-sampling, SMOTE, and random forest, which could increase performance (Rahman et al., 2020).

The sample size for 1 under the 'Churn' variable must be expanded to examine whether we can retrieve more precise accuracy scores and confusion matrices. This is one of the most straightforward ways we can adjust the model.

Other data characteristics that we need to consider as limitations are outliers. I did not remove or treat any outliers during the preparation and cleaning stage because the numbers were realistic for each variable. However, outliers could affect the model's precision, requiring domain knowledge based on specific explanatory variables. For example, outages of 7 could be an input error, but it would be best to confer with the development or technical services teams if such cases happen.

F2. Recommended Steps

Our research question is: *What factors in this telecommunications dataset can predict the probability of customers who are more likely to churn?* Our reduced Model B is less complex and consists of only statistically significant variables compared to our initial model, Model A. However, based on the pseudo R^2 metric, Model B has a worse fit than Model A. The imbalance ratio of the dataset and accuracy scores of both models also indicated that there may be some bias at play when it comes to predictions. Given what we know of the dataset and the bias of both models, I cannot confidently say we could use either model reliably to answer the research question.

The following recommended steps include further enhancing the data and considering other model approaches before moving on to business action items that address the research question. There are some valuable steps we can take immediately to refine Model B further, which involves expanding the dataset and determining a different reduction method approach. Since the dataset is imbalanced with a non-churn and churn ratio of 3:1, getting additional sample data of churned customers can help make a model more accurate. As mentioned in section F1, other sampling techniques, such as random sampling, random under-sampling, SMOTE, and random forest, could increase performance and minimize bias (Rahman et al., 2020).

Developing a model with a different reduction approach could help determine the most impactful and statistically significant variables. Perhaps it could lead to a better model. For example, doing forward stepwise variable selection using the area under curve metric (AUC) of each variable could be more time-consuming, yet exploring this method and AUC metric could yield a better model fit (*Forward Stepwise Variable Selection for Logistic Regression - Forward Stepwise Variable Selection*, n.d.). This would involve recursively selecting the variable with the highest predictor score, adding it to the model, checking for multicollinearity, checking how the other regression metrics perform, and then redoing the steps to get the most effective model. Other featured reduction methods could also be explored to ensure we can select the most statistically significant model and one that we can establish as best performing and accurate.

Part VI: Demonstration

G. Third-Party Code References

1. GeeksforGeeks. (2024, May 23). *Plot multiple histograms on the same plot with Seaborn*. GeeksforGeeks. https://www.geeksforgeeks.org/plot-multiple-histograms-on-same-plot-with-seaborn/
2. *How to Check Linearity Assumption in Logistic Regression with a Large Dataset?* (n.d.). Cross Validated. https://stats.stackexchange.com/questions/626068/how-to-check-linearity-assumption-in-logistic-regression-with-a-large-dataset
3. Kennethleungty. (n.d.). *Logistic-Regression-Assumptions/Box-Tidwell-Test-in-R.ipynb at main · kennethleungty/Logistic-Regression-Assumptions*. GitHub. https://github.com/kennethleungty/Logistic-Regression-Assumptions/blob/main/Box-Tidwell-Test-in-R.ipynb

4. *seaborn: statistical data visualization — seaborn 0.13.2 documentation*. (n.d.). https://seaborn.pydata.org/index.html
5. *statsmodels.discrete.discrete_model.LogitResults — statsmodels 0.8.0 documentation*. (n.d.). https://www.statsmodels.org/0.8.0/generated/statsmodels.discrete.discrete_model.LogitResults.html

H. References

1. Ahmad, A. K., Jafar, A., & Aljoumaa, K. (2019). Customer churn prediction in telecom using machine learning in big data platform. *Journal of Big Data*, *6*(1). https://doi.org/10.1186/s40537-019-0191-6
2. Bobbitt, Z. (2020, October 13). *The 6 Assumptions of Logistic Regression (With Examples)*. Statology. https://www.statology.org/assumptions-of-logistic-regression/
3. Bobbitt, Z. (2022, May 19). *What is a "Good" Accuracy for Machine Learning Models?* Statology. https://www.statology.org/good-accuracy-machine-learning/
4. Bobbitt, Z. (2024, February 12). *How to Interpret Log-Likelihood Values (With Examples)*. Statology. https://www.statology.org/interpret-log-likelihood/
5. Choueiry, G. (2020, June 1). *What is an Acceptable Value for VIF? (With References)*. https://quantifyinghealth.com/vif-threshold/
6. *FAQ: What are pseudo R-squareds?* (n.d.). https://stats.oarc.ucla.edu/other/mult-pkg/faq/general/faq-what-are-pseudo-r-squareds/#:~:text=A%20pseudo%20R%2Dsquared%20only,to%20the%20OLS%20R%2Dsquared.

7. *Forward stepwise variable selection for logistic regression - Deciding on the number of variables*. (n.d.). [Video]. DataCamp. https://campus.datacamp.com/courses/introduction-to-predictive-analytics-in-python/forward-stepwise-variable-selection-for-logistic-regression?ex=9

8. *Forward stepwise variable selection for logistic regression - Forward stepwise variable selection*. (n.d.). [Video]. DataCamp. https://campus.datacamp.com/courses/introduction-to-predictive-analytics-in-python/forward-stepwise-variable-selection-for-logistic-regression?ex=5

9. Hazra, A., & Gogtay, N. (2016). Biostatistics series module 4: Comparing groups - categorical variables. *Indian Journal of Dermatology/Indian Journal of Dermatology*, *61*(4), 385. https://doi.org/10.4103/0019-5154.185700

10. Jankovic, D. (2023, August 10). A simple interpretation of logistic regression coefficients. *Medium*. https://towardsdatascience.com/a-simple-interpretation-of-logistic-regression-coefficients-e3a40a62e8cf

11. Middleton, K. (2022a, November). *D208 - Webinar: Getting Started with D208 Part I (November 2022)* [Video]. Panopto. Retrieved July 13, 2024, from https://wgu.hosted.panopto.com/Panopto/Pages/Viewer.aspx?id=15e09c73-c5aa-439d-852f-af47001b8970

12. Middleton, K. (2022b, November). *D208 - Webinar: Getting Started with D208 Part II (November)* [Video]. Panopto. Retrieved July 28, 2024, from https://wgu.hosted.panopto.com/Panopto/Pages/Viewer.aspx?id=39bbe2db-de7d-4bf5-913ß

13. Rahman, H. a. A., Wah, Y. B., & Huat, O. S. (2020). Predictive Performance of Logistic Regression for Imbalanced Data with Categorical Covariate. *Pertanika Journal of Science & Technology*, *28*(4). https://doi.org/10.47836/pjst.28.4.02

14. Smith, G. (2018). Step away from stepwise. *Journal of Big Data*, *5*(1). https://doi.org/10.1186/s40537-018-0143-6

15. StatQuest with Josh Starmer. (2018, June 4). *Logistic Regression Details pt1: Coefficients* [Video]. YouTube. https://www.youtube.com/watch?v=vN5cNN2-HWE